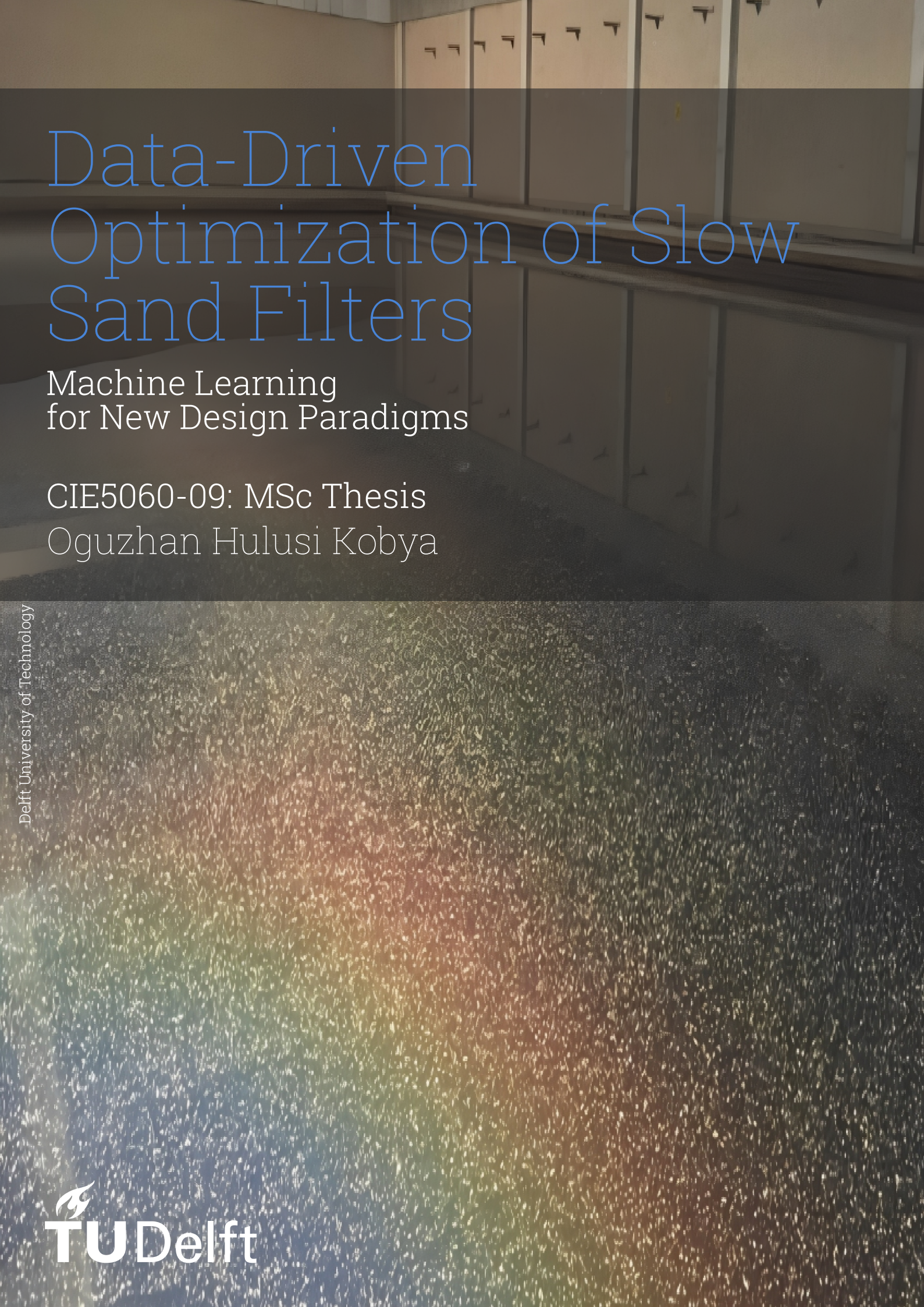

Data-Driven Optimization of Slow Sand Filters

Machine Learning for New Design Paradigms

CIE5060-09: MSc Thesis

Oguzhan Hulusi Kobya

Delft University of Technology

TU Delft

DELFT UNIVERSITY OF TECHNOLOGY

MASTER THESIS

Data-Driven Optimization of Slow Sand Filters: Machine Learning for New Design Paradigms

Author:
Oguzhan KOBYA
4734602

Supervisors:
Prof. dr. ir. Jan Peter van der Hoek
Prof. dr. ir. Doris van Halem
Prof. dr. ir. Luuk Rietveld
Prof. dr. Thom Bogaard
Greg Kyritsakas PhD

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Civil Engineering and Geo sciences

February 18, 2025

DELFT UNIVERSITY OF TECHNOLOGY

Abstract

Faculty of Civil Engineering and Geosciences
Department of Civil Engineering and Geo sciences

Master of Science

Data-Driven Optimization of Slow Sand Filters: Machine Learning for New Design Paradigms

by Oguzhan KOBYA
4734602

This thesis presents a comprehensive investigation into the optimization of Slow Sand Filter (SSF) performance through data driven modeling, with a focus on bacterial (E. coli, Coliform) and viral (Enterovirus, Adenovirus, Bacteriophage MS2) removal efficiencies. Combining an extensive literature review, exploratory data analysis (EDA), and machine learning modeling using XGBoost, this research provides new insights into the complex interactions between design parameters and SSF performance, ultimately contributing to the development of more efficient and sustainable SSF configurations.

The study began with a systematic literature review of 135 experimental studies, identifying key design parameters and operational conditions of SSFs, including Filtration Rate (FR), Hydraulic Retention Time (HRT), Effective Grain Size (D_{10}), Total Bed Height (T.B.H.), and *Schmutzdecke* characteristics. The literature review was supplemented with Exploratory Data Analysis (EDA) techniques such as scatterplots and correlation matrices to uncover trends and potential relationships between design parameters and removal efficiencies for bacteria and viruses. While the scatterplots and correlation matrices revealed limited linear relationships, they highlighted the complexity of the interactions, indicating the need for advanced modeling techniques capable of capturing nonlinear patterns.

To address these complexities, XGBoost, a powerful machine learning model known for its ability to handle nonlinear relationships and parameter interactions, was developed and applied to the dataset. The model's predictive capabilities were evaluated using performance metrics such as R^2, Mean Squared Error (MSE), and cross-validation results. Additionally, feature importance analysis was conducted using the XGBoost model, which provided insights into which design parameters and operational conditions had the most significant influence on removal efficiencies.

A key contribution of this research was the development of interactive tools, such as feature contribution sliders and interactive heatmaps, which enabled dynamic exploration of how changes in design parameters affect removal efficiencies. These tools facilitated the simulation of design scenarios, offering an intuitive understanding of parameter impacts and guiding the formulation of new design paradigms. The model outputs were used to propose optimized parameter ranges that improve bacterial and viral removal while maintaining operational efficiency.

The thesis further explored the robustness and reliability of the XGBoost models, discussing the results from controllable and uncontrollable design parameters and operational conditions and comparing the model's predictive performance for bacterial and viral removal.

In conclusion, this research highlights the effectiveness of XGBoost modeling for both predicting and explaining SSF performance. The insights gained provide a foundation for developing more efficient SSF design paradigms, enabling targeted adjustments to parameters based on model predictions. Additionally, the interactive tools created in this study offer practical resources for engineers to simulate different design configurations and optimize SSF performance. Overall, this thesis emphasizes the potential of machine learning to revolutionize the design and operation of SSFs, providing a foundation for future advancements in sustainable and effective water treatment technologies.

Acknowledgements

The completion of this thesis marks a significant milestone in my academic journey. My years at TU Delft have been a blend of fulfilling moments and challenging times. Despite the obstacles that arose along the way, I've been able to persevere and push through. I believe no, I know that the reason I managed to stand up every time and keep pushing myself to the limits is because of the unwavering support, guidance, and encouragement of many individuals, both within and beyond TU Delft. These individuals, knowingly or unknowingly, have played a pivotal role in helping me reach the end of this academic chapter.

First and foremost, I would like to express my deepest gratitude to my supervisors, professor Jan Peter van der Hoek and professor Doris van Halem, for their invaluable guidance, insightful feedback, and unwavering support throughout this research. Their expertise and dedication have been instrumental in shaping the direction and quality of this thesis. I am also grateful to them for believing in me and allowing me to embark on this journey of completing my MSc thesis while working almost full-time for the municipality of Amsterdam. Thank you both very much.

I am equally grateful to professor Luuk Rietveld for stepping in and filling the role of professor Jan Peter van der Hoek during his absence due to illness. His expertise and support allowed me to refine my modeling approach and complete my research in ways that would not have been possible otherwise. Thank you, professor, for your invaluable contributions. Luuk has also introduced me to Greg, whose support has been instrumental in successfully developing my machine learning models. I would like to extend my heartfelt thanks to Greg for his valuable contributions.

On a personal note, I am profoundly thankful to my family for their unconditional love, encouragement, and patience throughout this journey. Their belief in me has been a constant source of strength. Above all, my sister, Nisa, deserves special recognition. She has been a pillar of support throughout my life and continued to be so during the challenging moments of working on this thesis.

I would also like to thank my friends, both at TU Delft and beyond. Special thanks to Anass and Fettah for always being there, even during the most difficult moments of my studies. Your friendship and unwavering support have been a lifeline during the hardest times.

A heartfelt thank you to Asya Gokus, an incredible person I had the privilege of meeting this year. You have been the most supportive partner I could have asked for, and your quiet yet unwavering encouragement has not only driven me to complete my studies but also inspired me to aim higher in life. I hope you achieve every one of your dreams, just as you've helped me realize mine. And I look forward to us achieving the dreams we've set together, side by side.

Last, but certainly not least, I would like to thank my Lord and Creator, Allahü teâlâ, for granting me the capacity, strength, and opportunity to embark on this journey in the first place.

Contents

List of Figures

List of Tables

List of Abbreviations

QMRA	**Q**uantitative **M**icrobial **R**isk **A**ssessment
SSF	**S**low **S**and **F**ilter
E.Coli	**E**scherichia **C**oli
DOC	**D**issolved **O**rganic **C**arbon
AOC	**A**ssimilable **O**rganic **C**arbon
TOC	**T**otal **O**rganic **C**arbon
NOM	**N**atural **O**rganic **M**atter
PPCP	**P**harmaceuticals **P**ersonal **C**are **P**roducts
FR	**F**iltration **R**ate
HRT	**H**ydraulic **R**etention **T**ime
RSF	**R**apid **S**and **F**ilter
BSF	**B**io **S**and **F**ilter
OLS	**O**rdinary **L**east **S**quares
T.B.H	**T**otal **B**ed **H**eight
MSE	**M**ean **S**quared **E**rror
BIC	**B**ayesian **I**nformation **C**riterion
CV	**C**ross **V**alidation
DEC	**D**ecimal **E**limination **C**apacity
U.C.	**U**niformity **C**oefficient
TSS	**T**otal **S**uspended **S**olids

List of Symbols

d_{10}	Effective diameter [mm]
d_{60}/d_{10}	Uniformity coefficient [-]
r	Pearson correlation coefficient [-]
R^2	Coefficient of determination [-]
n	Number of data points or samples [-]
n_{leaf}	Number of leaf nodes in a decision tree [-]
n_{left}	Number of samples in the left child node [-]
n_{right}	Number of samples in the right child node [-]
T	Number of models or trees in an ensemble [-]
$\mathcal{L}$	Loss function [-]
$f_t(x)$	Prediction function of the decision tree at iteration t [-]
$r_i^{(t)}$	Residual of the i-th sample at iteration t [-]
g_i	First-order gradient of the loss function with respect to the prediction for the i-th sample [-]
h_i	Second-order gradient (Hessian) of the loss function with respect to the prediction for the i-th sample [-]
G_L, G_R	Sum of gradients for the left and right child nodes, respectively [-]
H_L, H_R	Sum of Hessians for the left and right child nodes, respectively [-]
λ	Regularization parameter for leaf weights [-]
γ	Regularization term for tree complexity [-]
$\Omega(f_t)$	Regularization term for the tree f_t [-]
k	Number of model parameters (features) [-]
$\hat{\sigma}^2$	Mean squared error (MSE) of the model [-]
L	Likelihood function [-]

Chapter 1

Introduction

1.1 Background Information

The availability of potable water is as essential as breathing oxygen for the human population. Due to increase in world population and urbanisation however, availability of clean water sources becomes an increasingly complex challenge, as can be seen in figure 1.1. The water that is available to the public often poses a significant threat to their health, as it comes from sources such as surface water, which contains contaminants and waterborne diseases. According to the World Health Organization, approximately 30 % of the world population does not have access to clean drinking water sources [1]. This resulted in the death of more than half a million people in low- and middle-income countries. To combat this, the World Health Organization has provided guidelines [2] that assess the quality of the drinking water. In accordance with the established guidelines, the Dutch Drinking Water Act mandates a quantitative microbial risk assessment (QMRA) to ensure that the annual infection risk remains below the threshold of one infection per 10,000 individuals [3] [4]. Every three years, a QMRA is conducted for the pathogens enterovirus, Campylobacter, Cryptosporidium and Giardia [4]. The influent and effluents of all drinking water treatment steps is monitored for the presence of both the previously mentioned pathogens and the associated indicator microorganisms (bacteriophages, Escherichia coli and clostridia). This is all done to make sure that the population of the Netherlands does not fall ill to waterborne diseases.

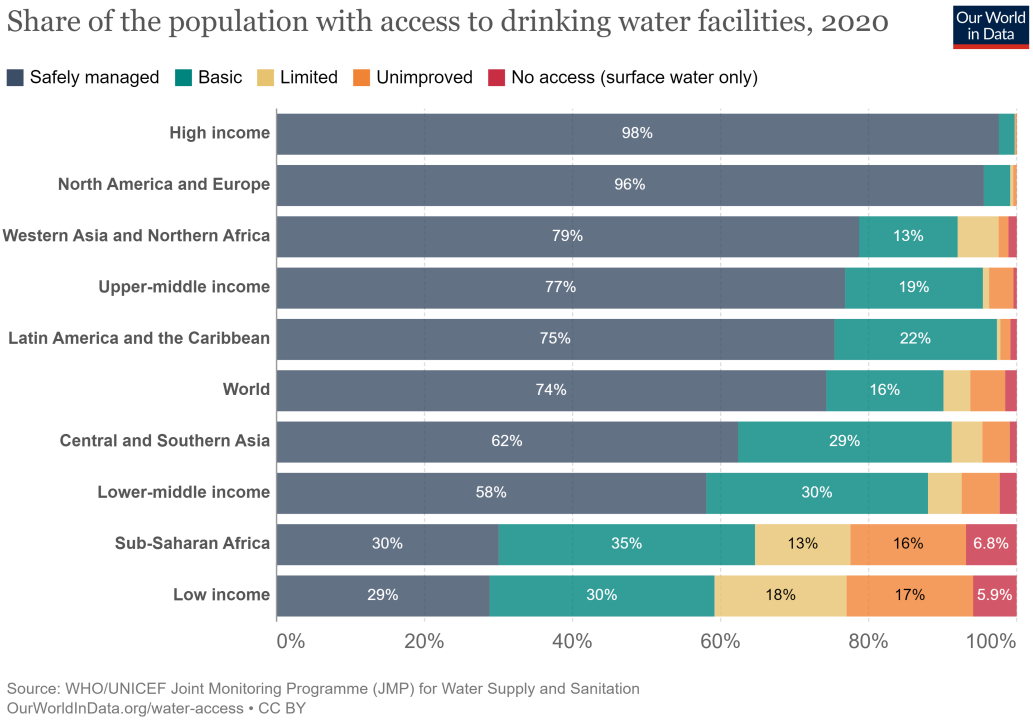

FIGURE 1.1: Share of population with access to drinking water facilities [5]

Several technologically advanced water treatment methods exist for the purification of polluted water, such as rapid sand filters and membrane filtration (microfiltration, ultrafiltration, nanofiltration, etc.). One of the more reliable, robust, attractive and low-investment cost drinking water technologies used for the removal of microorganisms, organic and particulate matter, and to create a biologically stable water, that prevents the growth of undesired pathogens, are Slow Sand Filters (SSFs) [6]. The use of SSFs is attractive as it does not require any chemicals of electricity for operation [7]. SSFs are often times present in the final stage of a drinking water treatment plant, after extended pre-treatment has taken place.

1.1.1 Removal of Contaminants in SSFs

As mentioned prior, SSFs are utilised for the purification of water contaminated by bacteria, sediments, and organic matter, et cetera. To achieve this goal, SSFs must target various types of contaminants, such as the pathogens bacteria and viruses, that can compromise the quality of drinking water. The removal of these contaminants can be seen as a boundary condition as not removing them can have severe consequences. Contaminants that need to be removed from drinking water include, but are not limited to, protozoa and larger organisms, several types of bacteria (such as E.Coli and Coliform), viruses (such as Enterovirus and Adenovirus), turbidity, organic contaminants (Dissolved Organic Carbon, Assimilable Organic Carbon, Total Organic Carbon) and inorganic contaminants (nutrients, iron) [1]. The removal of pathogenic contaminants is essential as these are known to cause waterborne diseases. An example of this is the larval form of Schistome Cercariae (flatworm), a protozoa. if this protozoa is ingested, it can develop into adult flatworms, which in turn result in the waterborne disease of schistosomiasis. Other contaminant types, such as turbidity and organic material, need to be removed, not only due to health risks associated with them, but also to make drinking water aesthetically pleasing and tasteful to the consumer. Furthermore, presence of organic material within distribution systems, such as Assimilable Organic Carbon (AOC), can lead to the regrowth of microbes, which in turn will lead to formation of biofilm and cause deterioration of the water quality.

The concentrations of contaminants such as protozoa, bacteria, viruses, and organic- and inorganic material are effectively reduced throughout the SSF. SSFs have proven to remove both turbidity and bacteria by 99 % to 99.9 % [8][9]. The removal for viruses is between 2 to 6 logs, which is similar to that of bacteria [10]. It must be noted that the characteristic of a SSF to remove turbidity, organic- and inorganic contaminants is not as consistent as for the removal of bacteria and viruses [1]. In terms of turbidity, studies have shown that the range of turbidity removal varies between 27 % and 97 % [1]. The reduction of turbidity in SSFs is strongly influenced by their design characteristics and operational conditions. These factors will be examined in detail in the following chapters.

The removal of organic- and inorganic contaminants has also been researched. Depending on the type of parameter that should be removed, such as TOC, DOC, or Pharmaceuticals Personal Care Products (PPCPs), filter characteristics and operational parameters, removal rates have shown significant differences [11].

The removal processes of several aforementioned contaminants can be visualized through the use of a breakthrough curve as can be seen in figure 1.2 [12]. The Y-axis in a breakthrough curve represents the logarithmic reduction of the concentrations of the contaminant $\log\left(\frac{C}{C_0}\right)$, while the X-axis typically represents the time or volume of water treated [13]. Using log reductions for removal efficiency provides a clear, standardized, and scientifically appropriate way to quantify and compare the performance of water treatment systems. This approach is particularly valuable when dealing with contaminant concentrations that vary over several orders of magnitude.

The removal of microorganisms in SSFs occurs primarily through adsorption and inactivation processes. Additionally, advection and dispersion significantly influence the distribution and transport of microorganisms within the filter bed. These mechanisms collectively contribute to reducing the effluent concentrations of microorganisms, enhancing the overall removal efficacy. However, the effectiveness of these processes is closely tied to various design parameters of an SSF, such as the filtration rate (FR), total bed height (T.B.H.), temperature, and effective grain size (D_{10}), which will be discussed in the subsequent sections.

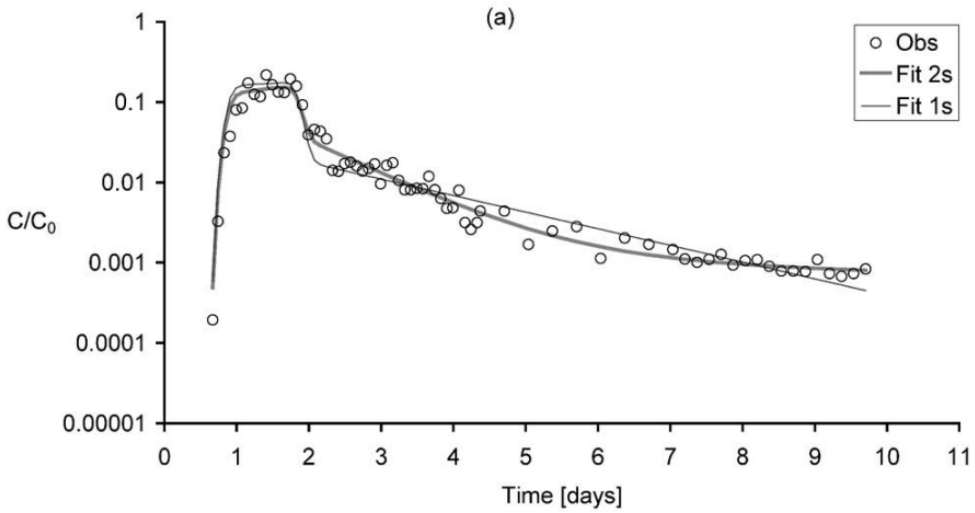

FIGURE 1.2: Measured breakthrough curve (open circles) and model of breakthrough curve of bacteriophages MS2 [12]

1.1.2 The Components of SSFs

A SSF usually consists of the following components: a supernatant water layer, a sand bed, a gravel layer and an outlet hose [14]. The components all serve a purpose. The supernatant water layer is a head of water that is used for driving the water through the filter bed, whilst also making sure that a certain retention period for the water can be achieved. This retention period is essential for biological (metabolic breakdown, predation, etc.), chemical and physical processes (diffusion, adsorption, sedimentation, etc.) to take place within the filter.

The filter bed is often times comprised of sand, as this type of filter medium is associated with lower costs, higher durability and availability. The water slowly percolates through the sand due to gravity, while several pollutants such as microorganisms, organic matter and inorganic particles are removed. The removal occurs both due to the physical filtration itself, but also due to biological degradation.

Although water undergoes treatment as it percolates through the sand bed, the majority of the purification process takes place at the uppermost layer of the sand bed, where biological and physical mechanisms are most active. Here, a biofilm layer forms during the first few weeks of operation, influenced by the influent water quality and driven by the presence of algal and particulate matter, along with dense biomass growth.Another term used for this biofilm layer is *Schmutzdecke* [15]. Water-borne microbes that pass through the filter are consumed by predatory bacteria, which are found in this layer. The growth of the microbial population in the *Schmutzdecke* is dependent on how much organic material is being supplied by in fluent water, as was stated previously. Thus, in raw water with high concentrations of contaminants, the *Schmutzdecke* is formed more rapidly. Bacteria are more active in the top of the sand filter, and this activity gradually decreases, as there is less organic material available in the lower parts of the SSF. Eventually, the surface of the filter will become clogged because of the accumulation of the microorganisms, inert particles and organic particles. This in turn leads to an increase in headloss, as there will be an increase in hydraulic resistance as [16]. Therefore, the *Schmutzdecke* must be periodically scraped off to maintain optimal filter performance. Afterwards, the ripening process of this biofilm layer begins anew and takes from 6 to 8 weeks to complete [17]. Without this process, the effectiveness of the SSF will go down immensely.

Even though the development of the *Schmutzdecke* is important for the disinfection capacity of the SSF, recent studies have shown that the presence of deep layered, active biofilm induced the removal of E.coli in a SSF, even after scraping off the *Schmutzdecke* [17].

Lastly, the gravel layer serves multiple purposes. Beneath the filter, a piping system is typically installed to transport the treated water. While the sand bed could potentially clog this system, the gravel layer prevents such blockages. Additionally, the gravel layer provides essential structural support for the sand bed, ensuring stability and effective filtration.

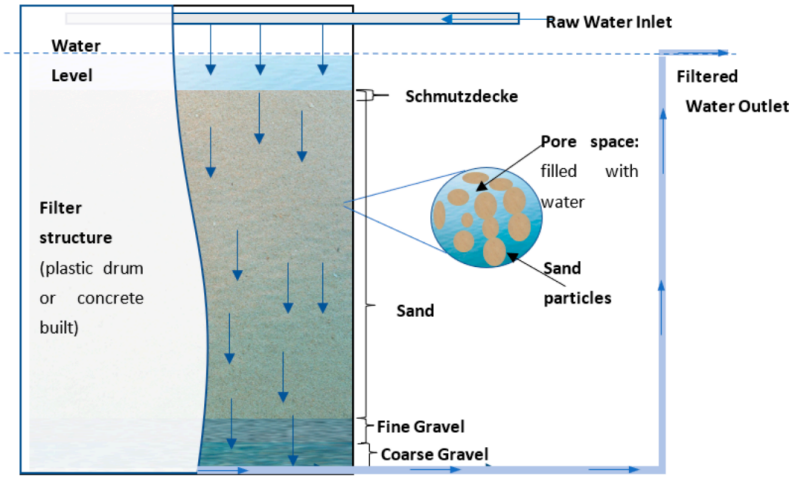

FIGURE 1.3: A General SSF Design [1]

1.1.3 Characteristics and Operational Conditions of a SSF

Regardless of the design parameters, an SSF should, at a minimum, achieve the following: reduce water turbidity (i.e., particle removal), decrease water color, reduce pathogen concentrations [1], and minimize the AOC content [18]. Additionally, it should effectively remove bacteria and viruses from the water. All SSFs must meet these boundary conditions. Additionally, more boundary conditions can be set for the SSFs, such as to be able to reduce the concentration of ammonium in the water.

The reduction of the aforementioned types of pollutants depends on several design parameters and operational conditions. Key design parameters, such as the FR, hydraulic retention time (HRT), D_{10}, T.B.H., supernatant water layer, bottom construction and scraping frequency, all play a significant role in determining the performance of the SSF. The quality of the feed water and the maturity of the biofilm layers also affect the quality of the effluent water [1] as mentioned prior. Other variables that affect the performance of a SSF and that cannot be controlled also exist, such as the temperature.

Some parameters have more significance when focusing on a specific type of pollution. An example of this was observed by Jenkins et al. [19], in which the researchers discovered that the removal of indicator bacteria was increased by between 0.16 and 0.30 logs through reducing the D_{10} from 0.52 to 0.17 mm. This supported the conclusion of Logan et al. [20], in which was stated that grain size is the most important parameter when focusing on the removal of cryptosporidium cysts and, possibly more pathogens. Another parameter, bed depth, was proven to be of little significance for the removal of Coliform bacteria [8]. When the bed depth was reduced from 0.97 m to 0.48 m, Bellamy et al., observed a reduction of 2 %. In the following chapters, the most important design parameters and their influences as stated in the literature will be discussed in greater detail.

1.2 Problem Statement

SSFs remain a crucial technology in drinking water treatment, valued for their ability to effectively remove contaminants through biological, physical, and chemical processes. However, despite their long history of use, the design and operation of SSFs are still largely based on outdated empirical data and generalized design guidelines. These guidelines often fail to consider whether the selected design parameter values and operational conditions truly result in the highest removal efficiencies. This resulted in suboptimal contaminant removal, overdimensioning of filters, and inappropriate cleaning strategies. Furthermore, the lack of clarity regarding the interplay between key design parameters, such as FR, HRT, D_{10}, T.B.H., and *Schmutzdecke* characteristics, limits the ability to optimize SSF systems for varying water quality conditions.

Addressing these limitations is essential for improving the overall performance and reliability of SSFs. Suboptimal design and operational practices can lead to inefficient contaminant removal, wasted resources, and unsustainable filter operation. In particular, low-loaded SSFs, where contaminant levels are minimal, present unique challenges: biofilm depletion can occur due to insufficient nutrient supply, compromising the filter's biological removal processes. Additionally, without proper optimization, maintaining the *Schmutzdecke* becomes difficult,

leading to inconsistent filtration efficiency. These inefficiencies not only reduce the reliability of SSFs but also result in increased water wastage during the development of the *Schmutzdecke*. Tackling these challenges is vital to ensuring that SSFs operate sustainably across varying conditions, providing consistent filtration performance and conserving valuable resources.

In conclusion, this study aims to bridge the knowledge gap by systematically analyzing the effects of key design parameters and operational conditions on SSF performance. By identifying optimal parameter combinations, this research seeks to provide actionable insights for maximizing removal efficiencies, improving operational sustainability, and mitigating common issues such as overdimensioning and inappropriate cleaning practices.

1.3 Research Question, Objective, and Scope

Research into the effects of key design parameters, such as T.B.H., FR, and D_{10}, will provide valuable insights into optimizing process conditions for SSFs, ultimately reducing both operational and investment costs. Additionally, a deeper understanding of the biological and physico-chemical processes within SSFs, including their relationship with filter depth, temperature, and other influencing factors, will contribute to the development of more effective design guidelines. The main research question of this MSc thesis is given below.

RQ: *"What range of values for design parameters and operational conditions are most compatible with SSFs, and ensure that the efficiency of pathogenic contaminant removal is not compromised? "*

In order to answer the research question, the following sub-questions will need to be answered first:

- **SQ1:** *Which design parameters and operational conditions,* ***both controllable and uncontrollable****, are most influential for predicting bacterial (E.Coli, Coliform) removal efficiency in SSFs?*
- **SQ2:** *Which* ***controllable*** *design parameters are most influential for predicting bacterial removal efficiency in SSFs?*
- **SQ3:** *Which design parameters and operational conditions,* ***both controllable and uncontrollable****, are most influential for predicting virus (Enterovirus, Adenovirus, Bacteriophage MS2) removal efficiency in SSFs?*
- **SQ4:** *Which* ***controllable*** *design parameters are most influential for predicting virus removal efficiency in SSFs?*

This thesis consists of a comprehensive approach that spans multiple key areas. The scope of the study outlines the methodological steps and focus areas required to optimize the design and operational parameters of SSFs and consists of the following components:

- **Literature Review:** Reviewing existing research to understand the relationships between design parameters, operational conditions and removal efficiencies in SSFs.
- **Data Analysis:** Analyzing datasets from 135 experimental studies and detect parameter interactions (Exploratory Data Analysis).
- **Machine Learning Modeling:** Building and validating predictive models to estimate removal efficiencies based on parameter values.
- **Design Recommendations:** Formulating practical design guidelines to optimize SSF performance for bacterial and viral removal.

1.4 Thesis Overview

This thesis is structured into six chapters, each building on the previous to comprehensively address the research objectives. Chapter 2 outlines the methodology employed in this study, detailing data collection through a literature review and application of an XGBoost model. The focus of this chapter is to explain the steps taken and tools used, without delving into detailed analysis. Chapter 3 reviews the relevant literature and explores the influence of key design parameters and operational conditions of SSFs through the use of scatterplots, incorporating Exploratory

Data Analysis (EDA) to identify trends, patterns, and potential relationships found within the constructed dataset. Chapter 4 presents the findings derived from the models, highlighting key insights into patterns, parameter interactions, and removal efficiencies. This chapter uses graphs and heatmaps to effectively illustrate the results and demonstrate what the models reveal about SSF performance. Chapter 5 synthesizes these findings, answering the main research question while providing deeper insights into the relationships between design parameters and their effects on SSF performance. Furthermore, the Chapter discusses the limitations of this study. Lastly, Chapter 6 provides the concluding remarks and offers directions for future research.

Chapter 2

From Exploration to Analysis: Building the Framework

This chapter outlines the research approach used to analyze the effects of key design parameters and operational conditions on the performance of SSFs. The methodology began with a literature review to identify critical parameters. Next, data from 135 experimental studies were collected, cleaned, and prepared for analysis. EDA techniques, such as scatterplots and correlation matrices, were then used to uncover trends and interactions between parameters. To capture complex, nonlinear relationships, machine learning models (XGBoost) were developed and evaluated using performance metrics. Finally, insights from the analysis were interpreted and validated to ensure their relevance for practical SSF design optimization. The following sections provide a detailed explanation of each step.

2.1 Framing the Focus: Key Parameters and Research Steps

The methodology began with an extensive literature review to identify relevant SSF experiments that analyzed the effects of key design parameters and operational conditions. The focus of the review was on studies that included detailed conclusions regarding the removal efficiencies of contaminants, specifically bacteria (Escherichia coli, Coliform bacteria), viruses (Enterovirus, Adenoviruses, Bacteriophages) and oocysts (Cryptosporidium, Giardia). While data on bacteria and viruses were abundant, data on oocysts were limited or inconsistent across studies. Due to the lack of sufficient data, the focus was narrowed down to bacteria and viruses, which offered a robust dataset for analysis. The dataset compiled can be found in Appendix A. During the literature review, special attention was given to six design parameters that were commonly reported across different studies. These design parameters are shown in figure 2.1.

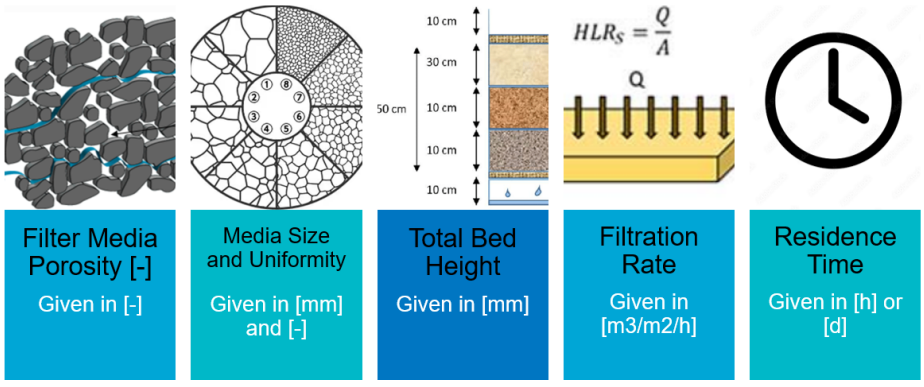

FIGURE 2.1: Pivotal design parameters in SSFs used for this research

These parameters were prioritized because they were consistently reported in studies and offered sufficient data for plotting and analysis. Beyond these primary parameters, other experimental conditions, such as pre-treatment types, influent characteristics, the uniformity coefficient (U.C.), whether the experiment was performed on lab-,

pilot- or full scale, and temperatures, were also recorded to provide context for the analysis. The comprehensive dataset included 135 experiments, offering a robust basis for exploring parameter effects.

The collected data was analyzed to uncover trends and interactions between design parameters and/or operational conditions. Scatterplots and other visualization techniques were employed to observe the relationships between design parameters and removal efficiencies. The focus was on evaluating the Decimal Elimination Capacity (DEC) for bacteria and viruses to identify patterns in filtration performance. For example, the analysis covered:

- The effect of increasing FR on contaminant removal.
- How porosity (n [-]) and D_{10} combinations influenced efficiencies.
- Whether a higher T.B.H. consistently improved performance.

Factors such as temperature, *schmutzdecke* development, and influent quality were often explored to explain discrepancies.

After compiling and analyzing the dataset through scatterplots, the final phase of the methodology focused on developing predictive models. Machine learning techniques, including Random Forest and XGBoost, were applied to uncover nonlinear relationships between design parameters and removal efficiencies. These models were chosen for their ability to handle complex interactions and provided insights into the relative importance of each parameter.

The models were validated using performance metrics such as R^2 and feature importance analyses. They revealed key interactions among parameters, such as the combined effects of FR, D_{10}, and T.B.H., that were not evident from linear analyses of scatterplots. These insights were further cross-validated against findings from the literature to ensure reliability and practical relevance.

In summary, the methodology combined literature insights, rigorous data analysis, and advanced modeling to derive meaningful conclusions about the optimization of SSF design 2.2. This iterative approach ensured that the findings were robust, scientifically grounded, and applicable to real-world conditions.

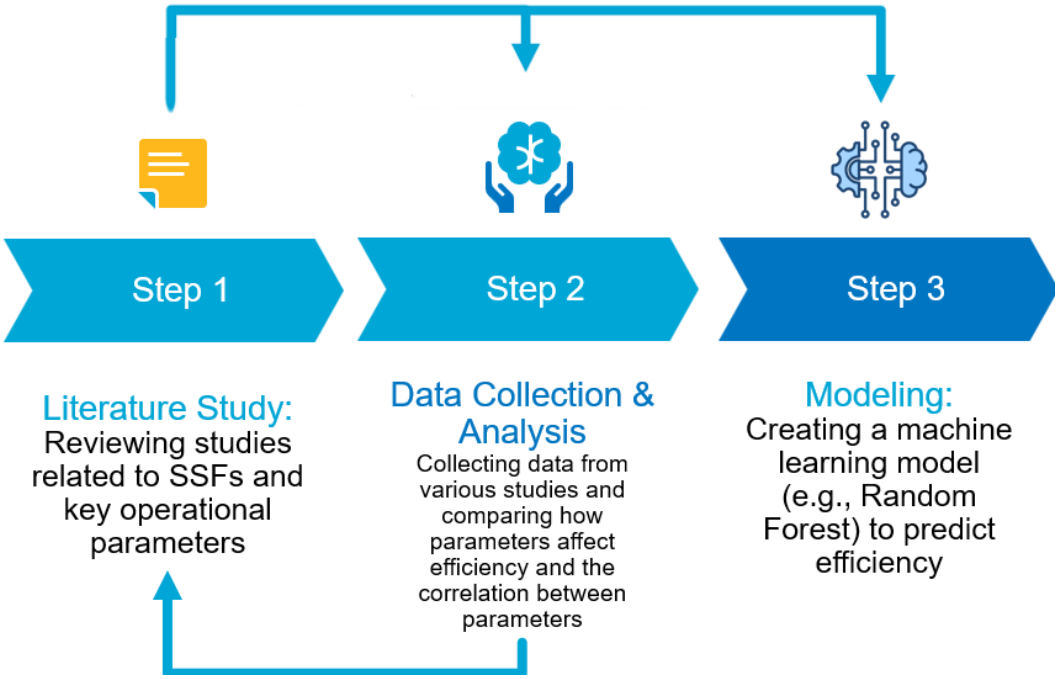

FIGURE 2.2: Methodology Overview of this MSc Thesis Research

2.2 Decoding Relationships: Insights Through Correlations and Scatterplots

The data analysis began with the construction of a correlation matrix to identify potential linear relationships between key design parameters or operational parameters, and the removal efficiencies of bacteria and viruses. A correlation matrix is a tabular representation of pairwise correlation coefficients, which quantify the strength and direction of linear relationships between variables. In this study, the matrix was created using the Pearson correlation coefficient, a widely applied statistical measure for assessing linear dependence between two continuous variables [21]. The analysis aimed to reveal trends and associations that could guide further investigation into the interactions between design parameters and SSF performance.

The Pearson correlation coefficient (r) is calculated using the following formula:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.1}$$

Where:

- x_i and y_i represent individual data points for variables x (e.g., FR) and y (e.g., removal efficiency for bacteria),
- $\bar{x}$ and $\bar{y}$ are the means of x and y, respectively,
- n is the total number of data points.

The coefficient r ranges between -1 and +1, where values close to +1 indicate a strong positive linear relationship, values close to -1 indicate a strong negative linear relationship, and values near 0 indicate no linear correlation. The correlation matrix can be seen in figure 2.3. The matrix was generated using Python's pandas library within the Jupyter Notebook environment. The code can be observed in Appendix B.

Although the Pearson correlation is effective for identifying linear relationships, the analysis revealed mostly low linear correlations between design parameters and removal efficiencies. A few high correlations were also observed; however, these were found to be unreliable as they were based on a limited number of data points, leading to skewed and misleading results. For example, parameters like porosity had only 35 data points available from the literature, exaggerating the strength of the observed correlations and reducing their validity. These findings highlight the limitations of correlation matrices in capturing the complex interactions governing SSF performance. Many SSF processes involve nonlinear dependencies, where a parameter might initially enhance removal efficiency but show diminishing or even adverse effects beyond a certain threshold. For instance, an increase in grain size might improve filtration performance by reducing clogging initially but later decrease efficiency due to reduced surface area. To address these shortcomings, subsequent analyses shifted focus towards machine learning techniques, specifically designed to capture nonlinear patterns and complex parameter interactions. Additionally, scatterplots were employed to visually inspect trends and further validate the findings. This combined approach aimed to provide a more robust and reliable understanding of the relationships between design parameters and removal efficiencies.

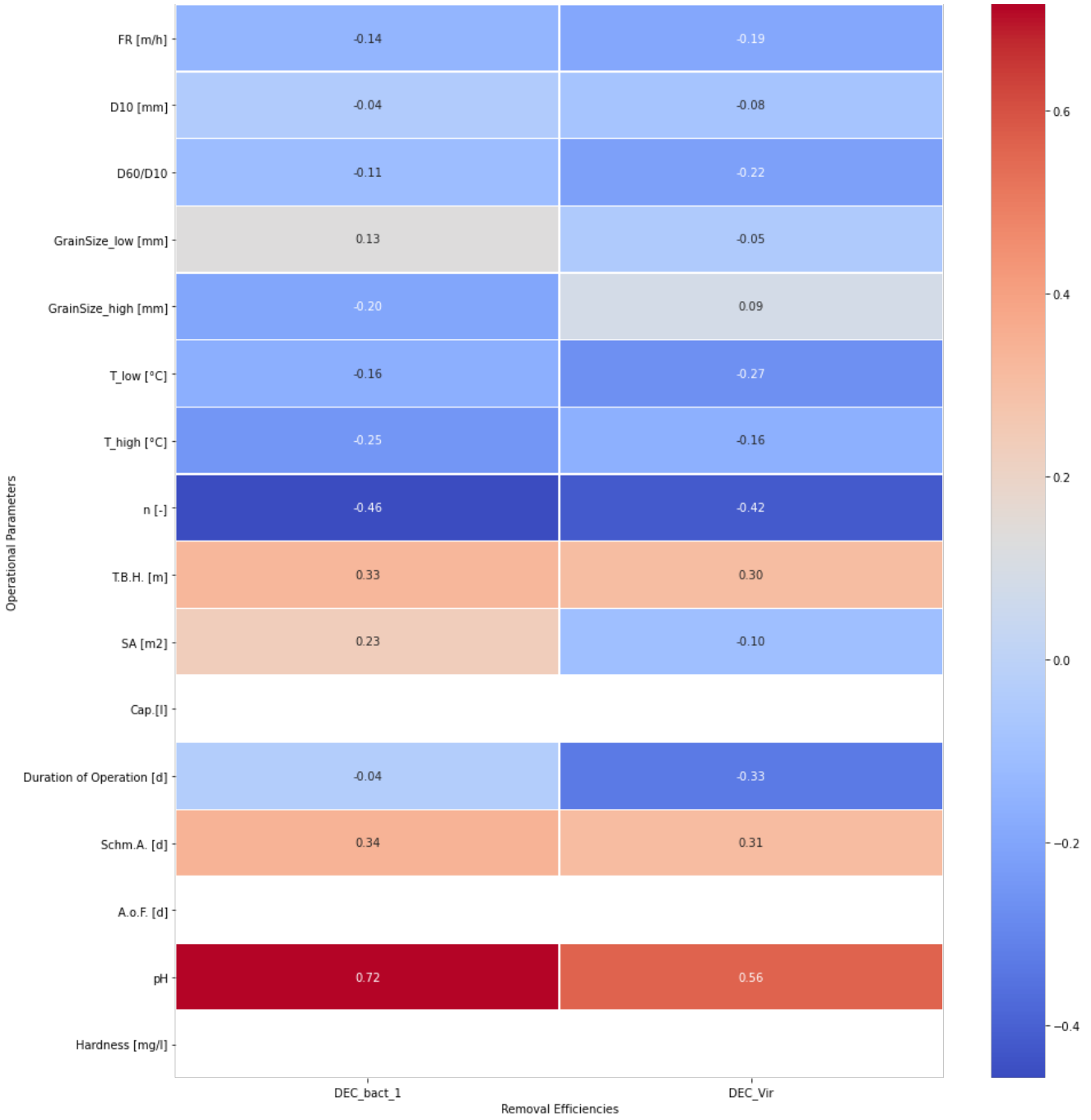

FIGURE 2.3: Correlation Matrix between Design Parameters and Removal Efficiencies of Bacteria and Viruses created by using data from the dataset given in Appendix A

After examining the relationships between the design parameters and removal efficiencies using a correlation matrix, the next step involved analyzing scatterplots. Scatterplots are a powerful visual tool for identifying patterns, trends, and potential correlations between variables, particularly when linear relationships are not evident or when the dataset is limited [22]. This method allows for a more intuitive understanding of the data by displaying individual data points in a two-dimensional space or three-dimensional space.

Each scatterplot was constructed with one or more design parameters on the axes (e.g., FR, D_{10}, or HRT) and the removal efficiency of contaminants (bacteria or viruses) on the y-axis or z-axis. This approach enabled the detection of potential nonlinear relationships, clusters, or outliers that could not be captured using a correlation matrix.

The scatterplots provided valuable insights into the relationships between design parameters and/or operational conditions and removal efficiencies. The scatterplots can be observed in Chapter 3, 'Integrating Literature and Data'. While 2D scatterplots effectively revealed trends between individual parameters and removal efficiencies, 3D and 4D scatterplots proved less useful due to data loss caused by missing values. This limitation underscored the need for quantitative methods, leading to the adoption of machine learning models to better capture nonlinear interactions and multifactorial effects.

2.3 Modeling Approaches: From Regression Analysis to Machine Learning

A two-step modeling approach was employed to understand and predict the removal efficiencies of bacteria and viruses in SSFs. The first step involved the use of Ordinary Least Squares (OLS) regression models, a traditional statistical technique often used to explore linear relationships between design parameters and removal efficiencies. OLS was initially chosen for its simplicity, interpretability, and ability to provide insights into the relationships between variables. However, the analysis revealed several limitations. The models had issues with overlapping variable effects, unstable results, sensitivity to missing data, and difficulty capturing complex non-linear relationships, as there were none, based on information from the correlation matrix. Additionally, the regression models were built on small subsets of the data, often containing as few as 30 datapoints per model, which is insufficient to produce robust and reliable results. Despite achieving high R^2 values, these concerns highlighted potential limitations in the accuracy and generalizability of the OLS models. Therefore, the focus shifted towards machine learning models.

Several categories of machine learning models exist. These models can all be divided into four categories: Supervised learning, Unsupervised learning, Semi-supervised learning, and Reinforcement learning [23]. In this MSc thesis paper, the focus was on supervised learning models. Supervised learning models are a category of machine learning models that learn from labeled datasets, in which each training instance consists of feature variables paired with corresponding output variables (e.g., DEC_{BACT}) [23]. These models aim to discover patterns in the training data that allow them to generalize and make accurate predictions on unseen data. Since the primary objective of this thesis is to estimate removal efficiencies of bacteria and viruses in SSFs based on design parameters such as FR, D_{10}, and total T.B.H., supervised learning models were selected as the most appropriate approach.

Within the category of supervised learning, different types of machine learning models also exist, such as Decision Trees, Random Forest, AdaBoost, the k-Nearest Neighbors (KNN) technique, and XGBoost, each with distinct strengths and applications [23]. For this research, XGBoost was chosen as a machine learning model. XGBoost was chosen due to its ability to handle non-linear relationships and interactions between variables [24], its robustness against multicollinearity and overfitting through built-in regularization [25], and its capacity to manage missing data effectively. Moreover, XGBoost performs well even with relatively small datasets, providing superior predictive accuracy compared to OLS [26].

While the several, aforementioned supervised learning models were considered, they were ultimately less suited for this study. This had several reasons. For example, while Decision Trees are simple and interpretable, they are prone to overfitting [27]. A single decision tree is insufficient in capturing complex relationships. The other type of machine learning algorith, Random Forest, improves on Decision Trees by averaging multiple trees, reducing variance, but lacks the gradient-boosting optimization that XGBoost has [23]. Random Forest modeling also does not have built-in regularization, making it more prone to overfitting as well. AdaBoost, another supervised learning model, works well for classification but can struggle with noisy data and is less effective for regression problems [23]. Additionally, KNN technique, while effective for both classification and regression, does not naturally handle missing data, as the model relies on distance-based similarity measures [23]. The dataset presented in Appendix A contains a considerable amount of NaN values, thus this model was also not suited.
Lastly, a key practical reason for choosing XGBoost was its ease of implementation using Python's built-in machine learning libraries. The XGBoost package is well integrated within scikit-learn packages, providing optimized tools for training, hyperparameter tuning, and feature importance analysis. This ensured an efficient modeling process and allowed for efficient experimentation with different parameter configurations.

2.4 Understanding Decision Trees: Concepts and Components

2.4.1 Basis of an XGBoost Model: Decision Trees

In order to understand how the XGBoost model works, it's essential to start with its foundation: decision trees. A decision tree is a flowchart-like structure used for both classification and regression tasks [28][29][30], as can be seen in figure 2.4. In this structure, internal nodes represent decisions based on the value of a feature, branches represent the outcomes of these decisions, and leaf nodes represent the final predictions or classifications. The topmost node is known as the root node, and the tree is built by recursively partitioning the dataset based on feature values, a process known as recursive partitioning.

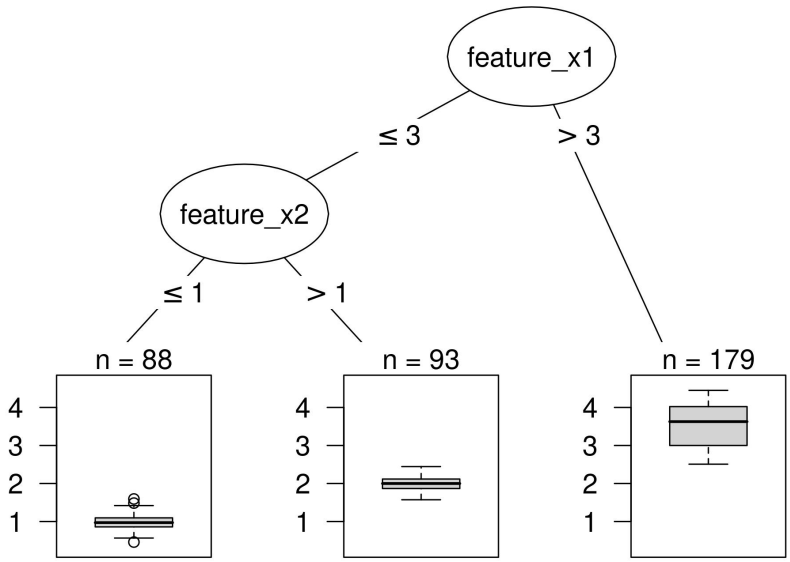

FIGURE 2.4: Example of a Decision Tree Splitting Data Based on Feature Thresholds and Displaying Target Distributions[31]

In order to better understand this concept, it will be explained, through the use of figure 2.4, by considering a scenario where the objective is to predict students' grades (on a scale of 1 to 4) based on two factors: the number of hours they study (`feature_x1`) and the number of practice tests they take (`feature_x2`).

The decision tree begins at the root node, which evaluates the number of hours a student studies (`feature_x1`):

- If the number of study hours is **3 or less** (≤ 3), the data is directed to the left branch.
- If the number of study hours is **greater than 3** (> 3), the data is directed to the right branch.

For students who studied 3 hours or less, the tree further evaluates the number of practice tests taken (`feature_x2`):

- If the number of practice tests is **1 or fewer** (≤ 1), the data is directed to the leftmost boxplot.
- If the number of practice tests is **greater than 1** (> 1), the data is directed to the middle boxplot.

For students who studied **more than 3 hours**, the data directly terminates in the rightmost boxplot, as no further splits are performed. Here, the decision tree does not perform additional splits in this particular example. This means that the model predicts grades for this group based on the observed data, without considering further factors like the number of practice tests taken. It's important to note that the tree's decision to stop splitting this group depends on the dataset and the algorithm's criteria (e.g., there may not have been significant variation in grades for this group based on other features). This does not necessarily imply that studying more hours always leads to better grades, as other unexamined factors could influence the outcomes, such as the quality of study time or practice tests.

The terminal nodes, represented by boxplots, summarize the distribution of grades for each subset:

- **Leftmost boxplot**: This represents students who studied 3 hours or less and took 1 or fewer practice tests ($n = 88$). These students tend to have lower grades, closer to 2.
- **Middle boxplot**: This represents students who studied 3 hours or less and took more than 1 practice test ($n = 93$). These students tend to have slightly higher grades, closer to 3.
- **Rightmost boxplot**: This represents students who studied more than 3 hours ($n = 179$). These students tend to have the highest grades, closer to 4.

This example illustrates how the decision tree splits data into smaller groups based on specific conditions, allowing for increasingly precise predictions of student grades as the tree progresses. This example will be referenced throughout the remainder of this chapter to provide a clearer understanding of how the developed XGBoost model operates.

2.4.2 Forming the Algorithm of the Decision Tree Model

The process of constructing a decision tree involves three main steps [28][30]. Considering the example of predicting student grades based on the number of hours they study and the number of practice tests they take, the algorithm identifies the optimal feature to split the dataset using Attribute Selection Measures (ASM), which evaluate how well each feature separates the data. In this case, the number of hours studied might be chosen as the first decision node, as it provides the best differentiation in students' grades.

Once identified, this feature becomes the decision node, dividing the dataset into smaller subsets, such as students who study more than three hours and those who study three hours or less. The tree-building process then continues recursively for each subset. For example, among students who study three hours or less, the algorithm might split the data further based on the number of practice tests taken, creating additional child nodes.

This process continues until one of the following conditions is met: all students in a subset have the same grade (e.g., all scored a 4), no additional features remain for splitting (e.g., practice tests and study hours have already been used), or the dataset becomes empty. This recursive approach ensures that the decision tree captures meaningful patterns in the data while dividing it into increasingly homogeneous groups.

In figure below 2.5, the algorithm for decision tree creation is visualized.

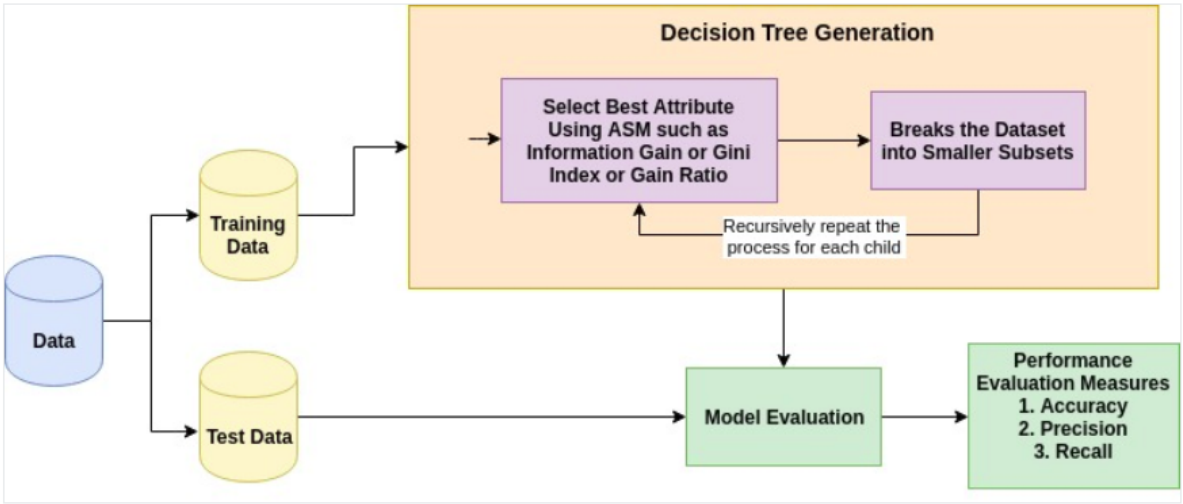

FIGURE 2.5: Overview of Decision Tree Construction and Evaluation: From Data Splitting to Performance Assessment. This example presents a classification problem [30]

As mentioned previously, ASM is a method used to determine the best way to split data at each step in a decision tree. It helps the algorithm identify which feature will create the most meaningful division of the data. For example, when predicting student grades, the decision tree might analyze features such as the number of hours studied and the number of practice tests taken. ASM assigns a score to each feature, ranking them based on how well

they separate the data. The feature with the highest score, such as the number of hours studied, will be selected as the splitting feature. If the feature is continuous, like hours studied, ASM also determines the best split point (e.g., "3 hours or less" versus "more than 3 hours"). This ensures that the tree divides students into groups that best predict their grades.

Common methods for scoring and selecting the best feature include [28][30]:

- **Information Gain**: Measures how much entropy is reduced by splitting on a particular feature.
- **Gain Ratio**: Adjusts Information Gain to account for the number of unique values in the feature.
- **Gini Index**: Measures how similar the resulting groups are after the split [28][32].

These measures ensure the tree makes effective splits, improving its ability to predict outcomes accurately.

Lastly, in the process of building a decision tree, the data is split into two parts: training data and test data, as can be seen in figure 2.5. This is a crucial step to ensure the model is accurate and generalizes well to new, unseen data [33]. The training data is used to build the decision tree. This is the portion of the data where the algorithm learns patterns and relationships between features (e.g., number of study hours, practice tests) and the target variable (e.g., grades). During training:

- The algorithm selects the best features for splitting the data using ASM.
- The dataset is repeatedly divided into smaller subsets, creating nodes and branches until the tree is complete or meets stopping criteria (e.g., minimum number of samples per node).

The test data is used to evaluate the decision tree after it has been trained. This dataset is kept separate from the training data and is not used during the learning phase. The purpose of the test data is to:

- Measure how well the decision tree performs on unseen data.
- Detect overfitting (when the model is too tailored to the training data and fails to generalize).

For example:

- The test data includes a new set of student records that the model has never seen before.
- The model predicts their grades based on the learned patterns, and the predictions are compared to the actual grades.

2.4.3 Application of Decision Tree Modeling to the Case Study

Now that the concepts of decision trees have been explained, a decision tree was implemented for the case study of SSFs, and more specifically for the dataset given in A using Jupyter Notebook. Jupyter Notebook provides built-in libraries and functions to create and train decision tree models efficiently and also build-in decision tree classifier model [30]. This eliminates the need to manually write the algorithm for constructing a decision tree, as this is both redundant and time-consuming. Instead, the following libraries were utilised [30]:

```
# Load libraries
import pandas as pd
from sklearn.tree import DecisionTreeRegressor  # Import Decision Tree Regressor for
    regression tasks
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
```

LISTING 2.1: Setting Up the Environment: Importing Libraries for Decision Tree Modeling [30]

The code in Listing 2.1 was necessary to build a decision tree classifier model. The libraries imported have different purposes. For data manipulation and analysis, Pandas was used. The most important library was DecisionTreeClassifier from the sklearn.tree module. This class was used to build the decision tree classifier model. Another class also exists, which is called the DecisionTreeRegressor [34]. In the case study, the DecisionTreeRegressor was used instead of the DecisionTreeClassifier because the target variable represents numeric values

rather than discrete categories. For testing and splitting the data, the train test split function was imported. This way, the function did not have to be manually designed. Lastly, from sklearn, metrics was imported, with which the performance of the machine learning model, in this case decision tree model, was evaluated [30]. However, as mentioned prior, in the case study, the target variable is not a discrete category but rather a numeric value. Therefore, to assess the performance of the decision tree, from sklearn.metrics, the mean squared error (MSE) [35] and R^2 score functions [36] were used. In regression tasks, where the objective is to predict a continuous value (e.g., DEC of bacteria), decision trees typically rely on the MSE as a criterion to assess the effectiveness of a split. The MSE quantifies the average squared difference between the actual values (y_i) and the predicted values ($\hat{y}_i$) within a subset. The lower the MSE is, the better the model predicts the outcome. The MSE is defined as [37]:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.2}$$

Where:

- n: Number of observations (data points) in the subset.
- y_i: Actual value of the target outcome (e.g., the observed removal efficiency for the i-th data point).
- $\hat{y}_i$: Predicted value of the target outcome (e.g., the average removal efficiency of all points in the subset).

The reason as to why the MSE was chosen instead of the Root Mean Squared Error (RMSE), is due to the fact that the MSE squares the errors, meaning that larger deviations from the predicted values are penalized more significantly than smaller ones.

The R^2 Score, also known as the coefficient of determination [36], measures how well the regression model explains the variance of the target variable. It is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.3}$$

Where:

- y_i: Actual value of the i-th observation.
- $\hat{y}_i$: Predicted value for the i-th observation.
- $\bar{y}$: Mean of the actual values.
- n: Total number of observations.

The R^2 score ranges from [36]:

- **1**: Perfect fit, where the model explains all the variability in the target variable.
- **0**: The model explains no variability, equivalent to predicting the mean value for all observations.
- **Negative values**: Indicate that the model performs worse than simply predicting the mean of the target variable.

For this study, different feature variables were given in an experiment done with a SSF, such as the FR, D_{10} and the T.B.H. Also, the Decimal Elimination Capacity (DEC, in logs) for bacteria (E.Coli, Total Coliform) and Viruses (Enterovirus, Adenovirus, Bacteriophage MS2) was given. The goal was to use the decision tree to predict the DEC (in this case the target variable), using the feature variables.

Also, it is important to state that decision tree models are unable to handle missing data (NaN values), as mentioned prior. In this research, the dataset presented in A contained a considerable amount of missing data in the feature variables, such as FR and D_{10}, or on the DEC of bacteria or viruses. Therefore, this data needed to be removed from the dataset in order to create a decision tree. That is also why XGBoost modelling was chosen in the research, as mentioned prior, as this is a model that incorporates both decision trees and that accounts for NaN values.

After the decision tree model had been constructed, the dataset was divided into a training set and a test set. For the case study, the choice was to have 80 % training data and 20 % test data, since the data set had only 47 rows after removing the rows with NaN values. In this case, it was important to make sure that the test data was large enough so that it could provide a good evaluation, while there was still a considerable amount of data in the training set. The code for the implementation of the decision tree model for the case study, using the feature variables FR, D_{10} and T.B.H., and the target variable DEC_{BACT}, is given below.

```
# Remove rows with NaN values in feature columns or target variable
Dataset = Dataset.dropna(subset=['FR [m/h]', 'D10 [mm]', 'T.B.H. [m]', 'DEC_bact_1'])

# Split dataset into features and target variable
feature_cols = ['FR [m/h]', 'D10 [mm]', 'T.B.H. [m]']

X = Dataset[feature_cols]  # Features

y = Dataset['DEC_bact_1']  # Target variable

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state
    =1)  # 80% training and 20% test


# Create Decision Tree Classifier object
clf = DecisionTreeRegressor()

# Train Decision Tree Classifier
clf = clf.fit(X_train, y_train)

# Predict the response for the test dataset
y_pred = clf.predict(X_test)
```

LISTING 2.2: Data Preprocessing and Decision Tree Regression Model Implementation [30]

Afterwards, the MSE and R^2 of the model could be calculated. The code for this is given below.

```
print("Mean Squared Error (MSE):", mean_squared_error(y_test, y_pred))
print("R   Score:", r2_score(y_test, y_pred))
```

LISTING 2.3: Calculating the MSE and R^2 of a random decision tree using data from the case study [35][36]

The results of a random decision tree generated using the feature variables (FR, D_{10}, and T.B.H.) and the target variable (DEC_{BACT}) from the case study revealed a MSE of 0.356 and an R^2 score of -0.141. As mentioned prior, the MSE represents the average squared difference between the actual and predicted values, with lower values indicating better accuracy. A MSE of 0.356 suggests that, while the model's predictions are not excessively far off, there is considerable room for improvement. For the R^2 score; a negative R^2 score, as observed here (-0.141), indicated that the model performs worse than a simple baseline prediction using the mean of the target variable [36]. These results highlighted the limitations of using a single random decision tree for predictive modeling [29][38]. To achieve more accurate predictions and a better understanding of the relationships between features and the target variable, more advanced ensemble methods were needed. The following section will explore XGBoost modeling, which combines multiple trees in an optimized manner to address these limitations and improve prediction performance [39].

The constructed random decision tree model for the case study could also be visualized and had been done so to give an idea of how the decision tree works. On the next page, a random decision tree containing the feature variables and target variable can be seen. The aforementioned root nodes, branches and leaf nodes can also be observed. The root nodes also state a value. This value is proportional to the value of the predicted DEC_{BACT}.

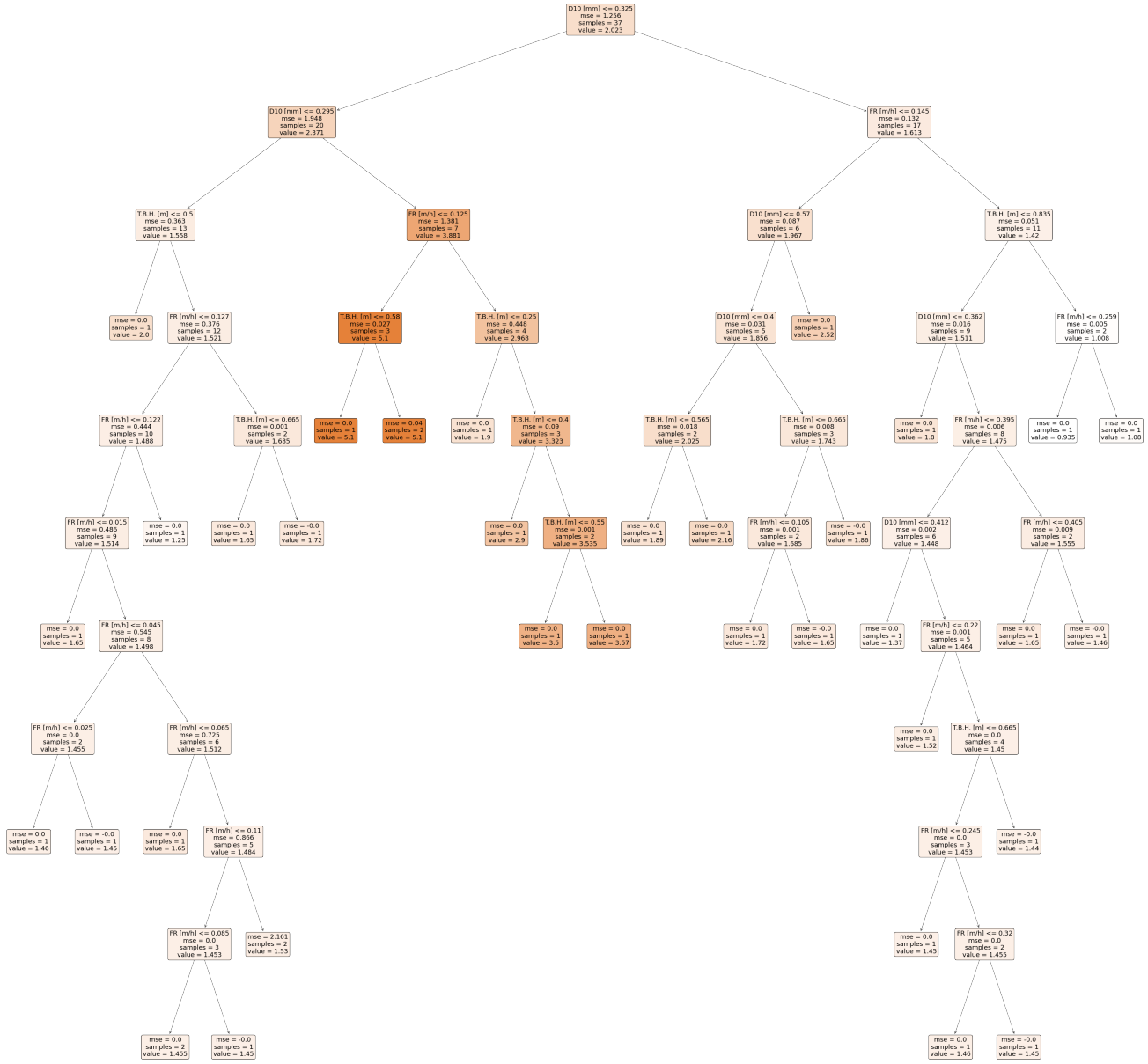

FIGURE 2.6: Example of a Random Decision Tree, created from the code given in Listing 2.2

2.5 Understanding the XGBoost Model: Concepts and Components

XGBoost is an advanced machine learning algorithm based on decision tree ensembles [39][40][41]. In essence, a decision tree ensemble combines multiple decision trees to improve prediction accuracy. Each tree contributes to the final prediction, either by averaging outputs (in regression tasks) or voting (in classification tasks). This ensemble approach reduces the weaknesses of individual trees, such as overfitting and limited predictive power, as showcased in the prior section, resulting in a more robust and reliable model. What sets XGBoost apart is its use of gradient boosting, an optimization technique where trees are built sequentially to correct errors made by previous ones. Additionally, XGBoost incorporates regularization techniques, enhances computational efficiency, and is highly scalable, making it well-suited for structured data tasks like regression, classification, and ranking problems. To better understand the concepts of the XGBoost model, first, the students' grade example from the previous section, will be continued here.

2.5.1 Basic Elements of Supervised Learning in XGBoost

As mentioned prior, XGBoost models make use of supervised learning, where training data x_i (with multiple feature variables, such as D_{10}, FR and T.B.H.) is used to predict a target variable y_i, such as DEC_{BACT}. This concept is not only applicable for XGBoost models however, as simpler models can also make use of simple learning. As an example, the linear model below is given to predict a target variable (y_i) based on the weighted sum of the feature variables (x_i),

$$\hat{y}_i = \sum_j \theta_j x_{ij} \tag{2.4}$$

For every x_{ij}, a weight θ_j is assigned in the case of the linear model above. Through learning of the data, the parameters θ are determined. Thus, these parameters are unknown from the start. Returning to the decision tree model that has been used for explaining the XGBoost methodology (e.g., predicting students' grades based on study hours and practice tests), the parameters θ can be thought of as, for example, the splitting rules (e.g., "Study Hours > 3") and the selected features (e.g., "Practice Tests").

In order to find the best parameters θ (e.g., splitting rules, selected features in the decision tree, etc.) that fit the feature variables to the target variable y_i (e.g., student' grades), an objective function is defined, which measures how well the model fits the training data. The objective function for the XGBoost model is given below.

$$\text{obj}(\theta) = L(\theta) + \Omega(\theta) \tag{2.5}$$

In this equation:

- $L(\theta)$: Represents the training loss, which quantifies how well the model's predictions ($\hat{y}_i$) align with the actual target values (y_i) in the dataset.
- $\Omega(\theta)$: Represents the regularization term, which helps control the model's complexity, preventing it from overfitting the training data.

A commonly used measure for the training loss function, in regression problems, is the MSE, which was introduced in the previous section. In the case of XGBoost, it is defined as followed:

$$L(\theta) = \sum_i (y_i - \hat{y}_i)^2 \tag{2.6}$$

The regularization term $\Omega(\theta)$ penalizes unnecessary model complexity, promoting better generalization to unseen data. Without regularization, a model may overfit the training data by learning noise or unnecessary details, resulting in poor generalization to new, unseen data. This concept is better explained through visualization as can be seen in figure 2.7.
Figure 2.7 illustrates the impact of regularization on model complexity and fit. In the top graph, excessive splits (t_1, t_2, t_3, t_4, t_5) result in a highly complex model, capturing every small fluctuation in the data. This leads to overfitting, where the model performs well on training data but fails to generalize to new data. In contrast, the bottom graph shows a balanced model that uses a single meaningful split (t_1) to capture the overall trend. This balance minimizes overfitting by controlling complexity ($\Omega(f)$) while keeping the training loss ($L(f)$) low, ensuring better generalization.

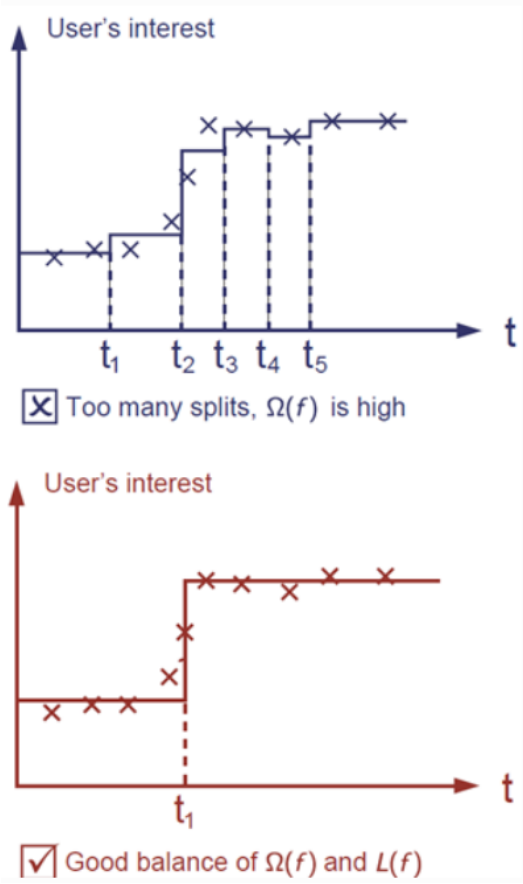

FIGURE 2.7: Effect of Regularization on Model Complexity and Fit [42]

2.5.2 Model Choice of XGBoosts: Decision Tree Ensembles

The XGBoost model is build upon decision tree ensembles [39][40][41][42]. This model contains a set of classification and regression trees (CART). An example of a regression decision tree was the earlier mentioned decision tree on students' grades or the one used for the case study, of which the code was given in Listings 2.2. However, the classification and regression decision trees used in the ensemble decision tree model of XGBoost differ from the traditional decision tree. In a traditional regression decision tree, such as the one used for predicting students' grades, each leaf node contains a final prediction, as can be seen in figure 2.6, which is typically the average of the target variable (e.g., DEC_{BACT}) for the data points in that subset. However, in XGBoost, the regression decision trees operate differently as part of the ensemble. Instead of each leaf node containing a direct prediction of the target variable (e.g., the average student grade or DEC_{BACT} in a subset), the leaf nodes in XGBoost trees store contribution scores. These scores represent adjustments to the overall prediction rather than final predictions themselves.

In contrast, in XGBoost:

1. The first tree might predict the baseline, such as the global average grade, $\hat{y}_1 = 3.5$.
2. Subsequent trees would iteratively refine this baseline by learning residuals (the difference between actual grades and predicted grades). For instance:
    - A second tree could add $+0.3$ for students who studied more than 3 hours.
    - A third tree might add $+0.1$ for students who took more than 1 practice test.

This iterative refinement ensures that each tree in the ensemble focuses on the errors left by previous trees, gradually improving the model's overall predictive accuracy. The final prediction is the sum of the contributions from all trees, regularized to prevent overfitting. This process is visualized in figure 2.8 below, in which ensemble decision trees, using the CART method, is utilised to predict whether a certain target group enjoys a computer videogame or not.

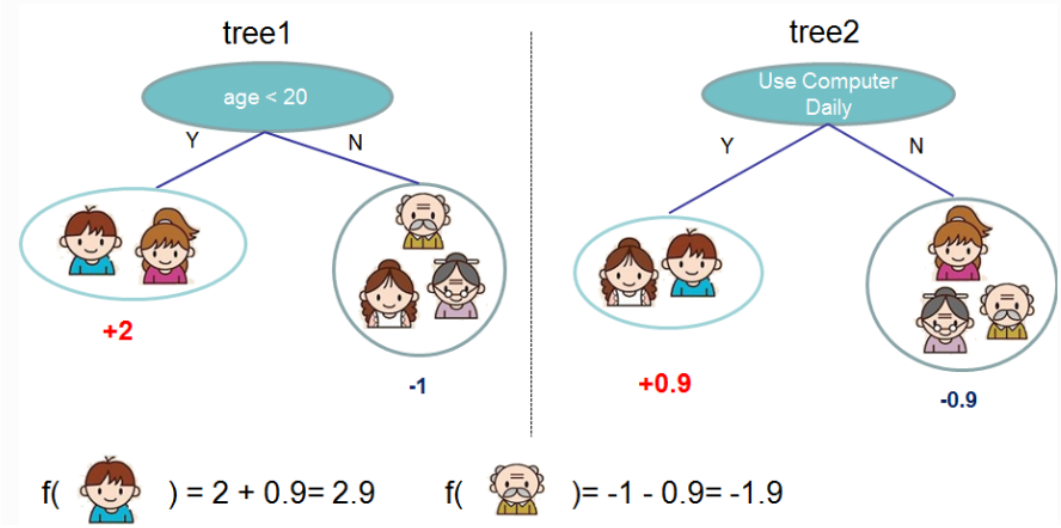

FIGURE 2.8: Ensemble Learning in XGBoost: Combining Contributions from Multiple Trees [42]

The model on decision tree ensembles is based on the objective function and training loss function, represented in equations 2.5 and 2.6, and is given by the following equation [40]:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F} \tag{2.7}$$

Here:

- $\hat{y}_i$: The final prediction for data point i (e.g., a student's grade).
- K: The total number of trees in the ensemble.
- $f_k(x_i)$: The prediction from the k-th tree for input features x_i (e.g., study hours, number of practice tests).
- $\mathcal{F}$: The space of all possible CARTs (Classification and Regression Trees).

The optimization objective combines two components:

1. **Training Loss**: Measures how well the predictions match the actual target values (y_i), such as the earlier stated MSE for grades.
2. **Regularization Term**: Controls the complexity of the trees to prevent overfitting. This is represented as $\omega(f_k)$, which penalizes trees with excessive splits or depth.

The objective function is:

$$\text{obj}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \omega(f_k) \tag{2.8}$$

Where:

- $l(y_i, \hat{y}_i)$: The loss function (e.g., MSE, measuring the difference between actual and predicted grades).
- $\omega(f_k)$: The complexity of the k-th tree.

In the case of predicting students' grades, the model would operate as followed:

1. **Feature Variables (x_i)**: Each student has feature variables such as "Study Hours" and "Number of Practice Tests." For example, $x_i = (\text{Study Hours: } 4, \text{Practice Tests: } 2)$.
2. **Tree Predictions ($f_k(x_i)$)**:

- **Tree 1**: The tree predicts a baseline based on "Study Hours."
  - If "Study Hours > 3," it contributes $f_1(x_i) = +1.5$.
- **Tree 2**: Refines this prediction based on "Practice Tests."
  - If "Practice Tests > 1," it adds $f_2(x_i) = +0.8$.

3. **Final Prediction ($\hat{y}_i$)**: The model sums the contributions from all trees:

$$\hat{y}_i = f_1(x_i) + f_2(x_i) = 1.5 + 0.8 = 2.3$$

The student's predicted grade is 2.3.

4. **Training Loss ($l(y_i, \hat{y}_i)$)**: The loss function calculates the difference between the actual grade (y_i) and the predicted grade ($\hat{y}_i$). For example, if the actual grade is 3.0, the MSE loss is:

$$l(y_i, \hat{y}_i) = (3.0 - 2.3)^2 = 0.49$$

5. **Regularization ($\omega(f_k)$)**: To avoid overfitting, the model penalizes trees with too many splits or excessive depth. For example, a tree that splits unnecessarily based on irrelevant feature variables (e.g., "Favorite Subject") will have a higher $\omega(f_k)$, discouraging such complexity.

2.5.3 Forming the Algorithm of the XGboost Model

As stated previously, the XGBoost model is a supervised learning model and in order to train this model, the objective function should is defined and then optimized. The objective function had already been stated in equation 2.8. Furthermore, the algorithm for the additive strategy used in XGboost modelling, which was stated prior, in equation 2.7, is given below in more detail:

$$\hat{y}_i^{(0)} = 0 \tag{2.9}$$

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \tag{2.10}$$

$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \tag{2.11}$$

$$\vdots \tag{2.12}$$

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{2.13}$$

Following the given equation 2.13, the equation for the aforementioned loss function, can be rewritten to equation 2.14:

$$\text{obj}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \omega(f_i) \tag{2.14}$$

Expanding it, the function can be written as followed below. The reason this is possible, is due to the fact that the results of the decision tree are adjusted, as shown in equations 2.9 to 2.13, and therefore the new objective is defined. For clarification purposes, in the case of XGBoost, the objective is the function that the XGBoost model seeks to minimize during training of the dataset.

$$\text{obj}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \omega(f_t) + \text{constant} \tag{2.15}$$

Equation 2.15 can be adjusted even further. In the previous section, it was stated that for training loss functions in regression problems, the MSE is used. The formula for the MSE was given in equation 2.2. By substituting equation 2.2 into equation 2.15, the following equations can be derived:

$$\text{obj}^{(t)} = \sum_{i=1}^{n} \left(y_i - \left(\hat{y}_i^{(t-1)} + f_t(x_i) \right) \right)^2 + \sum_{i=1}^{t} \omega(f_i) \tag{2.16}$$

This can be further simplified as:

$$\text{obj}^{(t)} = \sum_{i=1}^{n} \left[2 \left(\hat{y}_i^{(t-1)} - y_i \right) f_t(x_i) + f_t(x_i)^2 \right] + \omega(f_t) + \text{constant} \tag{2.17}$$

In XGBoost modelling, using a Taylor expansion up to the second order is preferred instead of the form of MSE [42]. The reason for this is that loss functions, besides MSE (quadratic), exist, such as logistic loss functions. For these type of functions, the Taylor expansion can approximate them into a quadratic form using gradients and curvature [43]. The conversion shown below has been implemented based on the insights provided in Sukanya Bag's article [44].:

$$L(x) \approx L(a) + L'(a)(x-a) + \frac{1}{2} L''(a)(x-a)^2 + \dots \tag{2.18}$$

$$L(y, \hat{y}) \approx L(y, \hat{y}^{(t-1)}) + \frac{\partial L}{\partial \hat{y}} (\hat{y} - \hat{y}^{(t-1)}) + \frac{1}{2} \frac{\partial^2 L}{\partial \hat{y}^2} (\hat{y} - \hat{y}^{(t-1)})^2 \tag{2.19}$$

1. First derivative (gradient):

$$\frac{\partial L}{\partial \hat{y}} = g_i \tag{2.20}$$

2. Second derivative (curvature):

$$\frac{\partial^2 L}{\partial \hat{y}^2} = h_i \tag{2.21}$$

Using these, the Taylor expansion becomes:

$$L(y, \hat{y}) \approx (y - \hat{y}^{(t-1)})^2 + (g_i)(\hat{y} - \hat{y}^{(t-1)}) + \frac{1}{2}(h_i)(\hat{y} - \hat{y}^{(t-1)})^2 \tag{2.22}$$

Reorganizing, this becomes:

$$L(y, \hat{y}) \approx \text{constant} + \underbrace{g_i f_t(x_i)}_{\text{First-order term}} + \underbrace{\frac{1}{2} h_i f_t^2(x_i)}_{\text{Second-order term}} \tag{2.23}$$

Now the objective loss function, at step t, becomes:

$$\text{obj}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \omega(f_t) \tag{2.24}$$

The regularization term, or the complexity of the tree, also needs to be defined. In XGBoost, this regularization term can be divided into an L1 and L2 regularization term [40][45], of which the equations for L1 and L2 are given below:

$$\text{obj}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \omega_{\text{L1}} + \omega_{\text{L2}} \tag{2.25}$$

$$\omega_{\text{L1}} = \alpha \sum_{j=1}^{T} |w_j| \tag{2.26}$$

$$\omega_{\text{L2}} = \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{2.27}$$

Here, λ and α are hyperparameters that control the regularization terms ω_{L1} and ω_{L2} [40][45]. Through L2 regularization, the weights are encouraged to be small, whereas in L1 regularization, the term encourages the weights to shrink towards zero. Now, the objective function for the tree can be simplified. This simplified objective function is presented by the tutorial of XGBoost [42]:

$$\text{obj}^{(t)} \approx \sum_{i=1}^{n} \left[g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2 \right] + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 + \alpha \sum_{j=1}^{T} |w_j| \tag{2.28}$$

Since all data points in the same leaf share the same weight w_j, the objective function can be reformulated, by grouping data points into leaves:

$$\text{obj}^{(t)} = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 + \alpha |w_j| \right] \tag{2.29}$$

where:

- $I_j = \{i | q(x_i) = j\}$ represents all data points in leaf j.
- The summation $\sum_{i \in I_j}$ ensures that all samples in the same leaf contribute together.

To simplify further, the summations of the gradient and curvature are shortened as followed:

$$G_j = \sum_{i \in I_j} g_i \quad \text{(Total gradient of the leaf)} \tag{2.30}$$

$$H_j = \sum_{i \in I_j} h_i \quad \text{(Total curvature of the leaf)} \tag{2.31}$$

Using these terms, the loss function becomes:

$$\text{obj}^{(t)} = \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2 + \alpha |w_j| \right] \tag{2.32}$$

As stated prior, the loss function needs to be minimized and in order to do so, the derivative of the function needs to be equal to zero. If the terms in the brackets equal zero, then the derivative will be equal to zero, therefore the following equation needs to be solved:

$$\frac{d}{dw_j} \left(G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2 + \alpha |w_j| \right) = 0 \tag{2.33}$$

Solving for w_j, the following is determined:

$$w_j^* = -\frac{G_j}{H_j + \lambda} \tag{2.34}$$

Now, w_j^* can be substituted back into the objective function to compute the optimal objective value reduction:

$$\text{obj}^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} \tag{2.35}$$

This equation explains how much the error in the model reduces when adding a new tree.

Now that the performance of a tree is defined through the use of equation 2.35, the goal now is to determine how the tree is build and whether splitting a node (leaf) in a decision tree improves the model or not. Instead of growing the tree indefinitely, it is evaluated whether splitting is beneficial. The equation given below [42] provides a numerical way to determine whether splitting a node improves the model's predictive power. This equation is called the Gain. In the previous section, it was also explained that decision trees make use of either the Information Gain, Gain Ratio of Gini Index. In XGBoost modelling, the following Gain formula is used for this:

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma \tag{2.36}$$

Each term represents a specific part of how a tree split is evaluated:

- **Left Leaf Score:** $\frac{G_L^2}{H_L+\lambda}$
- **Right Leaf Score:** $\frac{G_R^2}{H_R+\lambda}$
  - Similar to the left leaf, this term measures the improvement on the right leaf.
- **Original Leaf Score:** $\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}$
  - Represents the score before the split, using both left and right samples combined.
- **Regularization Term:** $-\gamma$
  - Ensures that a split only happens if it significantly improves the model. If the gain is smaller than γ, the branch is not added (pruning).

2.5.4 Application of XGBoost Modelling to the Case Study

Now that the concepts of XGBoost modelling have been explained, the XGBoost model was implemented for the case study of SSFs, and more specifically for the dataset given in A using Jupyter Notebook. Just as was the case for decision tree modelling, Jupyter Notebook provides built-in libraries and functions to create and train decision XGBoost models [46]. This eliminates the need to manually write the algorithm for constructing the XGBoost model, as this is both redundant and time-consuming. Instead, the following libraries, given by the XGBoost developers, were utilised [46] [47]:

```
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split, GridSearchCV, StratifiedKFold
from sklearn.metrics import mean_squared_error, r2_score
# Initialize the XGBoost Regressor
xgb = XGBRegressor(random_state=42)
```

LISTING 2.4: Initializing and Importing Libraries for XGBoost Regression Model [47]

The mean_squared_error and r2_score from the sklearn.metrics library and the train_test_split package from the sklearn.model_selection libraries had been explained prior. Besides these packages, the GridSearchCV and the StratifiedKFold packages from the sklearn.model_selection were also implemented [48]. GridSearchCV was utilised to find the best combination of hyperparameters to optimize the XGBoost model performance. The StratifiedKFold package ensured that each fold in cross-validation has the same proportion of target values [49].

Next the feature variables and target variables were defined yet again, as was the case for the earlier decision tree model, and afterwards the data was split into test and training data:

```
# Drop rows with missing target values
data_bact = data_bact.dropna(subset=['DEC_bact_1'])
X = data_bact[['FR_m/h', 'D10_mm', 'HRT_h', 'D60/D10'  ]]
y = data_bact['DEC_bact_1']
# Create quantile-based bins for the target variable
y_binned = pd.qcut(y, q=10, labels=False)  # Divide into 10 bins (quantiles)
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state
    =43, shuffle=True)
```

LISTING 2.5: Defining the feature variables (e.g., T.B.H. and FR) and target variable (e.g., DEC of bacteria, and splitting the data into test and training data

The variable y_binned was defined, as using y would result in an error in Jupyter Notebook. This was due to the fact that y, which is in the case study, the DEC of bacteria, a continuous variable. StratifiedKFold requires a classification style target [49], meaning it expects discrete classes rather than continuous numerical values, hence why the target variable was converted into bins.

The hyperparameters of the XGBoost model were then defined. These parameters were systematically adjusted and fine-tuned throughout the training and testing phases, by observing the MSE of the test set and the cross-validation set, to achieve the best possible model accuracy. Before these parameters were defined, the XGBoost model performed poorly, resulting in lower MSE scores for both test and CV MSEs. After consulting with Post - Doc Researcher Grigorios Kyritsakas, certain hyperparameters were chosen to be implemented in the model [47][50]. The parameter grid and the chosen parameters and their values are shown below:

```
# Set up a parameter grid for hyperparameter tuning
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [6, 8, 10],
    'learning_rate': [0.01, 0.05, 0.1],
    'subsample': [0.6, 0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0],
    'reg_alpha (L1 Regularization): [0.1, 0.5, 1],
    'reg_lambda (L2 Regularization)': [5, 10]
}
```

LISTING 2.6: Defining a Parameter Grid for Hyperparameter Tuning [50]

- `n_estimators`: The number of trees used in the model. Choosing more trees to use may lead to a better performance, however, the computation time also increased considerably. Therefore, a maximum of 300 was chosen.
- `max_depth`: The maximum depth of each tree. Deeper trees can model more complex data but are more prone to overfitting. The max depth was chosen based on the resulting MSE for the cross validation.
- `learning_rate`: Shrinks the contribution of each tree to prevent overfitting. This function serves the same purpose as the previously stated L2 Regularization.
- `subsample`: The fraction of training data used per boosting round. It helps prevent overfitting.
- `colsample_bytree`: The fraction of feature variables randomly sampled for each tree. Reducing this can also help prevent overfitting.

The following code initializes `StratifiedKFold`, which was used to split the data into multiple folds while preserving the distribution of the target variable:

```
skf = StratifiedKFold(n_splits=4, shuffle=True, random_state=42)
```

LISTING 2.7: Initializing Stratified K-Fold

In this code, n_splits ensured that the dataset was divided into four equal folds with the same distribution of the binned target variable y_binned, for cross validation. Afterwards, GridSearchCV [48] was used to optimize the hyperparameters of the XGBoost model by testing multiple combinations of the parameters that were stated in Listing 2.6. The code for this is depicted below:

```
grid_search = GridSearchCV(
    estimator=xgb,
    param_grid=param_grid,
    cv=skf.split(X, y_binned),  # Use the binned target variable for stratification
    scoring='r2',
    verbose=1,
    n_jobs=-1
)
```

LISTING 2.8: Performing Grid Search with StratifiedKFold [48]

- `estimator=xgb` specified the use of the XGBoost regressor model instead of the XGBoost classification model.
- `param_grid=param_grid` defined the hyperparameter search space.
- `cv=skf.split(X, y_binned)` ensured that cross-validation was stratified using the binned target variable.
- `scoring='r2'` evaluated model performance using the R-squared metric.
- `verbose=1` printed updates during training.
- `n_jobs=-1` enabled parallel computation to speed up the search.

After the best parameters were found, the final model was trained, using the function grid_search.fit(X, y) in Jupyter Notebook [48], and then used for predictions and performance evaluation.

```
grid_search.fit(X, y)
print(f"Best Parameters: {grid_search.best_params_}")
print(f"Best R^2 Score from CV: {grid_search.best_score_}")
best_xgb = grid_search.best_estimator_
y_pred = best_xgb.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error on Test Set: {mse}")
print(f"R^2 Score on Test Set: {r2}")
```

LISTING 2.9: Fitting Grid Search to Training Data

- `best_xgb` retrieved the best model from GridSearchCV.
- `y_pred = best_xgb.predict(X_test)` generated predictions on the test set.

2.5.5 Using the Model for Parameter Adjustment

After testing the model, the XGBoost model was used to explore the effects of adjusting design parameters on removal efficiencies. By inputting adjusted values for key design parameters, such as `FR_m/h` (FR), `T.B.H._m` (T.B.H.), and `D10_mm` (D_{10}), the model estimated the corresponding removal efficiencies, as can be seen in figure 2.9. This approach allowed for a systematic exploration of how changes in design parameters impacted bacterial and virus removal efficiency. The ability of the XGBoost model to provide reliable predictions based on parameter adjustments demonstrated its practical utility for optimizing the design of slow sand filters. The code for this interactive tool is given in Appendix E.

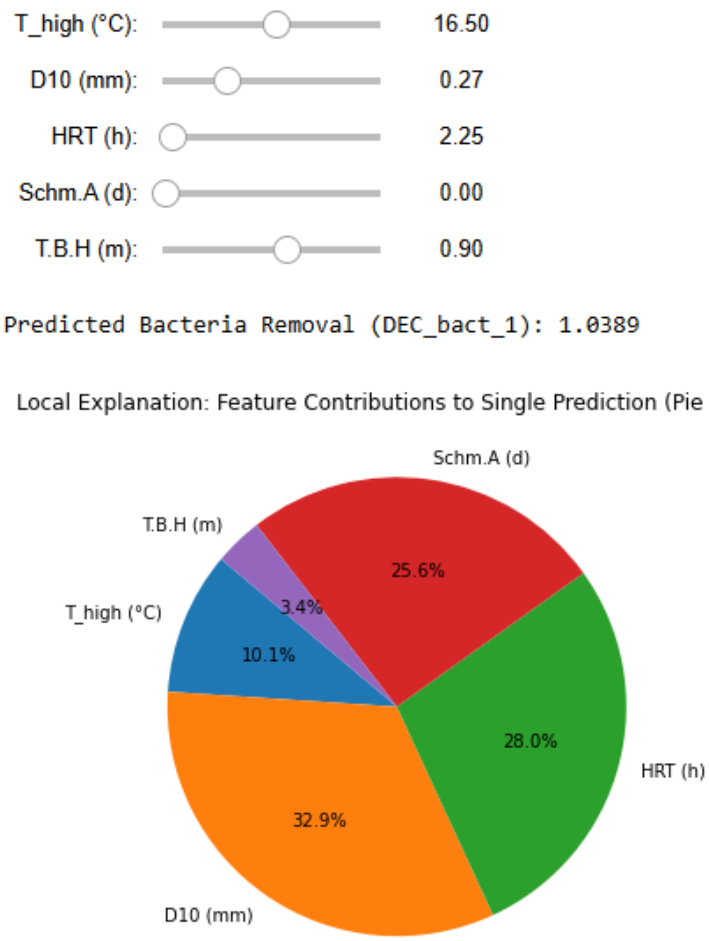

FIGURE 2.9: Adjusting Parameters in XGboost, leading to prediction of Bacteria Removal, using 5 features. The code corresponding to the model is given in E

Chapter 3

Integrating Literature and Data: An Analytical Approach

The purpose of the literature review is to explore the key design parameters of SSFs and to understand how specific factors, like FR and D_{10}, influence the ability of SSFs to remove bacteria and viruses, as stated in different papers. To achieve this, a broad range of studies will be analyzed, with the aim of compiling and quantifying data on the relationships between these design parameters and the removal efficiencies of contaminants. Quantifying both the parameters and their outcomes is particularly important because it allows identification of a spectrum of values that could optimize contaminant removal under various conditions. Furthermore, key findings regarding the effects of the design parameters on the removal efficiencies of bacteria and viruses will be noted from the studies, as these will later be compared and validated using the dataset. In addition, the literature review will take into account the differences in experimental setups across studies, recognizing that these variations may influence the reported removal efficiencies. By treating the review as a kind of case study, the intention is to better understand how diverse experimental conditions might affect the results. Lastly, the findings from the literature review will lay the groundwork for the later analysis of lab- and pilot-scale data.

3.1 The Filtration Rate

3.1.1 Definition of The Filtration Rate

The performance of a SSF treatment is highly affected by the FR that is applied in a SSF [51]. This design parameter is an independent parameter and is the flow rate of water entering the filtration system relative to the surface area of the filter. It indicates how quickly water is being processed through the filter. Due to these associations, the FR is often fixed after the SSF has been constructed. However, different configurations to alter the FRs can still be realized. In laboratory or pilot scale, applying different FRs is made easier, through control of conditions. For SSFs, a range of 0.1 to 0.4 m/h is suggested for FRs by Huisman and wood (1974) [52].

The FR influences the contact time between the water and the filter media. Lower FRs mean longer contact times, which generally improve the removal efficiencies of contaminants. Higher FRs can lead to rapid clogging of the filter media, requiring more frequent maintenance and cleaning.

3.1.2 Effects of Filtration Rates on Removal Efficiencies of Bacteria and Viruses for Drinking Water Preparation in Literature

In experiments focused on the removal of E. coli, fecal coliform, and Campylobacter in drinking water treatment, the FRs used have varied significantly, yet all resulted in effective removal efficiencies for bacteria. It is stated that changes to the FR only will not have a significant effect on the removal of bacteria. In a study conducted by Wim et al. [53] , it was demonstrated that the development of the *Schmutzdecke* had a significantly greater impact on the removal efficiency of *E. coli* than the choice of FR. The configurations of each of the SSFs in the study of Wim et al. has been showcased in table 3.1. In this table, it is made clear that different configurations of SSFs can lead to different outcomes of removal efficiencies for bacteria.

While lower FRs generally allowed for extended contact time and thus higher removal efficiency, the presence of a mature *Schmutzdecke* led to superior removal capabilities regardless of the FR employed. The study highlighted that, even at higher FRs (e.g., 0.4 m/h), effective removal could be achieved if the *Schmutzdecke* was well-developed. Conversely, in experiments without a *Schmutzdecke*, even lower FRs (e.g., 0.08 m/h) yielded suboptimal removal efficiencies. However, studies have also shown that even remarkably low FRs, ranging from 2

to 20 L/day, can effectively remove total coliform by over 95 % by day 7 in bio-sand filters (BSFs), 97.4 % in household SSFs, and even up to more than 99 % [54][55][56], depending on the development of the *Schmutzdecke*. These findings underline the critical role of biofilm development in optimizing SSF performance.

Design Parameter	**0.25 m/h**	**0.3 m/h**	**0.4 m/h**
Filter Medium/Media	sand	sand	sand
Grain size(s)	0.13–0.37	0.30	0.15–0.6
Total Bed Height (cm)	150	150	150
Surface Area (m^2)	-	2.56	-
Type of Water	Pre-treated surface water	Pre-treated surface water	Pre-treated surface water
Duration of Experiment	Continuous operation	10 days	Continuous operation

TABLE 3.1: Configurations of the SSFs utilized in the study of Wim et al. [53]

For virus removal, the maturity of the filter bed is more critical than the FR as well [57], as both virus and bacteria removal primarily occur through microbial activity. Similar to the removal of bacteria, a wide range of FRs have been utilized in studies for the removal of pathogenic viruses. Both high (0.15 - 0.4 m/h) and low (0.29 m/d) FRs have proven to remove enteric viruses, such as the Polio viruses, by > 99.93 % and > 1.8 $\log_{10}$, respectively [55][58][59][60].
As mentioned earlier, both high and low FRs have demonstrated efficient removal; however, higher FRs can lead to faster clogging. One way to combat this, is through thorough pre-treatment of the influent water [53][61]. In a study by Pereira et al. pre-treatment of raw influent water was significantly enhanced which mitigated clogging issues and extended operational duration [61]. Without pre-treatment, even lower FRs (e.g., 0.075 m/h) still led to rapid clogging within 15 days, whereas pre-treated influent allowed SSFs to operate efficiently at higher FRs (e.g., 0.3 m/h) for up to 71 days. The findings align with other studies, such as Wim et al., by showing that a mature *Schmutzdecke* can develop effectively even under higher FRs, provided the influent is adequately pre-treated.

Based on findings from previously mentioned studies, it is evident that establishing the optimal FR for a SSF is a difficult task as the removal efficiencies for bacteria and viruses do not solely depend on the FR but also on other factors, such as the source water quality, temperature, biological maturity of the *Schmutzdecke*, pre-treatment of the influent water and residence time, of which some have a direct correlation with the FR. Furthermore, from literature review, it becomes apparent that lab-based filters have shown higher removal efficiencies for bacteria and viruses in comparison to SSF operations in the field. This could be attributed to improved control of operational conditions, particularly the hydraulic residence time [1]. Due to the differences in the design parameters/operational conditions, both high and low ranges of FRs have proven to be adequate for the removal of the target contaminants. When designing a SSF, the FR should ideally be selected based on the specific system configuration and the quality of the influent water it is intended to treat. This principle is also supported by drinking water treatment companies, as demonstrated in a technical fact sheet from one such company, which highlights that the filtration rate depends on various design parameters and operational conditions, including bed area, filter medium gradation, and influent water quality [62]. This principle is further substantiated by Abdiyev et al. which have stated that if the raw water has a particle concentration of less than 25 mg/L, the range of filtration rate that should be used is between 0.08-0.4 m/h, and if it is above 25 mg/L, it should vary between 0.1-0.2 m/h [63]. This might also explain the different values set for the FR by different researchers. As mentioned prior, Huisman and Wood have stated that a FR between the ranges of 0.1 to 0.4 m/h is effective [52], while Visscher has stated that FRs between 0.1 and 0.2 m/h would result in an effluent with good quality and higher FRs would spoil the quality of the effluent water. In contrast to this again, the studies of Muhammed et al. and Sadiq et al. have both proven that higher FRs may show high removal efficiencies for higher filtration rates [64] [65]. In all these studies, the quality of the influent water differed.

From literature study, the following **remarks** can be made regarding the FR:

- Both high and low ranges of FRs have proven effective for the removal of bacteria and viruses, but their efficiency depends heavily on additional factors, such as influent water quality, pre-treatment, sand size, residence time, and temperature. The biological maturity of the *Schmutzdecke* also plays a significantly larger role, as highlighted in multiple studies [53].
- Optimal FR selection should be based on the influent water quality and SSF configuration. For raw water with low pollutant concentrations, higher FRs may be viable, while higher pollutant concentrations necessitate lower FRs to prevent rapid clogging [63].

3.1.3 Exploratory Data Analysis of Effect of Filtration Rate in the Removal of Bacteria and Viruses

To better understand the effects of the FR on the removal of bacteria and viruses, scatterplots were created to illustrate the relationship between FR and removal efficiencies for both bacteria and viruses (figures 3.1 and 3.2). These scatterplots, presented separately for bacteria and viruses, aimed to visualize any potential relationships or trends between the filtration rate and contaminant removal efficiencies. However, the scatterplots reveal that there is no apparent or consistent correlation between FR and the removal efficiencies for either bacteria or viruses, as was already deduced through use of the correlation matrix. This absence of a clear trend can be attributed to the complex, multi-factorial nature of the removal processes in SSFs discussed earlier. Different configurations of the SSF result in different outcomes of effective FRs.

For example, parameters such as:

- D_{10} and n [-], as these directly affect the retention and filtration mechanisms, influencing the removal of contaminants.
- T.B.H., as an increased bed height will lead to higher removal of contaminants due to extended filtration pathways.
- Operational conditions, such as temperature, have a significant effect on the speed of biological processes in SSFs

The other design parameters and operational conditions interact with the FR and contribute to the observed variability in removal efficiencies. Thus, the scatterplots reflect a composite of these interactions rather than isolating the effect of FR. The lack of a discernible pattern suggests that while FR is a critical design parameter, its impact cannot be evaluated in isolation. Instead, the scatterplot must be analyzed alongside other key variables. If an interesting pattern, an outlier, or a result that doesn't align with the literature is observed, the specific study responsible for that point will be examined. In that study, the other key parameters, aside from the FR, will be compared to understand why the observed result occurs.This way, the outlier can be explained. The scatterplots regarding the removal efficiencies for bacteria and viruses through influence of FRs are given in figures 3.1 and 3.2.

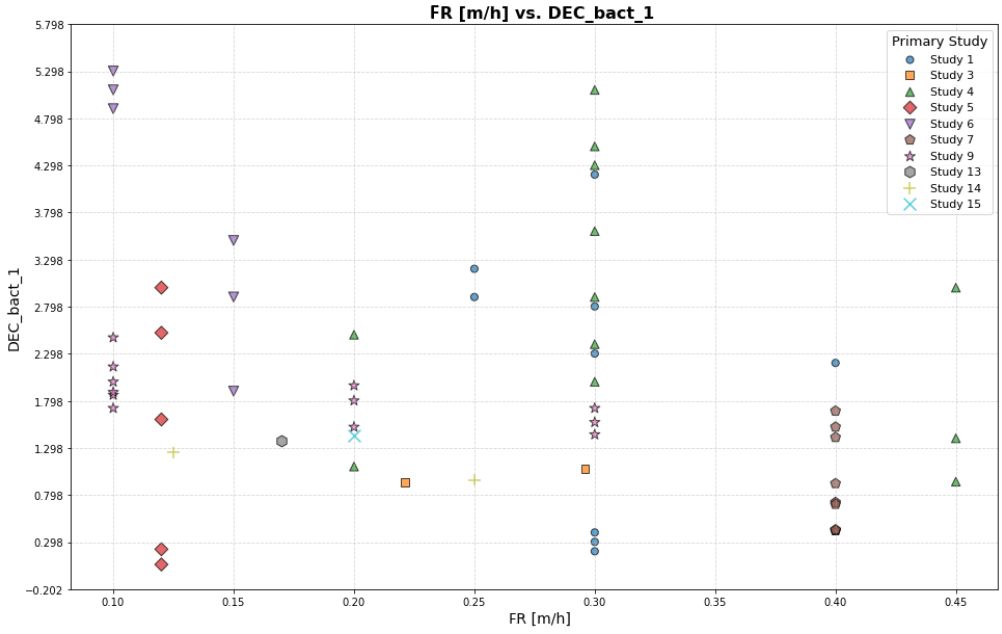

FIGURE 3.1: Scatterplot of FRs against Removal Efficiency of Bacteria. Details on the studies can be found in the dataset, provided in Appendix A

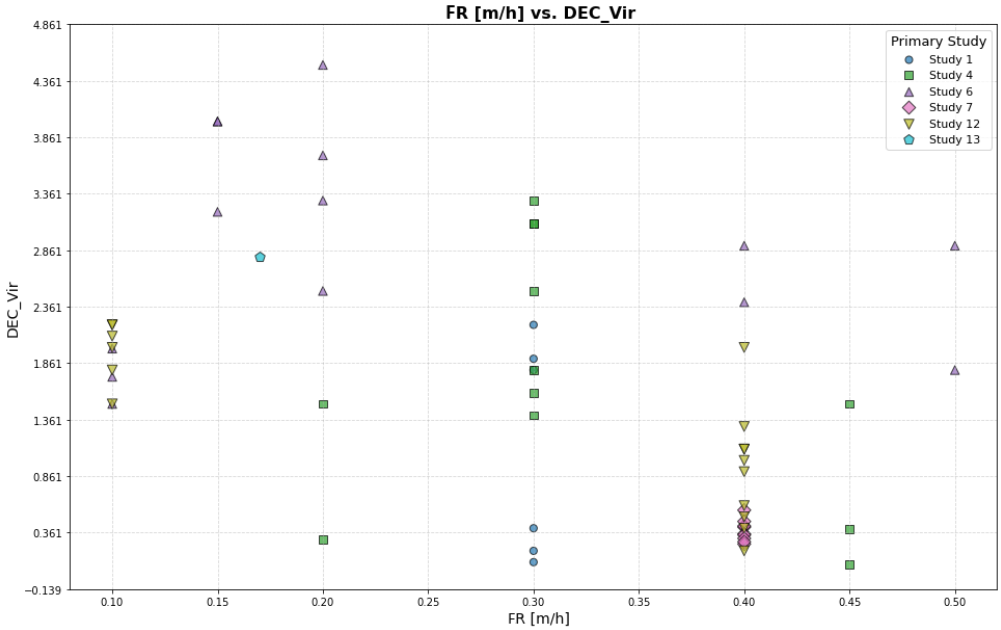

FIGURE 3.2: Scatterplot of FRs against Removal Efficiency of Viruses. Details on the studies can be found in the dataset, provided in Appendix A

In most studies, multiple experiments were conducted using the same FRs but also different FRs to test its effects on the removal efficiency of bacteria and viruses. An example of a study where the same FRs were used, but the experimental outcomes for bacteria and viruses differed significantly, is the study conducted by Jack Schijven et al. [4], depicted as study 4 in the scatterplots. As can be seen from the previous figures, the removal efficiencies varied between > 0.783 and < 5.283, even though the FRs did not change at all or changed slightly. The FR varied between 0.20 and 0.45 m/h in all 13 experiments in this study. The change in FR alone does not explain the high variation between removal efficiencies. To understand better how the other design parameters in correlation with the FR affected the design parameter, three experiments in study 4 were looked at; the experiment with the highest removal efficiency (5.10, experiment 4D3), the experiment that performed adequately (2.90, experiment 4D2), and an experiment that performed poorly (0.94, experiment 4W2). By observing the other design- and operational parameters, the differences in the removal efficiencies can be explained. The most significant differences in design and operational conditions are showcased in table 3.2.

Experiment	FR [m/h]	Grain Size [mm]	Porosity [-]	Age of Schmutzdecke [d]	Temperature [℃]	Removal of Bacteria
4W2	0.45	0.65	0.33	327	4	0.94
4D2	0.30	0.53	0.40	4	14	2.90
4D3	0.30	0.53	0.40	53	16	5.10

TABLE 3.2: Differences in values for design parameters in experiments of study 4 [4]. More information on other design parameters/operational conditions can be found in Appendix A

From the table it is clear to see that the reason why 4W2 has such a low removal efficiency, is due to the low temperatures that is performed on. The reason why 4D2 is lower than 4D3 is due to a less developed *Schmutzdecke*. If that was not the case, the removal efficiency for 4D2 could possibly reach similar results as the removal efficiency found for 4D3, but this is uncertain. In short, if the temperature was high for 4W2, the removal efficiencies would be much higher. Between the three experiments, the experiment 4D3 seems most reliable, as this was done under normal circumstances in terms of temperature and *Schmutzdecke* development.

Another interesting observation to cover is the high removal efficiencies found in experiments of study 6 [16]. Here it can be observed that the removal efficiencies are highest of all the gathered data, namely that of > 5.298. This is interesting because it matches what is commonly found in the literature: lower FRs tend to result in higher removal efficiencies. However, even though this is the case, other experiments in another study, namely that of study 9 [64], showed that removal efficiencies at the same FR value of 0.1 m/h were much lower, namely that of > 2.798. This contradicts the findings from the literature review, as study 4 demonstrated that FRs of 0.3 m/h

achieved removal efficiencies exceeding 5.298. To understand this better, in table 3.3, the biggest differences for operational conditions and design parameters are showcased between the three studies.

Experiment	FR [m/h]	Total Bed Height	Type of Water	Lab/Pilot/Fullscale	Removal of Bacteria
9_{1C}	0.10	0.73	1	Lab	1.86
4D3	0.30	1.24	3	Pilot	5.10
6E	0.10	0.60	3	Lab	5.30

TABLE 3.3: Differences in values for design parameters in experiments of study 4, 6 and 9 [4][16][64]. More information on other design parameters/operational conditions can be found in Appendix A

By taking a look at table 3.3, the differences in removal efficiencies can be explained yet again. The difference between 9_{1C} and 6E is mainly in the type of water that was experimented upon. The type of water has been given the category 3 for both 4D3 and 6E. This category means that the water is pre-treated surface water. 9_{1C} has category 1, which stands for raw surface water. It seems that the experiment in which the water was pre-treated resulted in higher removal efficiency when compared to filtering raw surface water, which is logical and aligns well with earlier mentioned studies. The differences between 4D3 and 6E can also be explained. Even though the FR is smaller for 6E (that of 0.10 m/h), the removal efficiency is still similar to the removal efficiency found in experiment 4D3, even though it uses a higher FR (that of 0.30 m/h). However, the T.B.H. used in 4D3 is higher to that of 6E, almost two times as high. So it could be the case that due to a much higher total bed height, the removal efficiency for a higher FR, is still has high as the removal efficiency for an experiment with a lower FR. All of the ranges can be explained in such a sense that different operational conditions and design parameters all in combination can change the removal efficiency of a SSF. As an example, suppose the T.B.H. in study 6 was also around 0.60 meters, similar to study 4. Then, the removal efficiencies would probably be much lower. This statement holds true for the experiments in study 4 at a FR at 0.45 m/h as well. These were also done at a T.B.H. of above 1 meters. This means that if they were done at a much lower T.B.H., their removal efficiencies would drop. In conclusion, the removal efficiency values from study 4 for a FR of 0.45 m/h can be considered outliers. When these outliers are excluded, the trend shown in figure 3.3 can be more clearly derived from the scatterplot.

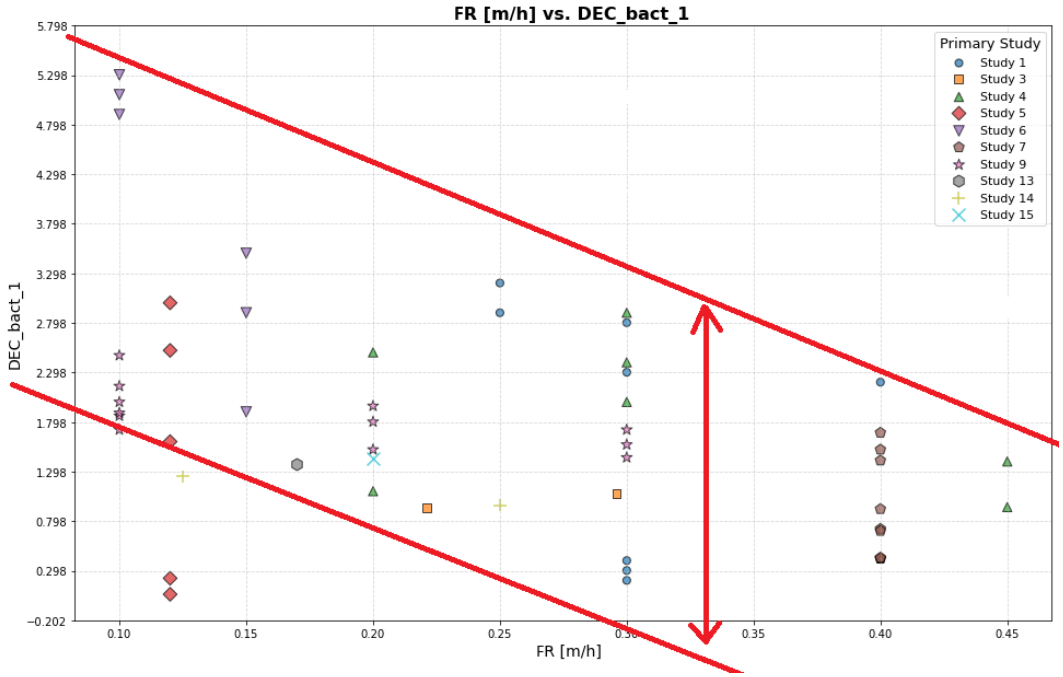

FIGURE 3.3: General downwards trend in scatterplot of FR vs removal efficiency of bacteria

The statement found in literature, that an increase in FR leads to a decrease in removal efficiency, is generally supported when other design parameters remain constant. This downward trend is visible in the scatterplot, highlighted by the red lines. However, variability within this trend, shown by the vertical spread of data points, reflects the influence of factors such as T.B.H., n [-], and D_{10}. As was mentioned before, these parameters can cause deviations and introduce outliers, emphasizing the importance of considering them in experimental design

and analysis. The scatterplot in figure 3.3 illustrates both the overall validity of what was found in literature and the complexity introduced by interacting design parameters.

In the scatterplot of FR vs. virus removal, a downward trend is also evident. However, similar to the scatterplot for the removal of bacteria, a considerable number of outliers can be observed in the scatterplot for the removal of viruses, indicating variability likely caused by differences in other design parameters. To validate this, experiments that deviate from what was found in literature, that a higher FR typically results in a lower virus removal efficiency, will also be closely examined.

One such deviation can be observed when comparing removal efficiencies from studies 6 and 12 [16] [66]. All experiments in study 12, conducted at a low FR of 0.10 m/h, yielded removal efficiencies below 2.361. In contrast, experiments from study 6 using a higher FR of 0.20 m/h demonstrated significantly better removal efficiencies, ranging between 2.861 and 4.861. Furthermore, experiments in study 6 conducted at a higher FR of 0.40 m/h resulted in removal efficiencies that were higher than those observed at an FR of 0.10 m/h in study 12. This further refutes what was stated by researchers in literature.

To investigate this discrepancy, these experiments have been analyzed further to determine whether other factors explain the variation. A closer examination of study 12 highlights why lower removal efficiencies were observed at 0.10 m/h compared to the higher efficiencies at 0.20 m/h in Study 6. The fundamental differences in experimental conditions between the two studies are a critical factor. For example, one experiment was conducted at a pilot scale, designed to replicate real-world filtration conditions, while the other was performed on a small column scale, which may not capture the complexities of full-scale operations. Additionally, differences in environmental conditions, such as temperature, further distinguish the studies. One experiment utilized raw surface water as influent, while the other used artificially inoculated water with specific contaminants. Therefore, direct comparisons between these experiments are not valid due to the inherent disparities in their experimental setups and conditions.

In summary, the scatterplots for both the removal of viruses and bacteria highlight that differences in operational conditions and design parameters significantly influence the observed trends. While the statement derived from literature, that higher FR leads to reduced removal efficiencies, holds true in many cases, variations in parameters such as influent type, experimental scale, temperature, porosity, and total bed height introduce outliers and inconsistencies. These differences emphasize the importance of considering all operational factors when interpreting the results, as they play a critical role in shaping the outcomes observed in the scatterplots.

3.2 The Hydraulic Retention Time

3.2.1 Definition of The Hydraulic Retention Time

An important design parameter that inversely correlates with the FR is the HRT. The HRT is the average time that the water molecules spend in the SSF, from the moment they enter to the moment they leave. It must be noted that the HRT is not one single number, but rather a distribution of times (residence distribution time), as is it is a dependent variable. One particle of water can move faster through the filter in comparison to another particle of water.

The theoretical HRT is dependent on the flow rate, volume and porosity of the filter medium and is defined by the following equation:

$$HRT = \frac{V \cdot n}{Q} \tag{3.1}$$

Where:

Q is the flow rate (m³/h)
n is the sand porosity
V is the total volume of the sand (m³)

From equation (3.1) it can be concluded that a longer HRT corresponds to a lower FR, if the T.B.H. is high, which can reduce the flow rate through the filter. This slower flow will then allow for more thorough filtration as water interacts with the filter media for a longer period. Filters with longer HRTs may also require less frequent maintenance as the slower flow rates can reduce the likelihood of rapid clogging.

As previously mentioned, equation (3.1) demonstrates that the HRT depends on the porosity and volume, and therefore also on the height of the filter bed. The porosity of the filter media has to do with the size and the homogeneity of sand particles in a SSF [52]. The homogeneity can be best presented by the uniformity coefficient *(U.C., or D_{60}/D_{10})*. This coefficient serves as a means to gauge the consistency of particle sizes within a sand sample. If the U.C. is close to one, it means that all the sand particles in the SSF have the same size. An U.C. of one will lead to higher filtration efficiencies. However, U.C.s of one are often not the case. U.C.s higher than one mean that the sand particles in the filter have varying sizes. Thus, the gaps between the larger particles will be filled by smaller particles, which in turn will lead to earlier clogging, uneven porosity, preferential flow paths for water and thus shorter HRTs [67]. Another critical parameter related to sand size is the D_{10}, which refers to the diameter of the sand particle at the 10th percentile of the grain size distribution. The D_{10} directly influences filtration performance, as smaller values of D_{10} correspond to finer grains that enhance the physical and biological removal of contaminants.

3.2.2 Effects of Hydraulic Retention Time on Removal Efficiencies of Bacteria and Viruses for Drinking Water Preparation in Literature

The effects of the HRT, n [-], grain size, D_{10}, T.B.H. and U.C. on the removal efficiency of bacteria and viruses are not clearly defined in the literature. Instead, most studies tend to focus on outlining the boundary values for these design parameters rather than focusing on the effect of increasing/decreasing the value of the parameter. In this section, analysis will be done whether these statements about the boundary values prove to be true or false.

Typical HRTs values in SSFs are recommended to range between 2.5 and 12.5 hours as boundary conditions [1]. Regarding the U.C., higher values are associated with earlier clogging compared to experiments conducted with lower coefficients. In a study by Guasparini et al., an SSF with a U.C. between 3.5 and 3.8 was used for cryptosporidium removal [68]. This approach proved highly inefficient, achieving only 48% removal. Conversely, another study using the same river water applied a much lower U.C. of 1.72 [58]. This adjustment significantly improved removal efficiency, achieving rates exceeding 99%. Logan et al. suggest that the uniformity coefficient should ideally remain below two [20].

When considering the D_{10}, Logan et al. recommend a range of 0.15–0.35 mm for sand in SSFs, with porosity values between 0.35 and 0.50. A decrease in n [-] below 0.35 would result in reduced HRT and lower removal efficiencies.

The impact of grain size on the removal efficiency of bacteria is well-documented in the literature. In literature, it is stated that smaller grain sizes generally result in higher removal efficiencies for coliform bacteria compared to larger sizes [1]. For example, one experiment demonstrated that reducing grain size from 0.62 mm to 0.13 mm increased coliform removal from 96.0% to 99.4%. Similarly, Jenkins et al. observed an increase in the removal rate of indicator bacteria by 0.16 to 0.40 logs when grain size was reduced from 0.52 mm to 0.17 mm [19]. However, for certain pollutants like Giardia cysts, changes in grain size had minimal impact on removal efficiency. Huisman et al. highlight an additional advantage of finer grain sizes: they limit the deep penetration of pollutants into the sand bed, making surface scraping sufficient for filter maintenance [52]. Furthermore, finer grains are associated with lower head losses, while larger grain sizes result in higher head losses due to larger pores between the particles [69].

In summary, smaller grain sizes generally improve bacterial removal efficiency, simplify filter maintenance, and reduce head losses, though their effect may vary depending on the type of pollutant being targeted. Logan et al. further emphasize this point, highlighting effective grain size as one of the most critical parameters influencing bacterial removal.

The thickness of a sand layer also has a significance influence on the HRT and thus removal efficiencies of SSFs, as it will take longer for the influent water to reach the end of the filter, because the filters have a larger volume. According to Abdiyev et al., thicker layers are more effective in the removal of fine and colloidal particles, viruses and turbidity of the influent water due to the previously mentioned statement [63]. However, after a certain thickness, high efficacy is no longer applicable for certain pollutant types. As an example, in a study by Williams, P.G., it was concluded that for the removal of bacteria, a 200 mm sand layer was sufficient for removing 99.5 % of fecal bacteria [70]. If turbidity and coliform bacteria were to be removed as well, a 300 mm sand layer would be suitable and in order to remove all viruses, a 600 mm layer would suffice [71]. In another study by Nancy et al., it was stated that biological activity occurred up to 0.5 m into the SSF [72], as a maximum removal of 98 percent removal of coliform organisms was achieved at 0.5 m of the filter. In the same study, bacterial concentrations in the influent water showed no significant differences at various sand depths. Consequently, it was concluded that the sand bed thickness beyond 0.5 meters had no substantial effect on bacterial removal. For other types of

contaminants such as suspended solids, BOD_5, NO_2, $NO_3{}^-$, TSS, along with the pH and conductivity, Nancy et al. have observed that the concentrations decreased substantially up to a depth of 1 m. The study by Nancy et al. further highlights that varying sand layer thicknesses can achieve high removal efficiencies, depending on the specific contaminant being targeted in the experiment.

In summary, in all previously mentioned studies, it has been stated that the thickness of the sand bed does not have a significant correlation with the removal of bacteria or viruses, as most biological activity occurs in the upper portion of the filter bed. Nevertheless, the effectiveness of bacteriological treatment becomes influenced by the depth of the bed when dealing with larger sand sizes. This occurs due to the reduction in the total surface area within the filter in a sand bed with larger grains. Additionally, higher flow rates may occur, leading to potential increases in percolation rates [73].

From literature study, the following **remarks** can be made regarding aforementioned design parameters in this section:

- Typical HRTs values in SSFs are recommended to range between 2.5 and 12.5 hours
- Higher U.C. values lead to earlier clogging compared to filters with lower U.C. values.
- For optimal performance, the U.C. should generally be below 2 [20].
- Sand porosity should ideally range between 0.35 and 0.50 for effective performance [20].
- Smaller grain sizes lead to higher removal efficiencies for coliform bacteria [1].
- Biological activity is concentrated in the upper 0.5 m of the filter bed [72].
- A 600 mm sand layer is necessary for virus removal [71].
- Larger grains in thicker beds may reduce total surface area, potentially lowering removal efficiencies[73].

3.2.3 Exploratory Data Analysis of Effects of Aforementioned Design Parameters in the Removal of Bacteria and Viruses

By analyzing the scatterplots of HRT against the removal efficiencies of bacteria and viruses, it becomes evident that the results largely align with the literature, showing that higher HRT generally leads to improved removal efficiency, as illustrated in figures 3.4 and 3.5. In most studies, there is a clear pattern where an increase in HRT corresponds to enhanced removal efficiencies. This trend is notably observed in the studies conducted by Arora et al., Wim et al., and Muhammad et al. [16] [53] [64], among others.

Some studies, however, deviate from what was stated in literature. A notable example is Study 5 [8], where all experiments conducted at an HRT of approximately 13 hours resulted in lower removal efficiencies for bacteria compared to experiments performed at shorter HRTs of 6 hours. Unfortunately, the reasons for this deviation could not be determined from the available literature. It is possible that differences in operational conditions contributed to these results, but without further details, it remains speculative. Interestingly, Study 5 utilized a larger total bed height and a more developed *Schmutzedecke* compared to the experiments in study 6 at a HRT of 6 hours. Both these factors are typically associated with improved removal efficiencies. Yet, the outcomes contradicted the literature, as the removal efficiencies decreased instead of increasing as expected. This inconsistency highlights the complexity of the factors influencing SSF performance and underscores the need for more detailed data to understand such deviations.

For other experiments, however, the lower efficiencies at higher HRTs can be explained by differences in operational conditions. For instance, in Study 9 [64], an HRT of 7.3 hours yielded a removal efficiency of 2.47, whereas in Study 6, an HRT of 6 hours achieved a significantly higher removal efficiency of 5.30. A key factor contributing to this discrepancy is the type of influent water used. In Study 9, raw surface water was utilized, while in Study 6, pre-treated water was employed. The pre-treatment process likely reduced the initial contaminant load, making it easier for the SSF to achieve higher removal efficiencies. Lastly, the scatterplot suggests that the initial statement regarding an optimal HRT range of 2.5 to 12.5 hours holds true. A parabolic trend appears, with the highest removal efficiencies observed around 6 hours.

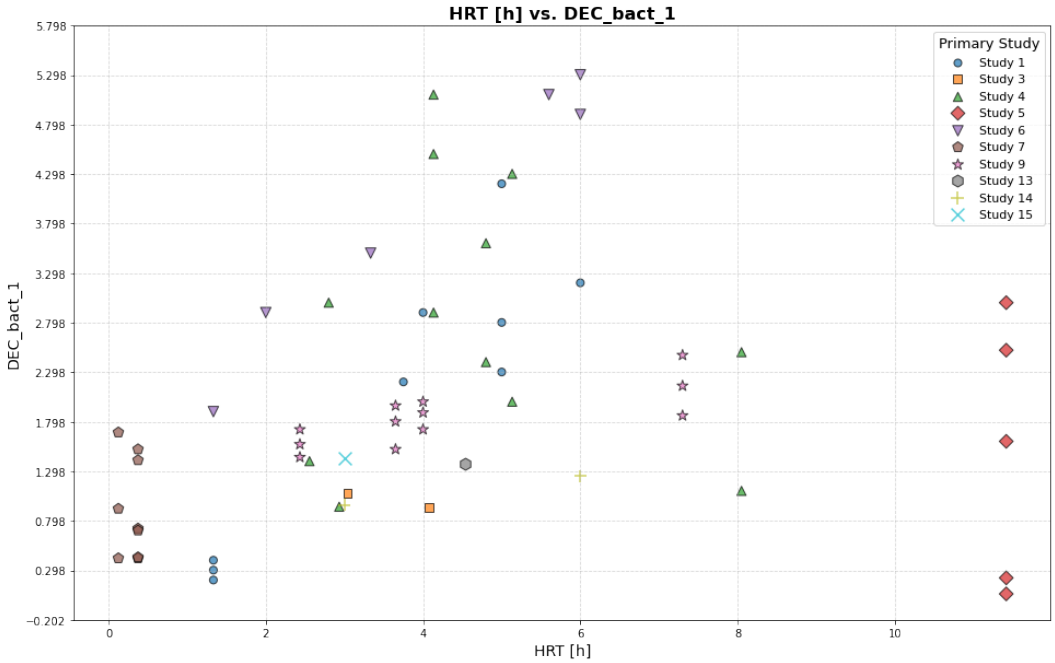

FIGURE 3.4: Scatterplot of HRTs against Removal Efficiency of Bacteria. Details on the studies can be found in the dataset, provided in Appendix A

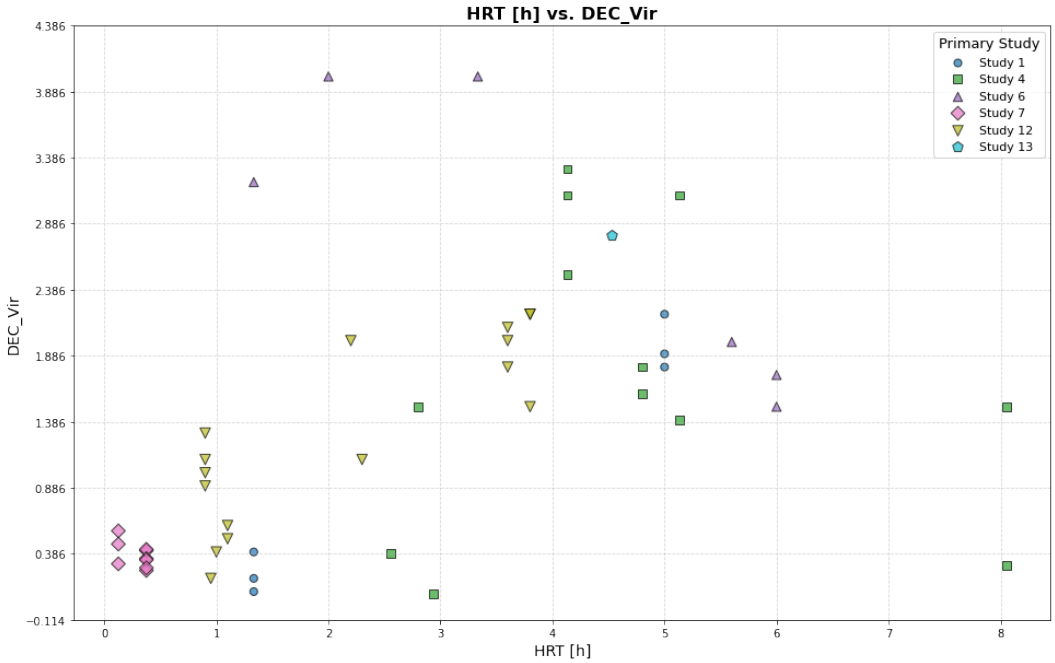

FIGURE 3.5: Scatterplot of HRTs against Removal Efficiency of Viruses. Details on the studies can be found in the dataset, provided in Appendix A

As discussed earlier in Section 3.1.3, variations in design parameters and operational conditions complicate the ability to definitively confirm or refute what is stated in the literature. Nevertheless, the overall trend aligns with well with what was found in literature, suggesting that higher HRTs generally result in improved removal efficiencies.

3.3 *Influence of the Schmutzdecke and Temperature*

The previously mentioned design parameters such as FR, HRT, particle size distribution and layer thickness all have a significant influence on the development of the *Schmutzdecke*, as was made clear and explained in the previous sections. The *Schmutzdecke* is formed after weeks to months of operation within the first 2 to 5 cm of the sand layer of the filter [1]. Up to 60 % of bacteria is removed within this layer [74] and 0.56 $\log_{10}$ removal of bacteriophage MS2 occurs per centimeter of the *Schmutzdecke* [75]. In comparison, within the rest of the filter, the removal of bacteriophage MS2 is only 0.06 $\log_{10}$ per centimeter. Thus, it is very important for a SSF to develop this biofilm layer. The importance of the *Schmutzdecke* can also be noticed after cleaning the filter. An example of this was found in a study by Wim et al. in which it was observed that scraping off the *Schmutzdecke* let to a decrease in the removal rate of bacteria by 1-2 $\log_{10}$ [53].

Other studies have found that the removal of pollutants in influent water depended strongly, not only on the *Schmutzdecke*, but the temperature of the water as well [4][63][76][77]. This was also derived in the previous section, when examining the scatterplots. When the weather is cold, conducting the filtration process indoors is recommended, even more so when subzero temperatures arise. Many experiments have been conducted in which it was showcased that by lowering the temperature, the removal rates for several pollutants were lowered. For example, in an experiment by Grützmacher et al. on the removal of microcystins [78], it was discovered that, when the temperature was decreased from 20 ^{o}C to > 4 ^{o}C, the removal rate of Microcystins decreased from >85 % to <60 %, reason being that the bacterial bio-degradation was slowed down due to the lower temperatures of operation. Other studies have also shown that decreases in temperature led to decreases in removal rates for other pollutants such as E.Coli, COD and TOC [63][79]. In contrast to this, other studies found that differences in temperature did not decrease the concentrations of some pollutants, such as Giardia lamblia or Cryptosporidium, under specific operation conditions [80]. Such an example can be found in the study of Fogel et al., in which it was stated that the temperature, which ranged from 0.5 to 20.0 ℃, did not have any direct influence on the removal efficiency of Giardia, Coliform and Cryptosporidium in the SSF [81]. It is mentioned by Fogel et al. as well that the temperature of the raw water has an indirect influence on the presence of cryptosporidium oocysts as the concentrations of oocysts in raw water may be dependant on its temperature. Through statistical analysis, it was

discovered by Fogel et al. that in colder waters, the presence of oocysts is significantly higher than in warmer waters.
In contrast to design parameters such as the FR, which can be deliberately chosen and adjusted during the design of SSFs, operational conditions like the *Schmutzdecke* and temperature cannot be directly controlled or selected. These operational conditions evolve naturally or depend on external environmental and design factors, making them distinct from the deliberately engineered aspects of SSF systems.

3.4 Key Findings

The analysis reveals that SSF performance depends on the interconnected relationships between key design parameters such as FR, HRT, D_{10}, and n [-]. No single parameter can fully explain removal efficiencies, as interactions between them often amplify or counteract individual effects. External factors, such as temperature, influent quality and the development of the *Schmutzdecke* also play a crucial role.

While scatterplots offer valuable visual insights, they fall short in capturing the nonlinear and multivariate nature of these interactions. Advanced modeling techniques, like XGBoost, are essential for accurately quantifying these complex relationships and guiding optimization strategies.

In conclusion, a holistic approach that integrates parameter interactions, environmental conditions, and iterative modeling refinements is essential for optimizing SSF design and achieving reliable contaminant removal efficiencies. In the next chapter, the results of this approach are presented.

Chapter 4

Results from Modeling: Insights into Design Parameters and Efficiency

This chapter presents the outcomes of the analysis conducted to evaluate the performance of the predictive models developed to estimate removal efficiencies for bacteria and viruses. The focus is placed exclusively on the XGBoost models, which were chosen for their ability to handle non-linear relationships, interactions between variables, and robustness against multi-collinearity and missing data. The results are organized as follows: first, the performance of the XGBoost models is presented, including their predictive accuracy on both cross-validation and test datasets. Then, the relative importance of design and operational parameters is explored through feature importance analyses, providing insights into their individual and combined contributions to model performance. Additionally, the chapter examines the comparative ability of the XGBoost models to predict bacterial and viral removal efficiencies, shedding light on differences in predictive accuracy and parameter relevance for each target variable. The findings highlight the strength of XGBoost as a predictive tool for optimizing SSF design and underscore the importance of key design and operational parameters in determining removal efficiencies.

4.1 Performance of XGBoost Models for Removal Efficiency for Bacteria

The predictions for bacterial removal efficiency using the XGBoost model show strong alignment with the actual values, as illustrated in the scatter plot of Actual vs. Predicted DEC_{bact1} in figure 4.1. This indicates that the model performs well in predicting bacterial removal, with an R^2 score of 0.968 on the test set and a MSE of 0.0641. The cross-validated R^2 score, while slightly lower at 0.599, reflects a reliable generalization performance. Furthermore, the feature importance analysis, obtained from the XGBoost model, provides insight into the relative contributions of each design parameter to the prediction of bacterial removal efficiency (`DEC_bact_1`). Figure 4.2 illustrates the importance scores assigned to each feature by the model. This version of the model used all the available features to evaluate their combined predictive power. However, as explained in the methodology, feature reduction was explored to determine the subset of features that yielded the lowest BIC. Based on the BIC, the optimal amount of features for predicting the removal efficiency for bacteria was equal to 5, as can be seen in figure 4.3. The most important features can be seen in figure 4.4. Reducing the number of features not only improved computational efficiency but also made the model clearer and easier to interpret. By reducing the number of features, the model's performance improved, as indicated by an increase in both the test set R^2 score (0.981) and the cross-validation R^2 score.

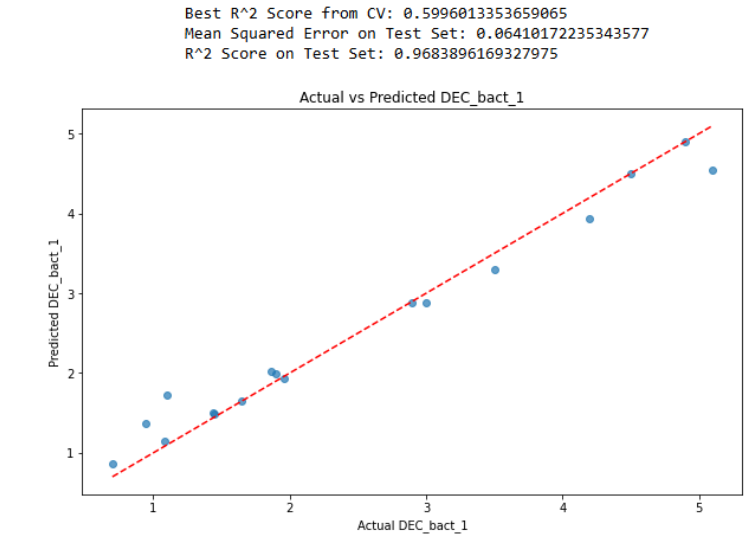

FIGURE 4.1: Prediction of bacteria removal using all features, showcased in figure 4.2, in the XGboost model

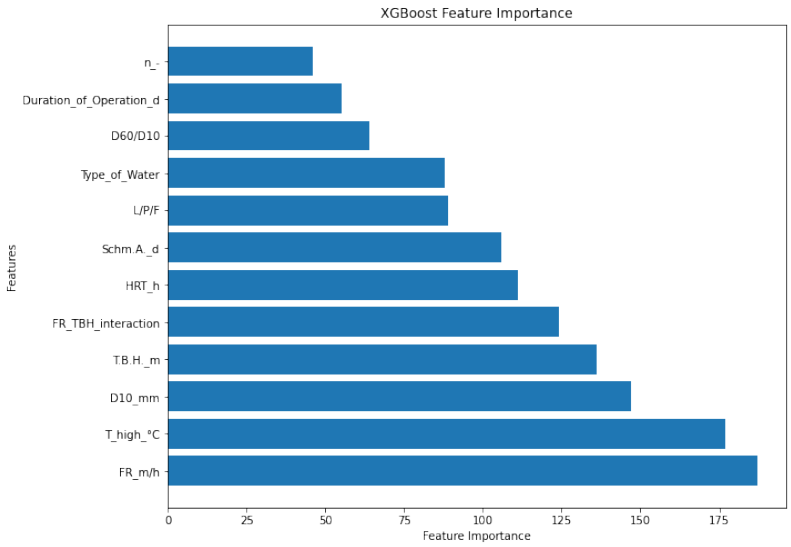

FIGURE 4.2: Features ranked from least to most important, in predicting bacteria removal for XG-Boost

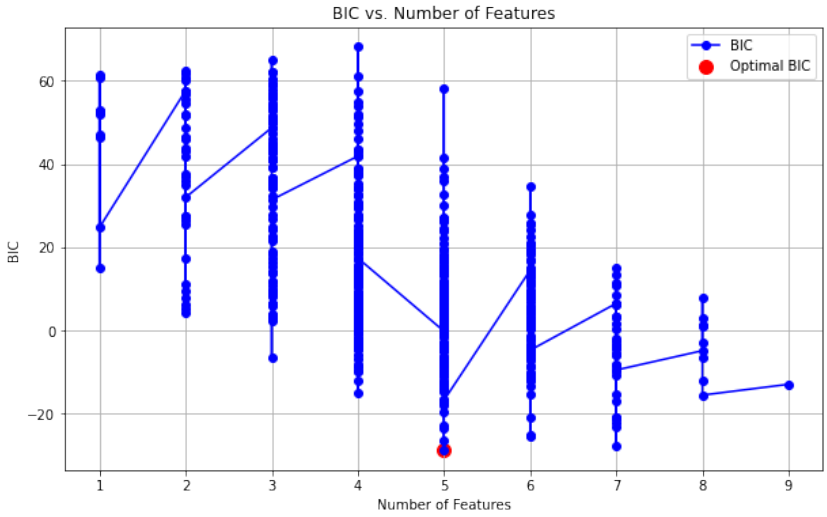

FIGURE 4.3: Determining the optimal amount of features of the XGBoost model using the BIC for Removal of Bacteria. The dots represent all possible combinations of features and their BIC values

The vertical lines in the BIC plot represent models with the same number of features, while the dots along each line correspond to all possible combinations of those features and their respective BIC values. The spread of dots shows that different feature combinations yield varying BIC scores for the same feature count. The red dot highlights the combination of features with the lowest BIC, indicating the optimal balance between model complexity and fit.

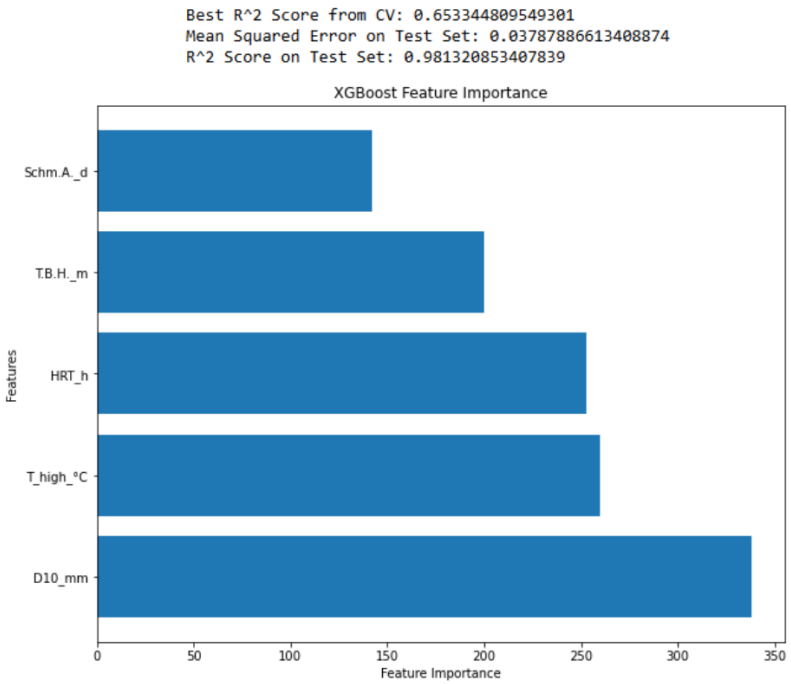

FIGURE 4.4: Key Features Identified by XGBoost for Predicting Bacterial Removal Efficiency After Excluding Non-Contributory Features through BIC

Interestingly, the most important features derived from the original model, shown in Figure 4.2, differed from those identified in the model operating on only five features. However, the five-feature model demonstrated greater reliability, as indicated by a higher R^2 score for the cross-validation datasets. The R^2 for the test set and the MSE was lower, however, when using the BIC parameters, compared to making use of either the top 5 parameters that were important or using all parameters. The shift in feature importance can likely be attributed to the reduction of redundancy, multicollinearity, and overfitting, allowing the model to focus more effectively on the truly influential parameters.
The selected features—D_{10}, *temperature*, *HRT*, *T.B.H.*, and *age of the Schmutzdecke*—reflect the key parameters influencing bacterial removal efficiency. For some of these features, such as D_{10} (*D10_mm*), temperature (*T_high_* ℃), and HRT (*HRT_h*), their impact on removal efficiency was already logical and previously discussed. Interestingly, parameters such as porosity (*n [-]*), duration of operation, FR, pre-treatment level of the water, and whether the experiment was conducted on a lab-, pilot-, or full-scale system appear less important in the final model. This can be attributed to the fact that their effects are already indirectly captured by the selected features. For example, FR and n [-] influence the HRT, while the duration of operation and maintenance account for changes in the development of the *Schmutzdecke*.

In summary, the final XGBoost model demonstrates strong predictive performance for bacterial removal efficiency. This is reflected in the high R^2 score of 0.981 on the test set and a notable improvement in cross-validation R^2 to 0.653, showcasing the model's ability to generalize well. The relatively low MSE of 0.037 further indicates that the predictions closely align with the observed values. In comparison, the other models (OLS regression and DT) performed considerably worse as can be seen in table 4.1.

Model	**MSE**	**R^2 (Training/CV)**	**R^2 (Test)**
OLS Regression	1.03	0.535 (Training)	-
Decision Tree (DT)	0.356	-0.141 (Training)	-
XGBoost (All Params)	0.0641	0.599 (CV)	0.968
XGBoost (Top 5 Params)	0.082	0.509 (CV)	0.905
XGBoost (BIC Params)	0.16	0.655 (CV)	0.813

TABLE 4.1: Performance comparison of different models for predicting the removal of DEC_{BACT}.

4.2 Performance of XGBoost Models for Removal Efficiency for Viruses

The same approach used for predicting bacterial removal efficiency was applied to predict the removal efficiency of viruses using the XGBoost model. First, the model was trained using all available features, and the results, including feature importance and model performance, are presented in figures 4.5 and 4.6. The model predicting virus removal outperformed the bacterial removal model, achieving an R^2 score of 0.965 on the test set and 0.798 during cross-validation.

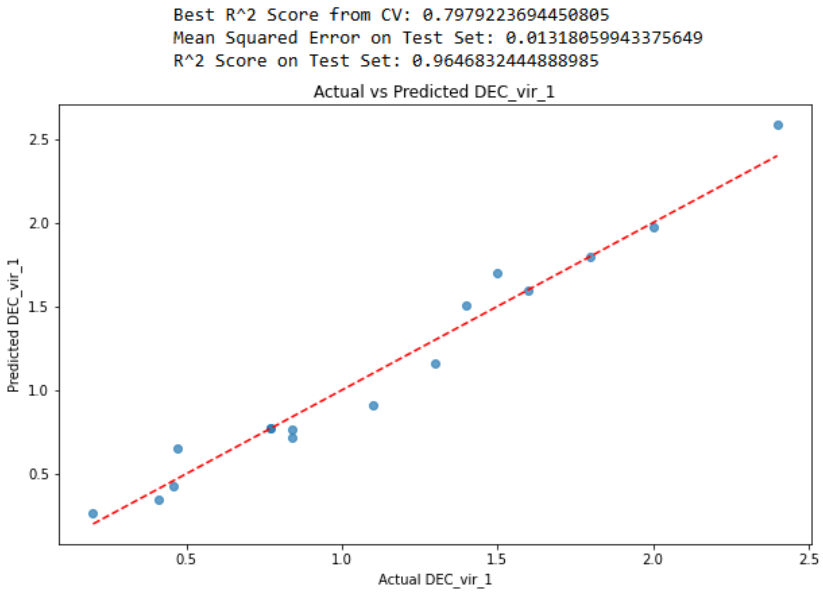

FIGURE 4.5: Prediction of virus removal using all features, showcased in figure 4.6, in XGboost model

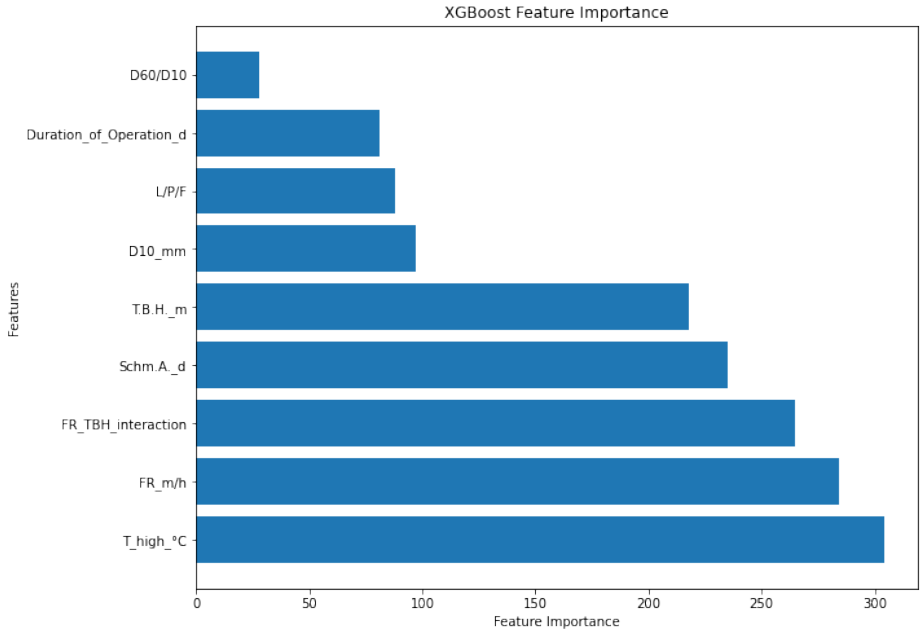

FIGURE 4.6: Features ranked from most import to least, in predicting virus removal for XGBoost

Next, the number of features was reduced based on the BIC, which identified three features as optimal, as can be observed in figure 4.7. While three features may not seem enough for predicting the removal efficiency, the BIC ensures that this number provides an optimal balance between simplicity and predictive accuracy. Adding more features could lead to overfitting, capturing noise instead of meaningful patterns, while reducing interpretability [82]. The results for the reduced-feature model are shown in figure 4.8 on the next page. The feature importance analysis reveals that temperature is the most influential factor for virus removal efficiency, emphasizing its role in enhancing microbial activity and biofilm performance. The interaction between FR and T.B.H., represented by their combined factor, captures non-linear effects, reflecting how their joint influence affects processes like microbial activity or particle retention, beyond the scope of linear relationships For instance, while a higher FR could reduce contact time, the T.B.H. may counterbalance this by providing greater filtration depth, resulting in a more complex and context-specific impact on removal efficiency. This interaction term differs from the HRT, as the HRT is a represented as a ratio (T.B.H. / FR), focusing on how changes in one parameter relative to the other affect the time water spends in the filter. Lastly, filtration rate independently contributes to removal efficiency by influencing the time viruses remain in the filter.

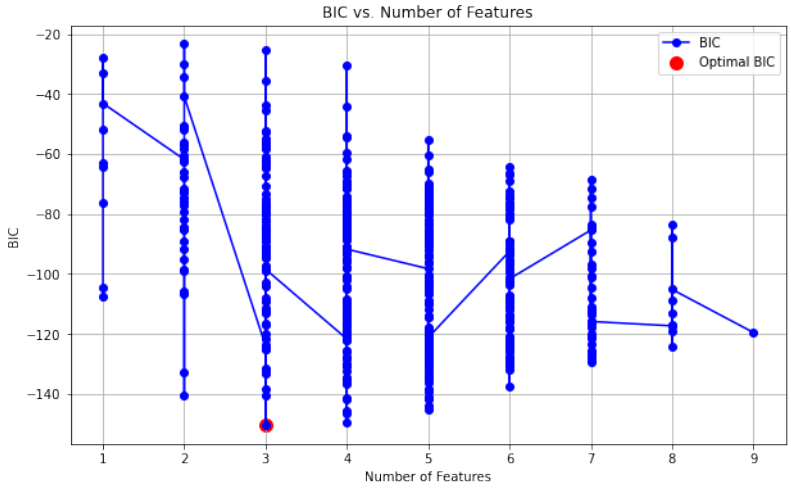

FIGURE 4.7: Determining the optimal amount of features of the XGBoost model using the BIC for Removal of Viruses

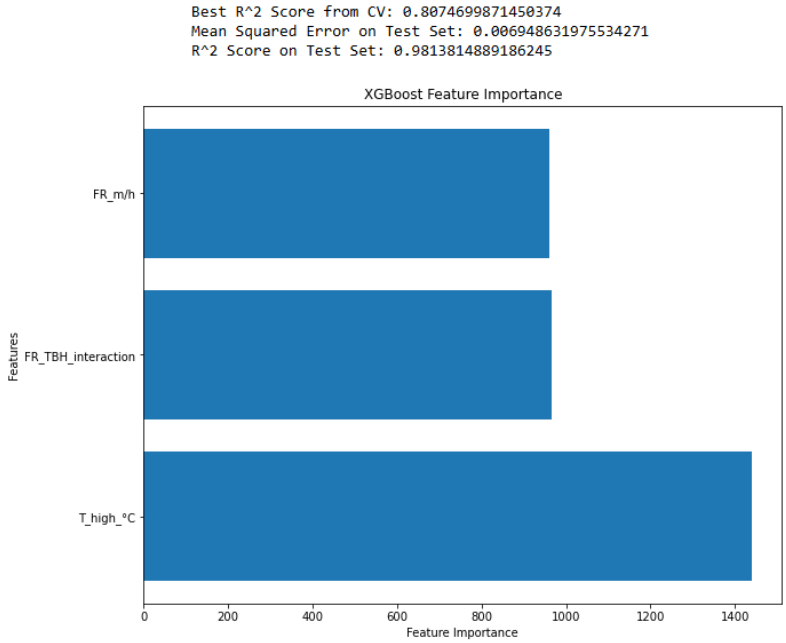

FIGURE 4.8: Key Features Identified by XGBoost for Predicting Virus Removal Efficiency After Excluding Non-Contributory Features through BIC

The final XGBoost model for virus removal prediction demonstrates strong predictive performance as well. This is reflected in the high R^2 score of 0.981 on the test set and in cross-validation R^2 (0.80) (Figure 4.8, meaning this model can generalize well. The relatively low MSE of 0.0069 further indicates that the predictions closely align with the observed values.

4.3 Interactive Analysis of Feature Contributions to Removal Efficiency

4.3.1 Interactively predicting removal efficiency through the use of sliders

After training the XGBoost regression model and reducing the number of features to the most significant ones, an interactive tool was developed to analyze and visualize the relationship between key design parameters and removal efficiencies for bacteria and viruses. Using the XGBoost regression model combined with SHAP (SHapley Additive exPlanations) values, both local (single prediction) and global (overall model behavior) feature contributions are quantified and visualized. SHAP is a model interpretability technique rooted in game theory, designed to explain the output of machine learning models by attributing contributions to individual features [83]. Each prediction is broken down into a base value (the average model prediction across the dataset) and the SHAP values, which represent how much each feature contributes to pushing the prediction above or below this baseline. In the context of this analysis, local SHAP values explain how each input parameter influences a single prediction, while global SHAP values aggregate these contributions across the entire dataset to reveal overall feature importance. Adjusting the parameters updates the predicted bacteria and virus removal efficiency in real-time [4.9] [4.10] . This dynamic prediction allows exploration of the influence of each parameter configuration on the filtration efficiency.

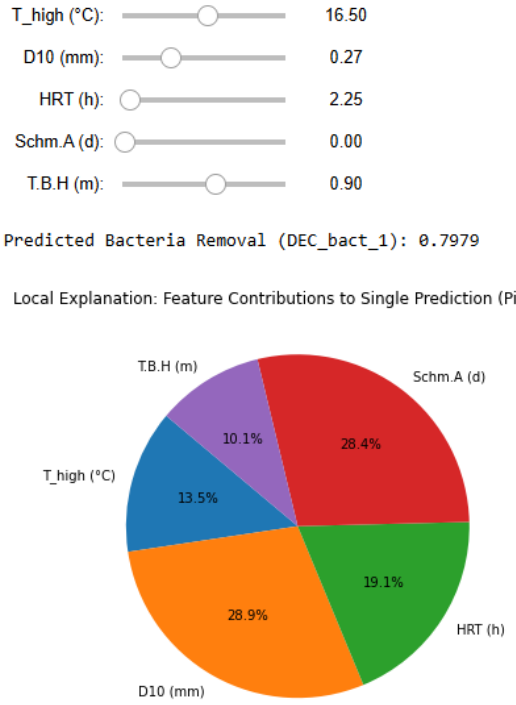

FIGURE 4.9: Interactive Prediction and Feature Contribution Analysis for Bacteria Removal Efficiency

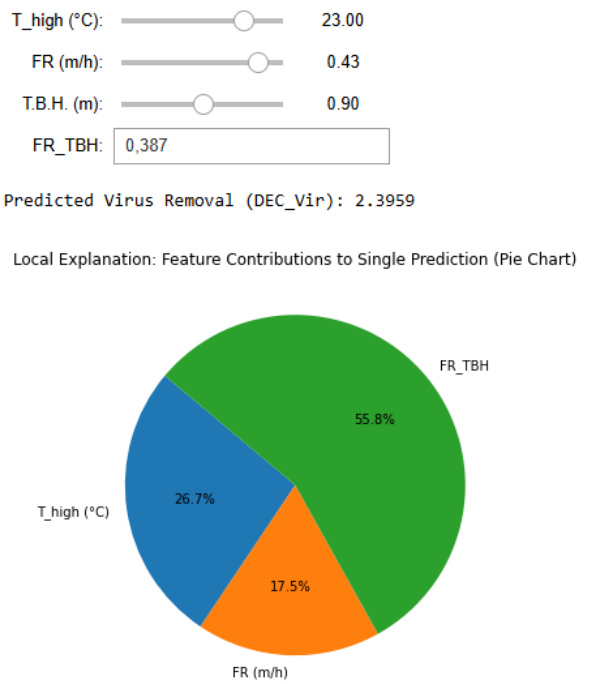

FIGURE 4.10: Interactive Prediction and Feature Contribution Analysis for Virus Removal Efficiency

FIGURE 4.11: Comparison of Interactive Prediction Interfaces for Bacteria and Virus Removal Efficiency using the XGBoost Model

The pie charts in figures 4.9 and 4.10 illustrates the contribution of each feature to an individual prediction. In the displayed example of figure 4.9, the D_{10} contributes the most (28.9%) to the predicted efficiency, followed closely by the age of the *Schmutzdecke* (28.4%). Other parameters, such as HRT and temperature, exhibit moderate contributions, while T.B.H has a smaller share. This visualization emphasizes how feature contributions can vary depending on specific parameter values. In the displayed example of figure 4.10, the temperature seemed to have most effect.

While the pie charts provide an intuitive overview of how features influence an individual prediction, they represent only a snapshot of a single instance. To gain a broader understanding of how these features behave across a wider range of predictions, a deeper examination of local feature importance through SHAP bar charts is essential. These bar charts offer a clearer perspective on how each feature contributes in varying contexts and highlight the dynamic nature of feature effects as input values change. Figure 4.12 and Figure 4.13 present the local feature importance for bacteria and virus removal efficiencies, respectively, based on SHAP values. These bar charts provide insights into how each feature contributes to a specific prediction, with contributions dynamically changing as feature values are adjusted.

For example, in the virus removal model (Figure 4.13), temperature and the interaction term between FR and T.B.H. stand out with a strong positive contribution, whereas FR contributes negatively. The magnitude of these SHAP values suggests that the temperature and interaction between FR and T.B.H. parameter have a dominant and positive effect on virus removal efficiency predictions, consistent with their prominence in the pie chart visualization, as can be seen in figure 4.10.

Together, these visualizations demonstrate how local feature importance varies based on the specific configuration of input parameters. The pie charts provide an intuitive representation of feature contributions to individual predictions, while the SHAP bar charts offer deeper insights into both the magnitude and direction of these contributions.

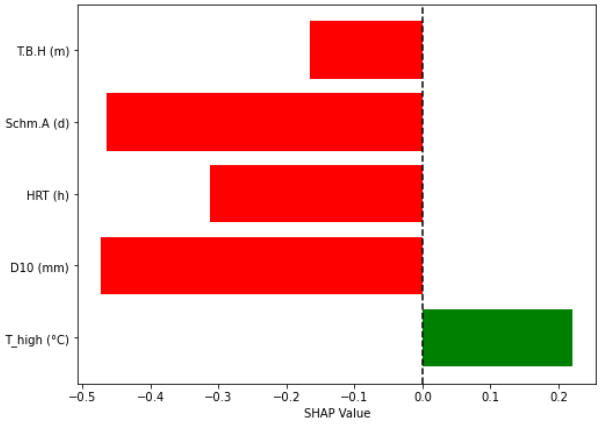

FIGURE 4.12: Local Feature Importance for Bacteria Removal Efficiency Using SHAP Values

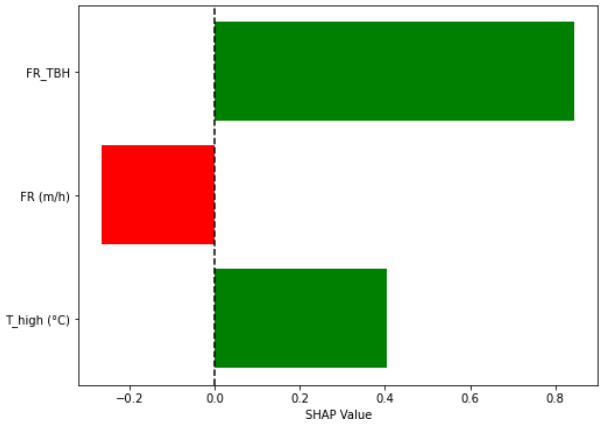

FIGURE 4.13: Local Feature Importance for Virus Removal Efficiency Using SHAP Values

FIGURE 4.14: Local Feature Importance for Bacteria and Virus Removal Efficiency Using SHAP Values. The bar chart illustrates the positive or negative impact of each feature on removal efficiency of bacteria and viruses, respectively

4.4 Model Performance Based on Controllable Design Parameters

In the results presented earlier, both design parameters (e.g., FR, T.B.H., D_{10}) and operational parameters (e.g., temperature, *Schmutzdecke* age) were considered to predict the removal efficiencies of bacteria and viruses. However, while operational parameters play a significant role in determining removal efficiency as mentioned prior, they are inherently less controllable in practical scenarios. For instance, temperature is largely dictated by environmental conditions, and the *Schmutzdecke's* development follows a natural biological process over time. In contrast, design parameters offer direct control to engineers and operators during the planning and optimization of SSFs.

This section focuses specifically on the predictive performance of the model when restricted to controllable design parameters. By isolating these parameters, the aim is to understand their individual contributions and interactions, as well as evaluate how reliably they can be optimized to achieve high removal efficiencies.

The model was retrained for prediction of both bacteria and viruses, using only the controllable design parameters listed below. The U.C. and the scale of the experiment (lab, pilot, or full scale) were added as controllable features in the model because sufficient data were available for these parameters. Although these features are not necessarily emphasized in the literature as important for the removal of bacteria and viruses, their inclusion provided the model with additional data to improve its performance.

- The U.C.
- The Filtration Rate
- The Effective Size
- The Hydraulic Retention Time
- Lab scale / Pilot scale / Full scale operation (-)

The T.B.H., n [-], and grain size from the original set of design parameters, shown in Figure 2.1, were excluded. T.B.H. was omitted because it is already accounted for in the HRT. Similarly, grain size was excluded as it is represented by the D_{10}. Finally, n [-] was left out because the porosity of sand typically remains consistent across different filter setups.

The performance of the model's prediction on both bacteria and virus removal is given in figures 4.15 and 4.16, respectively.

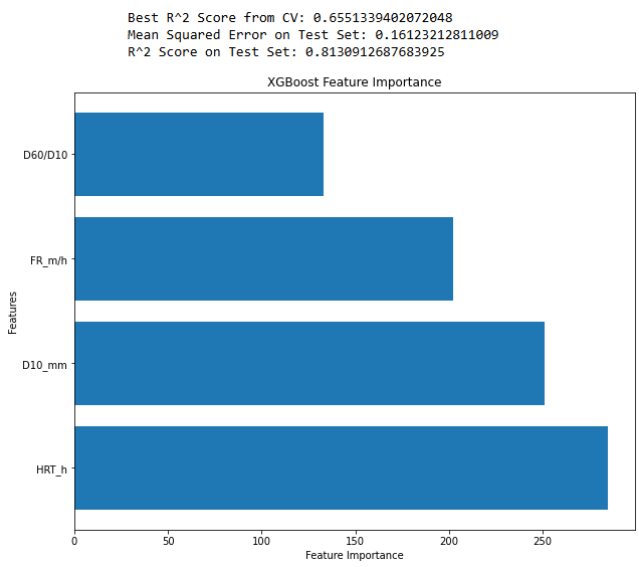

FIGURE 4.15: Interactive Prediction and Feature Contribution Analysis for Bacteria Removal Efficiency

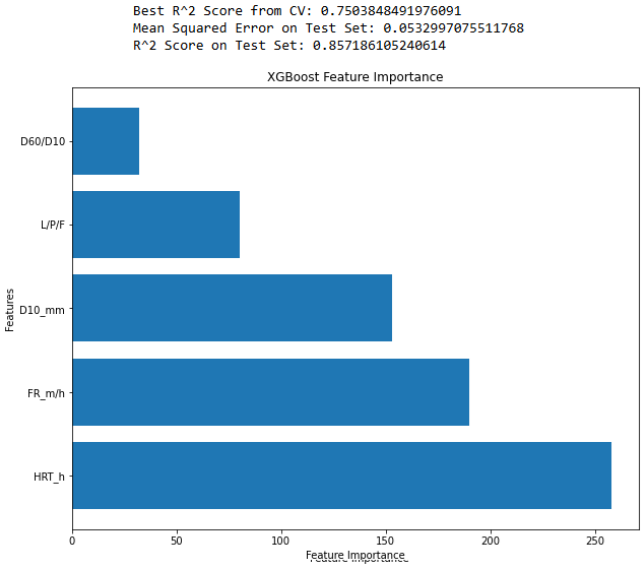

FIGURE 4.16: Interactive Prediction and Feature Contribution Analysis for Virus Removal Efficiency

FIGURE 4.17: Performances of the XGBoost Model for Bacteria and Virus Removal, using only controllable parameters

When relying exclusively on design parameters, the models maintain a reasonable predictive accuracy, albeit with some reduction in performance. For bacteria removal, the R^2 score on the test set decreased to 0.813, compared to the earlier value of 0.981 when operational parameters were included. However, the cross-validation R^2 score remained relatively stable at 0.655. For virus removal, the model performed better, with an R^2 score of 0.857 on the test set and 0.750 for cross-validation. Despite these relatively high values, there was still a noticeable decline compared to the model's performance when operational parameters were considered. The exclusion of temperature and *Schmutzdecke* development introduces a degree of variability and uncertainty in the predictions. This suggests that while design parameters provide actionable insights for SSF optimization, they cannot fully replicate the nuanced interplay and dependencies introduced by operational conditions.

In the design-focused model, optimal ranges for controllable parameters can be conclusively identified, forming a robust baseline. Meanwhile, the operational-focused model reveals how temperature, *Schmutzdecke* age, and other factors dynamically affect removal efficiency.

4.5 Interactive Video Analysis of Design Parameter Effects on Removal Efficiency and Key Findings

A video was created to showcase the interactive analysis of different design and operational parameter setups and their effects on removal efficiency predictions of bacteria. By systematically adjusting parameters such as the temperature, D_{10}, HRT, the age of the *Schmutzdecke*, and the T.B.H., several unique configurations were explored. The video aimed to interpret whether the observed contributions aligned with expectations based on established literature. While the video focused on bacteria, the same interactive approach was applied to evaluate prediction performance for virus removal, ensuring consistency across both contaminant types. The video is available on YouTube and can be viewed here: Interactive Analysis of Design Parameter Effects on Removal Efficiency in Slow Sand Filtration Systems.

Using the interactive tool, heat maps were created that showcase interactions between two design parameters and/or operational conditions and the prediction of removal efficiencies for both bacteria and viruses. The heatmaps for prediction of bacteria and virus removal are shown in Appendices F and G, respectively. Although two-dimensional heat maps provided valuable information about pairwise interactions, they were inherently limited in capturing the interplay among three or more parameters simultaneously. Due to this, multidimensional visualizations were created as well. The multi-dimensional visualizations can be observed in Appendix H

To validate the reliability and logical consistency of the tool's predictions, several key findings are presented in Appendices I and J, corresponding to the model using both controllable and uncontrollable parameters, and the model using only controllable parameters, respectively. These findings will focus on whether the observed trends align with established knowledge from existing literature presented in this MSc thesis report on SSFs. By showcasing specific examples, the tool's ability to replicate known relationships and uncover nuanced insights will be demonstrated. For the removal of bacteria, key insights using both controllable and uncontrollable features in the XGBoost model are presented in tables I.1 and I.2, while those using only controllable features are shown in table J.1. Similarly, for the removal of viruses, insights using both controllable and uncontrollable features are provided in table I.3, and those using only controllable features are shown in table J.2.

Chapter 5

Discussion

This chapter provides an in-depth interpretation of the findings presented in the previous chapter, offering a comprehensive analysis of how the results address the research questions and objectives outlined at the beginning of this study. The discussion explores the significance of key design parameters and operational conditions, their interactions, and their impacts on the efficiency of SSFs for bacterial and virus removal. Additionally, the findings are compared with existing literature to highlight agreements and discrepancies. The strengths and limitations of the study are critically examined, followed by practical implications and recommendations for future research. Ultimately, this chapter aims to synthesize the insights gained from the results, contextualize them within the broader field of water treatment technologies, and offer actionable conclusions for both academic and practical applications.

5.1 Answering the Main Research Question

5.1.1 Restating the Main Research Question and Objectives

This study aimed to address the knowledge gap surrounding the design and optimization of SSF operations. The primary research question guiding this research was:

"What range of values for design parameters are most compatible with SSFs, and ensure that the efficiency of contaminant removal is not compromised?"

In pursuit of this objective, four sub-questions were formulated. These sub-questions sought to identify which design parameters and operational conditions, both controllable and uncontrollable, had the most significant impact on removal efficiencies and how changes in these influence contaminant removal.

5.1.2 Addressing the Sub-Questions Through Model Results and Exploratory Data Analysis

During the research, six design parameters were selected for analysis regarding their impact on the removal of both viruses and bacteria. These parameters included n [-], D_{10}, U.C., T.B.H., FR, and HRT, as they were consistently reported across various studies. However, the XGBoost model revealed that some of these parameters had limited influence on contaminant removal. In contrast, other factors, such as temperature and the age of the *Schmutzdecke*, played a more significant role in explaining the removal efficiencies of bacteria and viruses. Consequently, the focus shifted from the initially selected parameters to those factors and operational conditions that demonstrated the greatest contribution to removal efficiency.

However, it was equally important to evaluate the model's performance when considering only controllable design parameters. While operational conditions like temperature and *Schmutzdecke* age are critical, they cannot always be directly controlled during filter operation. By isolating the analysis to design parameters that can be actively managed, such as FR, T.B.H., and D_{10}, more actionable insights can be derived to optimize filter design and operational strategies. Therefore, the most important controllable design parameters will also be discussed separately, namely the U.C., FR, and whether the experiment was conducted at a laboratory, pilot, or full scale.

SQ1: Which design parameters and operational conditions, both controllable and uncontrollable, are most influential for predicting bacterial (E.Coli, Coliform) removal efficiency in SSFs?

The XGBoost model revealed the following design parameters and operational conditions as the most influential for bacterial removal (Figure 4.4):

1. **D_{10} (D10_mm):** Smaller grain sizes enhance mechanical filtration by effectively retaining bacterial particles, a finding consistent with previous literature [20]. However, excessively fine grains can lead to clogging, reducing long-term filter performance. The XGBoost model reinforced this observation: dynamic slider adjustments across the range of 0.10 mm to 0.70 mm revealed that an optimal grain size of approximately 0.33 mm achieved the highest removal efficiencies.

2. **Temperature (T_high °C):** Elevated temperatures enhance biological activity within the *Schmutzdecke* and filter medium, promoting more efficient bacterial degradation and adsorption, a finding well-supported by existing literature [4][63][76][77]. However, insights from the XGBoost model revealed a nuanced relationship: while moderate temperatures improve removal efficiency, temperatures above 25°C resulted in diminishing returns and even a slight decline in bacterial removal performance.

3. **HRT (HRT_h):** Literature suggests that extended hydraulic retention times allows water to remain in contact with biofilm layers for longer periods, enhancing bacterial removal efficiency [67]. Optimal retention time strikes a balance between contaminant removal efficiency and hydraulic performance. Furthermore, HRTs ranging from 2.5 to 12.5 hours were recommended by literature as was stated previously [1]. However, insights from the XGBoost model reveal a more complex interaction: both low and high HRTs can achieve high removal efficiencies, but their effectiveness is strongly influenced by other operational conditions such as the age of the *Schmutzdecke*. For instance, with a newly formed Schmutzdecke—where the biofilm layer is still underdeveloped—low and high HRTs yield comparable results. Conversely, when the *Schmutzdecke* is well-established (approximately 1200 days old), higher HRTs become less effective in enhancing bacterial removal. This indicates that the interaction between HRT and biofilm maturity plays a crucial role in determining overall filtration performance.

4. **T.B.H. (TBH_m):** According to the literature, taller filter beds provide increased filtration depth and prolonged contact time, enhancing bacterial retention. However, for bacterial and viral contaminants, efficiency gains plateau beyond a certain height, and further increases in height no longer significantly improve removal efficiency [63][70][71][72]. The XGBoost model demonstrated that both low and high total bed heights yielded high removal efficiencies for bacteria. This finding does not contradict the literature, as removal efficiencies become consistently high from approximately 0.30 m onwards. Beyond this initial threshold, further increases in bed height do not result in substantial improvements in bacterial removal efficiency.

5. **Age of *Schmutzdecke* (Schm.A_d):** The literature states that a mature *Schmutzdecke* significantly enhances bacterial removal efficiency through key biological processes such as predation, metabolic breakdown, and adsorption mechanisms [1][53][74][75]. However, it also emphasizes that an older *Schmutzdecke* does not necessarily guarantee improved removal performance. Overly mature biofilm layers can lead to clogging, reduced water flow, and ultimately impaired filter efficiency. This observation is further validated by the XGBoost model. By dynamically adjusting the *Schmutzdecke* age slider, it becomes clear that while a moderately mature biofilm supports optimal bacterial removal, excessively aged biofilms exhibit diminishing returns and, in some cases, reduced filtration efficiency, most likely due to clogging and increased hydraulic resistance. This underscores the importance of timely maintenance and effective biofilm management to sustain peak filter performance.

SQ2: Which controllable design parameters are most influential for predicting bacterial removal efficiency in SSFs?

Focusing on only the controllable parameters from figure 4.15, the XGBoost model suggests that the **FR** and the **U.C.** are also important parameters for predicting removal of bacteria.

In literature, it is stated that a lower FR generally allows for a longer contact time between the water and the *Schmutzdecke* and filter media, facilitating more effective bacterial adsorption, predation, and biological breakdown [55][58][59][60].. At higher FRs, the reduced contact time can hinder these processes, potentially allowing bacteria to pass through the filter before sufficient retention occurs. However, excessively low FRs can also

cause operational challenges, such as clogging and stagnation, reducing overall efficiency. The XGBoost model disproves this, showcasing that both high and low FRs may lead to high percentages of bacteria removal.

With regards to the U.C.; literature suggests that the U.C. should generally be below 2 for optimal performance of SSFs [20]. The XGBoost model supports this, showing that the highest removal efficiencies are achieved when the U.C. falls within the range of 1.4 to 2. Within this range, the grain sizes are most likely relatively uniform, minimizing the risk of preferential flow paths and ensuring effective bacterial retention. A lower U.C. (closer to 1.4) results in consistent pore sizes, enhancing filtration uniformity, while a higher U.C. (approaching 2) still maintains adequate hydraulic performance. However, when the U.C. exceeds 2, the increasing variation in grain size can cause uneven pore distribution, leading to clogging in smaller pores and short-circuiting in larger ones, ultimately reducing filtration efficiency.

SQ3: Which design parameters and operational conditions, both controllable and uncontrollable, are most influential for predicting virus (Enterovirus, Adenovirus, Bacteriophage) removal efficiency in SSFs?

For the prediction of virus removal, the XGBoost model identified the following parameters and operational conditions as the most influential (Figure 4.8):

1. **Temperature (T_high °C):** The XGBoost model results indicate that virus removal remains effective across a broader temperature range compared to bacterial removal, despite both being influenced by the *Schmutzdecke*. This difference can be explained by the primary mechanisms driving their removal. Bacterial removal heavily relies on biological processes within the *Schmutzdecke*, such as predation, metabolic breakdown, and enzymatic activity [4][63][76][77]. These biological mechanisms are highly temperature-dependent, causing bacterial removal efficiency to peak sharply within an optimal temperature range. In contrast, virus removal is primarily governed by physico-chemical mechanisms, including adsorption onto biofilm surfaces, electrostatic interactions, and physical trapping within the *Schmutzdecke*. These processes are less sensitive to temperature variations, allowing virus removal efficiency to remain relatively stable across a wider temperature spectrum. While biological activity in the *Schmutzdecke* also contributes to virus removal, it is not the dominant factor in determining its efficiency.

2. **FR (FR_m/h):** According to the literature, lower FR generally improves virus retention by allowing more time for adsorption and physical entrapment processes to occur [55][58][59][60]. Conversely, higher FR reduces virus removal efficiency by shortening retention time and increasing the likelihood of viral particles passing through the filter without adequate contact with the biofilm or filter media. The XGBoost model supports these findings but also reveals a threshold effect: while extremely high FR diminishes removal efficiency, an excessively low FR can also result in poor virus removal rates.

3. **Interaction Between FR and T.B.H. (FR × TBH):** While HRT is mathematically derived from FR and TBH, the interaction term (FR × TBH) in the XGBoost model captures additional, context-specific insights into how these parameters jointly affect virus removal efficiency. Unlike the theoretical HRT, which assumes ideal flow conditions and linear relationships, the interaction term may reflect real-world complexities such as uneven flow distribution, channeling, and localized clogging within the *Schmutzdecke*. It accounts for non-linear behavior, showing how specific combinations of FR and T.B.H. can produce synergistic or threshold effects on virus removal. However, interpreting interaction terms like FR × T.B.H. is inherently more challenging because their effects are not isolated but depend on the values of both contributing parameters. For example, a high FR might enhance or diminish virus removal depending on the corresponding T.B.H. value, and vice versa. This interdependence creates a dynamic relationship that cannot be fully understood by examining either parameter in isolation.

SQ4: Which controllable design parameters are most influential for predicting virus removal efficiency in SSFs?

Focusing on the controllable parameters from figure 4.16, the XGBoost model reveals that, similar to bacterial removal, **the U.C.**, **FR**, and $\mathbf{D}_{10}$ emerge as key parameters for predicting virus removal efficiency. Additionally, whether the experiment was conducted on a **lab-, pilot-, or full-scale setup** also appears to play a significant role. However, the order of importance differs between bacterial and viral removal, suggesting nuanced differences in how these parameters interact with each contaminant type.

For bacterial removal, D_{10} seems to carry more weight compared to virus removal. This may be explained by the larger size of bacterial cells, which could make physical straining through the filter media a more dominant

removal mechanism. In contrast, viruses, being much smaller, rely more heavily on adsorption mechanisms and interactions with the biofilm in the *Schmutzdecke*.

The HRT remains the most critical design parameter for the removal of both bacteria and viruses. This is consistent with the idea that prolonged contact time between water and the filtration media enhances the opportunity for adsorption, biological breakdown, and predation to occur [67]..

Interestingly, FR appears to have a greater influence on virus removal compared to D_{10}. This could be because virus removal efficiency is more sensitive to the rate of water flow through the filter, as higher flow rates can reduce the likelihood of viral particles being adsorbed onto biofilm surfaces. In contrast, for bacteria, the physical straining effect associated with D_{10} contributes more directly to removal efficiency. Additionally, the U.C. seems to play a less critical role in virus removal than it does for bacterial removal. This aligns with the fact that uniform pore size distribution primarily influences physical filtration and hydraulic flow uniformity, which are more relevant for bacteria due to their larger size. Viruses, on the other hand, rely on physico-chemical adsorption processes that are less directly affected by pore size uniformity.

Finally, the observation that experimental scale (lab, pilot, or full scale) impacts removal efficiency suggests that operational conditions, such as FRs, biofilm development, and flow uniformity, may differ significantly across these scales. These differences can influence the effectiveness of key parameters like HRT and FR, potentially explaining some of the variability observed in the model's predictions.

5.1.3 Answering the Main Research Question

Through analysis of 2D heatmaps, multidimensional visualizations, and by use of coding techniques, key findings were derived regarding the design parameters and operational conditions that determine the removal efficiency of SSFs for bacteria and viruses. The key findings were represented in the previous Chapter. Using the key findings, new design paradigms can be formed. The design paradigms for bacteria and viruses can be seen in tables 5.1 and 5.2, respectively.

TABLE 5.1: Optimal Design Paradigms for Bacterial Removal in SSFs

Parameter (Combinations)	**Design Paradigm 1**	**Design Paradigm 2**
Temperature	Maintain filter operation strictly within 15–20 °C to ensure optimal bacterial removal efficiency.	If optimal temperatures cannot be maintained, operate within a range of 10–20 °C for acceptable bacterial removal efficiency.
D_{10}	Implement a D_{10} of 0.30–0.35 mm to achieve peak bacterial removal efficiency.	If a D_{10} of 0.30–0.35 mm is not feasible, implement a D_{10} between 0.45–0.70 mm under stable temperature conditions.
HRT	Maintain an HRT between 4–6 hours, particularly within the temperature range of 15–20 °C.	At higher temperatures (>20 °C), shorter HRTs may provide improved bacterial removal efficiency.
Schmutzdecke Age	Operate with *Schmutzdecke* layers aged between 80–300 days to ensure consistent bacterial removal.	For older *Schmutzdecke* layers (>300 days), adopt grain sizes between 0.35–0.45 mm to maintain efficiency.
T.B.H.	Utilize a T.B.H. between 1.0–1.6 m, paired with an HRT of 4–6 hours, for maximum efficiency.	If a T.B.H. of 1.0–1.6 m is not achievable, operate with lower T.B.H. values (0.5–0.7 m) at moderate temperatures (16–19 °C).
FR	Maintain an FR between 0.05–0.16 m/h, paired with an effective grain size of 0.30–0.35 mm.	For lower ranges of FRs, maintain an FR below 0.12 m/h to ensure bacterial removal remains consistent regardless of HRT.
U.C.	Ensure a U.C. between 1.4–2.0. At higher temperatures (>20 °C), ensure U.C. remains within a narrower range of 1.62–1.9.	If a U.C. of 1.4–2.0 cannot be maintained, higher U.C. values can still yield efficiency when paired with an FR of 0.05–0.12 m/h.

TABLE 5.2: Optimal Design Paradigms for Virus Removal in SSFs

Parameter (Combinations)	**Design Paradigm 1**	**Design Paradigm 2**
Temperature	Maintain temperatures between 10–20 ℃ for optimal virus removal efficiency.	Outside 10–20 ℃, optimize FR (0.10–0.30 m/h) and T.B.H. (0.4–0.6 m).
Temperature + FR	At moderate temperatures (16–18 ℃), pair with low FR values (0.10–0.30 m/h) for virus removal.	At FR >0.40 m/h, stabilize temperature (16–18 ℃).
Temperature + T.B.H.	At T.B.H. >1.2 m, lower temperatures (14 ℃) can be maintained.	-
FR	Maintain an FR between 0.12–0.30 m/h with T.B.H. (0.4–0.5 m).	At higher FR (>0.30 m/h), T.B.H. should remain below 1.2 m.
FR + D_{10}	Use FR (0.05–0.11 m/h) with D_{10} (0.30–0.35 mm).	For D_{10} (>0.35 mm), use lower FR (<0.3 m/h).
FR + HRT	Maintain HRT (4–8 hours) with FR below 0.3 m/h.	At HRT >8 hours, reduce FR (<0.2 m/h).
FR + U.C.	Ensure U.C. (1.4–2.6) with FR (0.05–0.11 m/h).	At U.C. >2.6, FR should be below <0.12 m/h.
D_{10} + HRT	Pair D_{10} (0.30–0.35 mm) with HRT (4–8 hours).	At shorter HRT (<4 hours), larger D_{10} (>0.35 mm) may still work.
Temperature + FR	At fluctuating temperatures, adjust FR (>0.40 m/h).	-
Temperature + HRT	At elevated temperatures (>18 ℃), reduce HRT (<4 hours).	-
U.C. + D_{10}	At U.C. >2.6, maintain FR (<0.12 m/h).	-
Temperature + U.C.	At temperatures >20 ℃, use narrow U.C. (1.62–1.9).	-

5.2 Robustness and Reliability of the Predictive Models

Ensuring the robustness and reliability of predictive models is essential for translating data-driven insights into actionable design improvements for SSFs. This subsection evaluates the performance of the developed XGBoost models for predicting bacterial and viral removal efficiencies, focusing on key performance metrics such as the R^2 score from cross-validation and the test set, along with the MSE on the test set. These metrics are used to assess whether the models generalize well to unseen data and if their predictions can be considered trustworthy.

5.2.1 Cross-Validation Performance

Cross-validation is a crucial step in evaluating model robustness, ensuring consistent performance across different subsets of data and mitigating overfitting risks. In other words, it assesses performance on unseen subsets of training data rather than completely new data.

For **bacterial removal efficiency (DEC_Bact_1)**, the XGBoost model achieved an **R^2 score of 0.653** from cross-validation when using both design parameters and operational conditions. This result indicates that the model can explain approximately **65.3%** of the variance in bacterial removal efficiency using the full set of parameters. Interestingly, when the model was trained using only **controllable design parameters**, a slightly higher **R^2 score of 0.655** was observed. This marginal improvement suggests that uncontrollable operational conditions might introduce noise or variability into the bacterial removal efficiency predictions, making the model slightly more effective when limited to design parameters alone.

For **viral removal efficiency (DEC_Vir)**, the XGBoost model demonstrated superior predictive power, achieving an **R^2 score of 0.807** from cross-validation when using both design parameters and operational conditions. This means the model successfully captures over **80%** of the variance in viral removal efficiency, highlighting strong predictive accuracy. When restricted to **controllable design parameters**, the model achieved an **R^2 score of 0.750**, slightly lower but still robust. The smaller drop in performance compared to the bacterial model suggests that viral removal efficiency is less sensitive to uncontrollable operational conditions and that design parameters play a more dominant role in determining efficiency outcomes.

These results collectively indicate that both bacterial and viral models can effectively capture significant relationships between **design parameters**, **operational conditions**, and **removal efficiencies**. However, the higher **R^2 score** for viral removal efficiency suggests a better fit and stronger predictive power for virus-related data. This could imply that viral removal processes are inherently more predictable under the given parameter set compared to bacterial removal, which may exhibit higher variability across the dataset. Moreover, the slight variance in bacterial model performance could point to more complex or nonlinear relationships in bacterial removal mechanisms, potentially influenced by subtle interactions between operational conditions and design parameters.

5.2.2 Test Set Performance

The evaluation on the test dataset provides insights into how well the models predict completely unseen data, offering an assessment of their final generalization performance, as shown in table 5.3 below for both models used in this study:

TABLE 5.3: Comparison of Model Performance for Bacterial and Viral Removal

Metric	Bacteria (Design + Operational)	Bacteria (Controllable)	Viruses (Design + Operational)	Viruses (Controllable)
R^2	0.981	0.813	0.981	0.857
MSE	0.0379	0.161	0.0069	0.053

Both models demonstrate strong performance, with high R^2 scores indicating their ability to explain a significant portion of the variance in the target variables, and low MSE values highlighting minimal prediction errors. The bacterial removal model captures key relationships between design parameters, operational conditions, and removal efficiency effectively, while the viral removal model exhibits slightly higher predictive power and consistency. Notably, both models maintain robust performance even when using only controllable design parameters.

5.2.3 Bias-Variance Tradeoff

The difference between cross-validation and test set scores offers valuable insights into the bias-variance trade off of the models. In machine learning, the bias-variance trade off describes the balance between bias and variance [84]. Bias refers to errors introduced when a model makes overly simplistic assumptions, causing it to underfit the data and fail to capture the underlying patterns. Variance refers to errors caused by the model being too sensitive to small fluctuations in the training data, leading to overfitting and poor performance on unseen data. Ideally, a model should strike a balance between these two, generalizing well to new data while not being overly influenced by training noise.

The bacterial model exhibits a noticeable difference between the CV R^2 score and the test R^2 score. This gap suggests that while the model performs well on unseen test dataset, it might still have overfitted slightly to the training data during cross-validation. The larger difference indicates that the model may have captured dataset-specific noise or variability rather than general patterns. Despite this, the high test score suggests strong predictive ability on unseen data, though it raises a caution about the model's stability across different datasets.

In contrast, the viral model demonstrates more consistent performance, with the CV R^2 score and the test R^2 score being closely aligned. This smaller difference indicates a well-balanced trade off between bias and variance, where the model avoids both oversimplification and excessive sensitivity to training data. Such alignment suggests that the viral model generalizes effectively across unseen datasets and performs reliably under varying conditions.

Both models, however, achieve performance scores within acceptable thresholds for predictive reliability.

5.3 Limitations

While this study provides valuable insights into the optimization of SSFs using data-driven techniques and machine learning models, several limitations must be acknowledged to contextualize the findings and highlight areas for improvement in future research.

5.3.1 Data and Model Limitations

Data Limitations

One key limitation lies in the dataset used in this study, which was compiled by combining data from experiments conducted across multiple studies, each with varying experimental conditions, measurement techniques, and reporting standards. This heterogeneity introduced inconsistencies that could not always be fully addressed during data pre-processing. Additionally, the geographic and environmental diversity of the dataset might have introduced variability or biases that were not fully accounted for.

Another limitation arises from the uneven distribution of data points across different parameter ranges. Some parameter combinations were well represented, while others were sparsely covered in the dataset. This imbalance may have led to biases in the model, where predictions for underrepresented parameter ranges might be less reliable. As mentioned earlier, XGBoost has the capability to handle missing values (NaN) effectively during training. However, while XGBoost can manage these gaps, minimizing the number of missing values remains essential for achieving optimal model performance. Excessive missing data can still reduce the model's ability to capture underlying patterns accurately and may lead to biased predictions.

Model Limitations

The XGBoost model operates under certain assumptions and simplifications. For instance, it does not explicitly account for temporal dynamics or long-term filter behavior. Additionally, while the model effectively captures non-linear interactions, it may still overlook subtle physical or biological mechanisms occurring within the SSF. Moreover, the model is purely data-driven and not based on physical formulas, which can limit its ability to accurately represent certain real-world filtration processes and mechanisms.

Lastly, the model was developed using data from 135 studies, and its performance was validated using internal test subsets derived from this dataset. However, the model could not be tested on external datasets beyond these 135 studies due to the lack of additional experimental data on SSFs. This limitation restricts the generalizability of the model's predictions and raises questions about its performance when applied to unseen or real-world scenarios.

5.3.2 Experimental Limitations

Another limitation arises from the experimental nature of the data. Most of the data used in this study were derived from controlled laboratory or pilot-scale experiments. These conditions may not fully replicate the variability and unpredictability of real-world SSF operations, where environmental factors, maintenance practices, and influent quality fluctuate significantly. Data from full-scale operational SSFs would likely have increased the robustness of the findings. While some experiments in the dataset were conducted at full scale, they were too limited and scattered to provide comprehensive insights. Additionally, this study primarily focused on static snapshots of SSF performance rather than analyzing performance over extended operational periods. Temporal dynamics, such as the maturation and degradation of the *Schmutzdecke*, were not explicitly modeled, potentially limiting the predictive capacity for long-term scenarios. Explicit inclusion of seasonal or temporal trends could improve the predictive accuracy of the model.

Chapter 6

Conclusion and Future Work

This chapter presents the concluding remarks of the study, reflecting on its contributions to both research and practice. It highlights how the developed model enhances understanding of SSF performance and its potential for guiding design optimizations. In addition, this chapter outlines directions for future work to build on the insights gained and further advance the design and modeling of SSF performance.

6.1 Concluding Remarks

This research demonstrated the value of data driven approaches, specifically using XGBoost modeling, to advance the understanding of the design parameters and operational conditions that play a part in the removal of bacteria and viruses in SSFs. The XGBoost model developed in this research not only provided decent predictions but also served as a powerful analytical tools for uncovering patterns and interactions that were difficult to detect with traditional methods. Through feature importance analysis and SHAP values, the models revealed which design parameters and operational conditions most significantly affected removal efficiencies and how these factors interacted with one another. This allows the simulation of different design scenarios and evaluation of their impact, supporting more informed design decisions. Additionally, the model provided a basis for understanding how changes in certain parameters, such as the FR, HRT, and D_{10}, influenced removal efficiencies, enabling targeted adjustments for performance improvement.

Beyond its predictive capabilities, the model can act as a guide for future research by identifying key parameters that merit further investigation. As mentioned previously, it highlighted which design parameters had the greatest influence on removal efficiencies, directing attention to areas where experimental studies could yield the most valuable insights. Additionally, by simulating parameter combinations that were not well represented in the experimental data, the model expanded the scope of exploration, providing hypotheses that could be tested in future experiments.

Lastly, the model's predictions also contributed to the formulation of design paradigms showcased in the previous chapter, offering a pathway towards more efficient and cost-effective SSF configurations. By leveraging model driven insights, future designs can be tailored to maximize contaminant removal while optimizing operational efficiency. This approach not only supports sustainable water treatment practices but also reduces resource consumption.

In conclusion, this study emphasizes the potential of machine learning to revolutionize the design and operation of SSFs, providing a foundation for future advancements in sustainable water treatment technologies.

6.2 Future Work Suggestions

Future research on SSFs should build upon the insights and limitations identified in this study. First, larger and more complete datasets should be collected, in which diverse geographic, environmental, and operational conditions are also noted. This will help reduce biases and ensure that underrepresented design parameters, such as U.C. and the age of *Schmutzdecke*, are better represented.

In addition, future studies should focus more on parameters that can be actively adjusted in practice. Parameters such as temperature or the age of the *Schmutzdecke*, while influential, cannot be directly controlled during SSF operation as mentioned prior, limiting their practical relevance. Future models should prioritize focusing on

adjustable parameters to improve their applicability in real-world systems, while also trying to implement the effect of non-adjustable factors into the model such as changing temperatures. Incorporating temporal analysis into machine learning models is essential, not only to understand effects of temperature but also other parameters as well, such as the development of the *Schmutzdecke*. By analyzing the maturation and degradation of the Schmutzdecke over time, as well as seasonal variations, researchers can develop more robust models that account for long-term performance dynamics.

Additionally, hybrid models that integrate data-driven approaches with physical simulation models, using equations to describe the filtration mechanics of SSFs, could help bridge the gap between statistical accuracy and physical interpretability.

Validation on external datasets remains a critical priority as well. Conducting new experiments or leveraging data from full-scale operational SSFs will help ensure that models perform reliably outside the original dataset of 135 studies. This step is also essential for confirming the model's performance on independent datasets, providing further assurance of its robustness and generalizability.

Lastly, the previous chapter highlighted that a key limitation of the model is its restricted generalizability, as its recommendations are confined to the parameter ranges provided in the dataset of 135 studies. Future work could overcome this limitation by incorporating additional data from studies that explore a broader range of design and operational conditions. While such data is currently unavailable, this gap could be addressed through pilot experiments designed to test a wider range of parameter values. Expanding the dataset in this way would enhance the model's ability to account for extreme or uncommon scenarios, ultimately improving its applicability and robustness.

By addressing these key areas, future research can contribute to more effective, reliable, and sustainable SSF designs.

Appendix A

Comprehensive Dataset of Analyzed Studies and Experimental Data

By clicking the link below, the dataset created and utilized in this MSc thesis research can be accessed. In addition, table A.1 provides detailed explanations of the abbreviations used in the data set.

Comprehensive Dataset of Analyzed Studies and Experimental Data.

TABLE A.1: Glossary of Slow Sand Filter Parameters

Parameter	Description
FR	Filtration Rate [m/h]
FM	Filter Medium (Sand/Gravel/Other medium)
D10	Effective Size [mm]
D10/D60	Uniformity Coefficient [-]
Grain Size Range	Range of particle sizes in the filter medium [mm]
n	Porosity
T.B.H.	Total Bed Height [m]
SA	Surface Area [m^2]
Capacity	Volume capacity of the filter system [l]
Type of Water	Surface Water, Seepage Water, Groundwater, etc.
Duration of Operation	Days the filter has been operational [d]
Schm.A.	Age of Schmutzdecke [d]
A.o.F.	Age of the filter in days [d]
S.F.	Scraping Frequency [y]
Con/Bat	Continuous/Batch operation
L/P/F	Lab, Pilot, or Full scale system
O/C	Open/Closed system
T	Temperature of the Water [℃]
SWL	Supernatant Water Layer [m]
R.E.	Removal Efficiencies (e.g., bacteria, protozoa, turbidity) [%]
FM_status	Status of the Filter Medium: Washed/Ripened/Fresh
Layers?	More than one layer: Yes or No
Type_L	Type of extra layer: gravel/sand/fine sand, etc.
Depth	Depth of the extra layer [m]
Year	Year of Research [y]
DOC	Dissolved Organic Carbon in influent water [mg C/l]
Turbidity	Turbidity in influent water [NTU]
pH	pH of influent water
conc. poll	Concentration of target pollutant
Hardness	Hardness of influent water [mg/l]
B.C.	Bottom Construction (Nozzles/Gravel support layer)
P.T.	Pre-treatment (conventional/extended)
H.R.T.	Hydraulic Retention Time [h]
DEC_bact_1	Decimal Elimination Capacity of Bacteria type 1 [logs]
DEC_bact_2	Decimal Elimination Capacity of Bacteria type 2 [logs]
DEC_Oo_GIa	Decimal Elimination Capacity of Giardia [logs]
DEC_Oo_Cry	Decimal Elimination Capacity of Cryptosporidium [logs]
DEC_bact_2	Decimal Elimination Capacity of Viruses [logs]

Appendix B

Calculating the Correlations Between Design Parameters and Removal Efficiencies of Bacteria and Viruses

This appendix provides the Python code used to generate the correlation matrix heatmap.

```
operational_parameters = [
    'FR [m/h]', 'D10 [mm]', 'D60/D10', 'GrainSize_low [mm]',
    'GrainSize_high [mm]', 'T_low [ C ]', 'T_high [ C ]', 'n [-]',
    'T.B.H. [m]', 'SA [m2]', 'Cap.[L]', 'Duration of Operation [d]',
    'Schm.A. [d]', 'A.O.F. [d]', 'pH', 'Hardness [mg/l]'
]
removal_efficiencies = ['DEC_bact_1', 'DEC_Vir']

# Ensure all relevant columns are numeric
for col in operational_parameters + removal_efficiencies:
    data[col] = pd.to_numeric(data[col], errors='coerce')

# Define a correlation function to handle NaN values
def correlation(x, y):
    valid_mask = ~x.isna() & ~y.isna()
    x, y = x[valid_mask], y[valid_mask]
    return x.corr(y)

# Initialize an empty DataFrame for the correlation matrix
correlation_matrix = pd.DataFrame(index=operational_parameters, columns=
    removal_efficiencies)

# Calculate correlations
for parameter in operational_parameters:
    for efficiency in removal_efficiencies:
        correlation_matrix.loc[parameter, efficiency] = correlation(data[parameter],
            data[efficiency])

# Convert the correlation matrix to numeric for plotting
correlation_matrix = correlation_matrix.astype(float)

# Plot heatmap
plt.figure(figsize=(14, 14))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Matrix: Operational Parameters vs. Removal Efficiencies')
plt.xlabel('Removal Efficiencies')
plt.ylabel('Operational Parameters')
plt.tight_layout()
plt.show()
```

LISTING B.1: Correlation Matrix Code for Operational Parameters and Removal Efficiencies

Appendix C

Development and Training of an XGBoost Model for Predicting Removal Efficiencies

This appendix provides the Python code used to train the XGBoost model, tune hyperparameters, and evaluate performance metrics for bacterial removal efficiency prediction.

```
import pandas as pd
import numpy as np
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split, GridSearchCV, StratifiedKFold
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder

# Rename the columns to remove special characters
data_bact = data.rename(columns=lambda x: x.replace('[', '').replace(']', '').replace('<', '').replace('>', '').replace(' ', '_'))

# Drop rows with missing target values
data_bact = data_bact.dropna(subset=['DEC_bact_1'])

data_bact['FR_TBH_interaction'] = data_bact['FR_m/h'] * data_bact['T.B.H._m']
data_bact['FR_squared'] = data_bact['FR_m/h'] ** 2
data_bact['D10_TBH_interaction'] = data_bact['D10_mm'] * data_bact['T.B.H._m']
data_bact['FR_D60_interaction'] = data_bact['FR_m/h'] * data_bact['D60/D10']

# Define the features and target variable
X = data_bact[['L/P/F', 'T_high_ C', 'FR_m/h', 'T.B.H._m', 'D10_mm', 'D60/D10',
               'FR_TBH_interaction', 'Schm.A._d', 'Duration_of_Operation_d', 'n_-', 'Type_of_Water', 'HRT_h']]
y = data_bact['DEC_bact_1']

# Create quantile-based bins for the target variable
y_binned = pd.qcut(y, q=10, labels=False)
print(f"Binned Target Distribution:\n{pd.value_counts(y_binned)}")

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, shuffle=True)

# Set up a parameter grid for hyperparameter tuning
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [6, 8, 10],
    'learning_rate': [0.01, 0.05, 0.1],
    'subsample': [0.6, 0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0],
    'reg_alpha': [0.1, 0.5, 1],
    'reg_lambda': [5, 10]
}

```

```
# Initialize the XGBoost Regressor
xgb = XGBRegressor(random_state=42)

# Set up StratifiedKFold using binned target
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

# Perform grid search with StratifiedKFold
grid_search = GridSearchCV(
    estimator=xgb,
    param_grid=param_grid,
    cv=skf.split(X, y_binned),
    scoring='r2',
    verbose=1,
    n_jobs=-1
)

# Fit grid search to the training data
grid_search.fit(X, y)
print(f"Best Parameters: {grid_search.best_params_}")
print(f"Best R^2 Score from CV: {grid_search.best_score_}")

# Train the model using the best parameters
best_xgb = grid_search.best_estimator_

# Predict on the test set
y_pred = best_xgb.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error on Test Set: {mse}")
print(f"R^2 Score on Test Set: {r2}")

# Feature Importance Plot
booster = best_xgb.get_booster()
importance = booster.get_fscore()
importance_df = pd.DataFrame(importance.items(), columns=['Feature', 'Importance'])
importance_df.sort_values(by='Importance', ascending=False, inplace=True)

plt.figure(figsize=(10, 8))
plt.barh(importance_df['Feature'], importance_df['Importance'])
plt.xlabel("Feature Importance")
plt.ylabel("Features")
plt.title("XGBoost Feature Importance")
plt.show()

# Visualize actual vs predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, alpha=0.7)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color='red',
    linestyle='--')
plt.xlabel("Actual DEC_bact_1")
plt.ylabel("Predicted DEC_bact_1")
plt.title("Actual vs Predicted DEC_bact_1")
plt.show()
```

LISTING C.1: XGBoost Model Training and Hyperparameter Tuning for DEC_bact_1 Prediction

Appendix D

Feature Selection for XGBoost Using Bayesian Information Criterion (BIC) and Model Evaluation

This appendix provides the Python code used to evaluate different subsets of features for bacterial removal efficiency prediction using Bayesian Information Criterion (BIC) and R^2 scores.

```
import itertools
from sklearn.metrics import mean_squared_error

# Function to calculate Bayesian Information Criterion (BIC)
def calculate_bic(n, mse, k):
    return n * np.log(mse) + k * np.log(n)

# List of all features
all_features = ['L/P/F', 'T_high_ C', 'FR_m/h', 'T.B.H._m', 'D10_mm', 'D60/D10',
                'FR_TBH_interaction', 'Schm.A._d', 'Duration_of_Operation_d']

# Store BIC and R^2 values for each feature subset
bic_values = []
r2_values = []
num_features_list = []
selected_feature_list = []

n_samples = X_train.shape[0]  # Number of training samples

# Iterate over all possible combinations of features
for num_features in range(1, len(all_features) + 1):
    for feature_subset in itertools.combinations(all_features, num_features):
        feature_subset = list(feature_subset)

        # Subset the training and testing data
        X_train_subset = X_train[feature_subset]
        X_test_subset = X_test[feature_subset]

        # Train the model using the best parameters
        model = XGBRegressor(**grid_search.best_params_, random_state=42)
        model.fit(X_train_subset, y_train)

        # Predict and calculate MSE
        y_pred = model.predict(X_test_subset)
        mse = mean_squared_error(y_test, y_pred)
        r2 = model.score(X_test_subset, y_test)

        # Calculate BIC
        bic = calculate_bic(n_samples, mse, num_features)

        # Store results
        bic_values.append(bic)
```

```
        r2_values.append(r2)
        num_features_list.append(num_features)
        selected_feature_list.append(feature_subset)

# Find the optimal number of features (minimum BIC)
optimal_index = np.argmin(bic_values)
optimal_num_features = num_features_list[optimal_index]
optimal_features = selected_feature_list[optimal_index]

print(f"Optimal Number of Features: {optimal_num_features}")
print(f"Features Selected: {optimal_features}")
print(f"Minimum BIC: {bic_values[optimal_index]:.4f}")
print(f"R^2 on Test Set: {r2_values[optimal_index]:.4f}")

# Plot BIC vs Number of Features
plt.figure(figsize=(10, 6))
plt.plot(num_features_list, bic_values, marker='o', linestyle='-', color='b', label='
    BIC')
plt.scatter(optimal_num_features, bic_values[optimal_index], color='red', s=100, label=
    'Optimal BIC')
plt.xlabel('Number of Features')
plt.ylabel('BIC')
plt.title('BIC vs. Number of Features')
plt.legend()
plt.grid()
plt.show()
```

LISTING D.1: Feature Subset Selection using BIC and R^2 Scores

Appendix E

Interactive Tool for Predicting Removal Efficiencies

This appendix provides the Python code used to create an interactive tool for predicting bacteria removal efficiency and visualizing the contributions of individual design parameters through SHAP values.

```
import shap
import numpy as np
import matplotlib.pyplot as plt
import ipywidgets as widgets
from IPython.display import display

def predict_bacteria_removal_with_visuals(model, scaler=None):
    """

    Parameters:
        model: Trained XGBoost model.
    """
    # Define sliders for each feature
    slider_T_high = widgets.FloatSlider(value=16.5, min=2.0, max=30.0, step=0.1,
        description='T_high ( C ):')
    slider_D10 = widgets.FloatSlider(value=0.270, min=0.1, max=0.7, step=0.01,
        description='D10 (mm):')
    slider_HRT = widgets.FloatSlider(value=2.25, min=0.1, max=70.0, step=0.1,
        description='HRT (h):')
    slider_Schm_A = widgets.FloatSlider(value=0, min=0, max=1200.0, step=0.1,
        description='Schm.A (d):')
    slider_TBH = widgets.FloatSlider(value=0.90, min=0.02, max=1.7, step=0.1,
        description='T.B.H (m):')

    # Output widgets to display prediction and visualizations
    output_prediction = widgets.Output()
    output_pie_chart = widgets.Output()
    output_bar_chart = widgets.Output()

    def update_visuals(change=None):
        # Get current values from sliders
        input_values = np.array([
            slider_T_high.value,
            slider_D10.value,
            slider_HRT.value,
            slider_Schm_A.value,
            slider_TBH.value
        ]).reshape(1, -1)

        # Scale the input if a scaler is provided
        if scaler:
            input_values = scaler.transform(input_values)

        # Predict using the trained model
        prediction = model.predict(input_values)[0]
```

```
41
42          # Calculate SHAP values
43          explainer = shap.TreeExplainer(model)
44          shap_values = explainer(input_values)
45
46          # Extract feature contributions
47          contributions = shap_values.values[0]
48          feature_names = ['T_high ( C )', 'D10 (mm)', 'HRT (h)', 'Schm.A (d)', 'T.B.H (m
                )']
49
50          # Update prediction output
51          with output_prediction:
52              output_prediction.clear_output(wait=True)
53              print(f"Predicted Bacteria Removal (DEC_bact_1): {prediction:.4f}")
54
55          # Update pie chart output
56          with output_pie_chart:
57              output_pie_chart.clear_output(wait=True)
58              plt.figure(figsize=(8, 6))
59
60              # Normalize contributions to sum to 1
61              normalized_contributions = np.abs(contributions) / np.sum(np.abs(
                    contributions))
62
63              plt.pie(
64                  normalized_contributions,
65                  labels=feature_names,
66                  autopct='%1.1f%%',
67                  startangle=140
68              )
69              plt.title("Local Explanation: Feature Contributions to Single Prediction (
                    Pie Chart)")
70              plt.show()
71
72          # Update bar chart output
73          with output_bar_chart:
74              output_bar_chart.clear_output(wait=True)
75              plt.figure(figsize=(8, 6))
76              plt.barh(feature_names, contributions, color=['red' if c < 0 else 'green'
                    for c in contributions])
77              plt.xlabel("SHAP Value")
78              plt.title("Local Explanation: XGBoost Feature Importance (Bar Chart)")
79              plt.axvline(0, color='black', linestyle='--')
80              plt.show()
81
82      # Attach the update function to slider changes
83      for slider in [slider_T_high, slider_D10, slider_HRT, slider_Schm_A, slider_TBH]:
84          slider.observe(update_visuals, names='value')
85
86      # Display the sliders and outputs
87      display(widgets.VBox([slider_T_high, slider_D10, slider_HRT, slider_Schm_A,
            slider_TBH]))
88      display(output_prediction)
89      display(output_pie_chart)
90      display(output_bar_chart)
91
92      # Trigger an initial update
93      update_visuals()
94
95  # Example usage
96  predict_bacteria_removal_with_visuals(best_xgb, scaler=None)
```

LISTING E.1: Interactive Tool for Predicting Bacteria Removal Efficiency with SHAP Visualizations

Appendix F

Heatmaps of Two-Parameter Interactions for Prediction of Bacteria Removal

This appendix presents the heatmaps generated from the interactive tool, showing the predicted bacterial removal efficiency across different combinations of key design parameters and/or operational conditions.

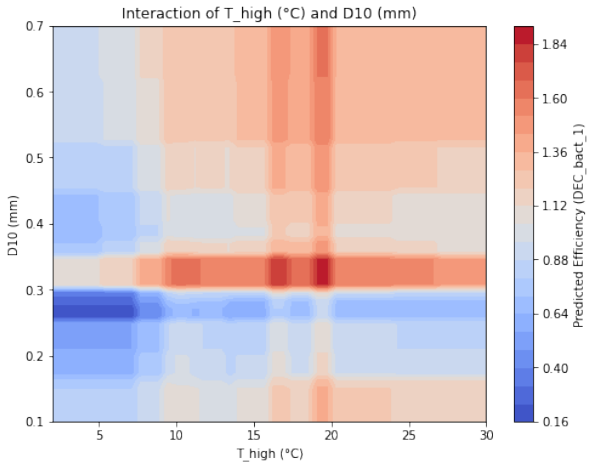

FIGURE F.1: Interaction of Temperature and Effective Size

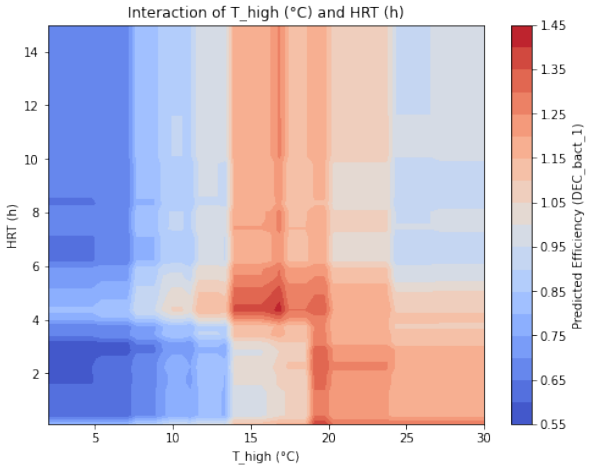

FIGURE F.2: Interaction of Hydraulic Retention Time and Temperature

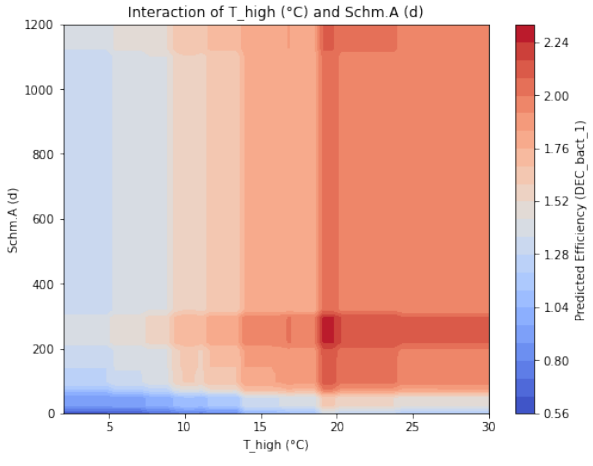

FIGURE F.3: Interaction of Temperature and Age of Schmutzdecke

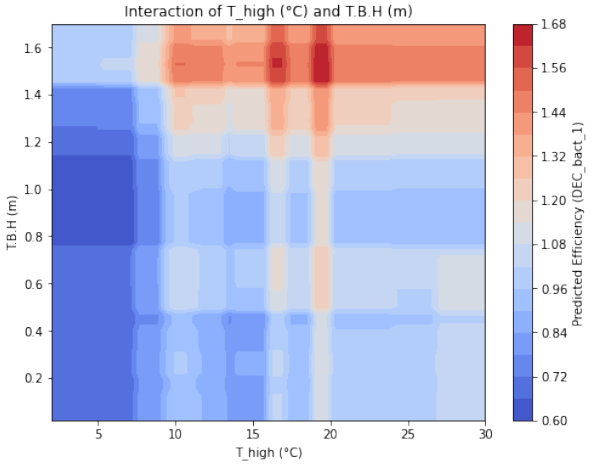

FIGURE F.4: Interaction of Total Bed Height and Temperature

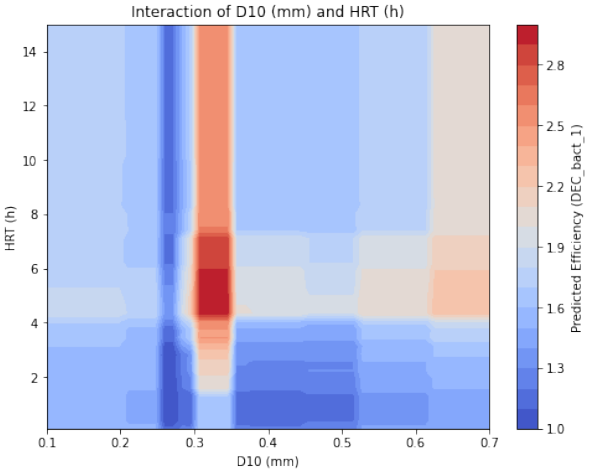

FIGURE F.5: Interaction of Effective Size and Hydraulic Retention Time

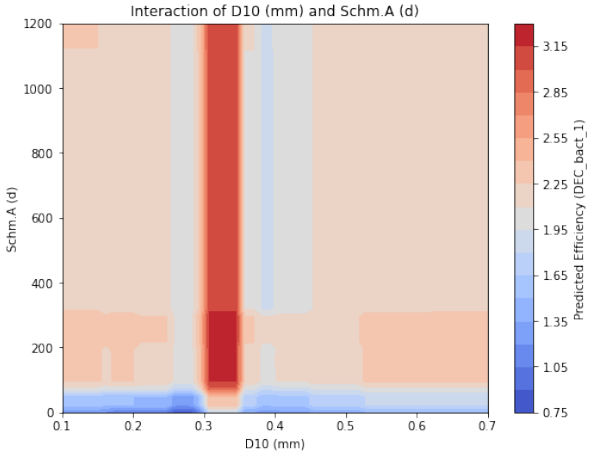

FIGURE F.6: Interaction of Effective Size and Age of Schmutzdecke

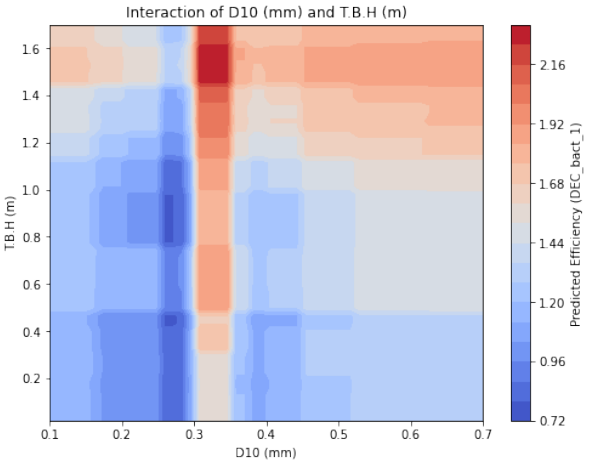

FIGURE F.7: Interaction of Effective Size and Total Bed Height

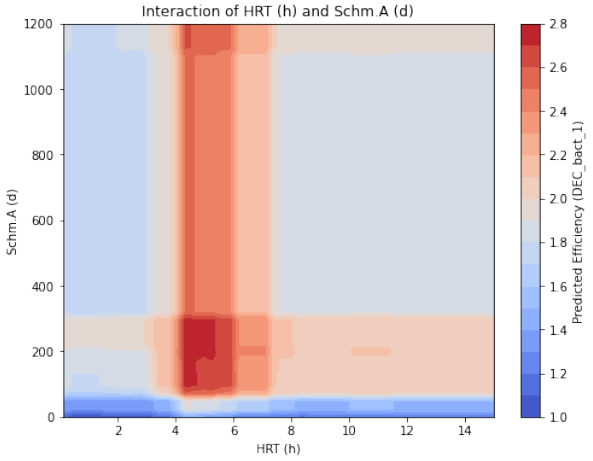

FIGURE F.8: Interaction of Age of Schmutzdecke and Hydraulic Retention Time

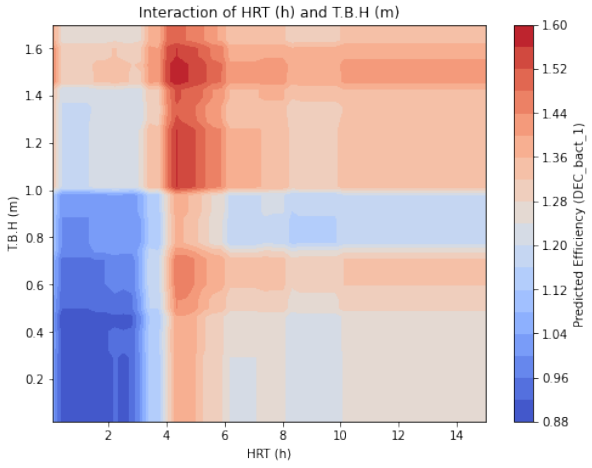

FIGURE F.9: Interaction of Hydraulic Retention Time and Total Bed Height

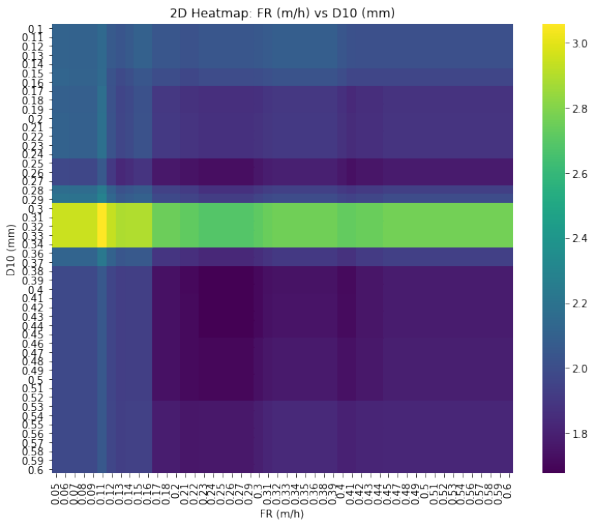

FIGURE F.10: Interaction of Filtration Rate and Effective Size

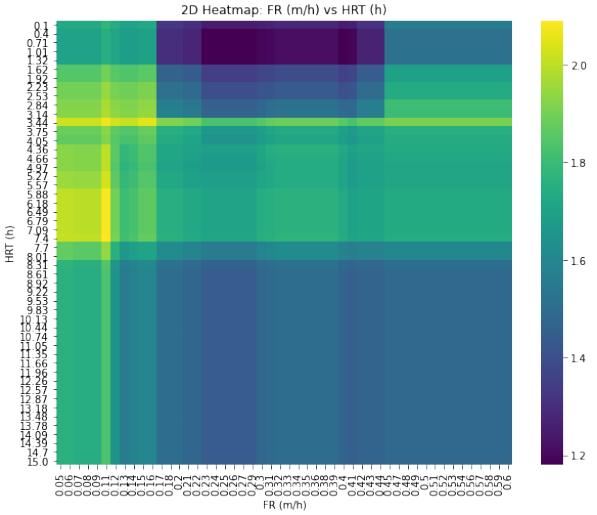

FIGURE F.11: Interaction of Filtration Rate and Hydraulic Retention Time

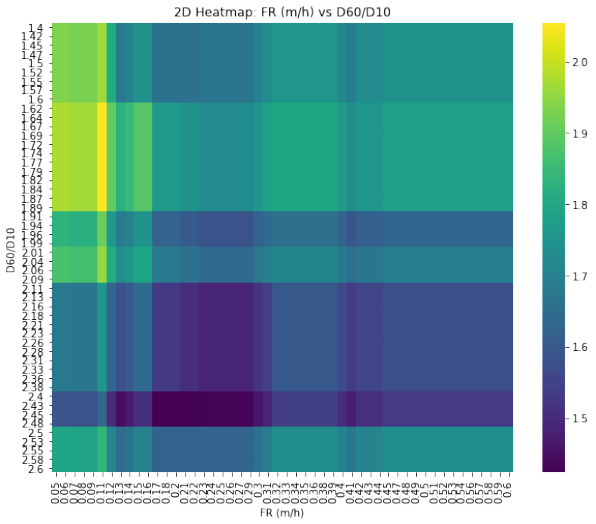

FIGURE F.12: Interaction of Filtration Rate and Uniformity Coefficient

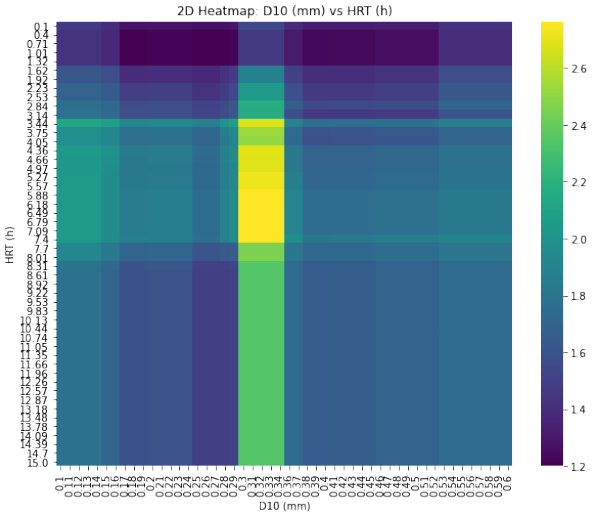

FIGURE F.13: Interaction of Effective Size and Hydraulic Retention Time

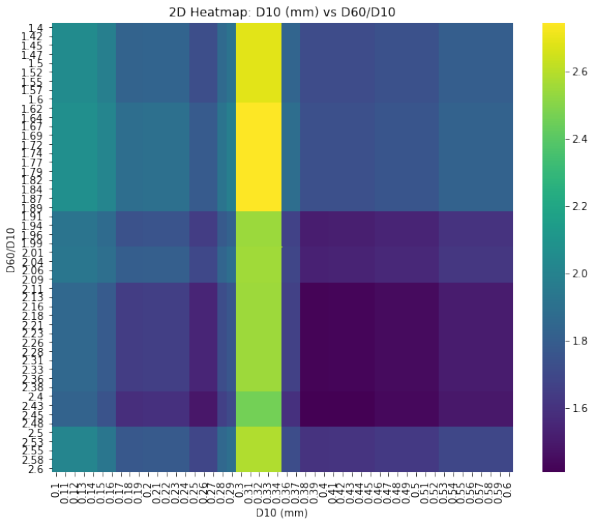

FIGURE F.14: Interaction of Effective Size and Uniformity Coefficient

Appendix G

Heatmaps of Two-Parameter Interactions for Prediction of Virus Removal

This appendix presents the heatmaps generated from the interactive tool, showing the predicted virus removal efficiency across different combinations of key design parameters and/or operational conditions.

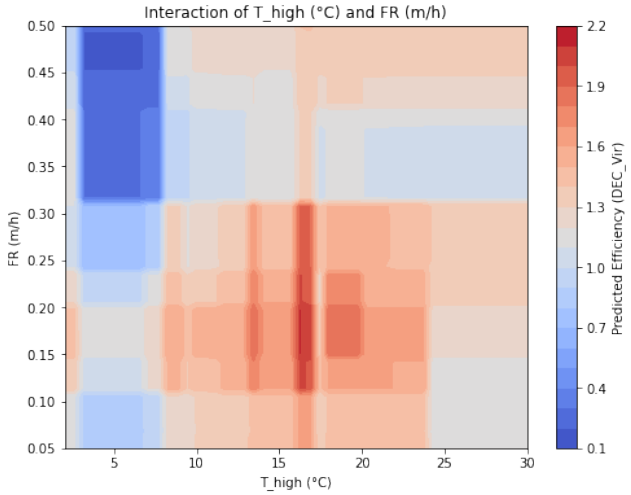

FIGURE G.1: Interaction of Filtration Rate and Temperature

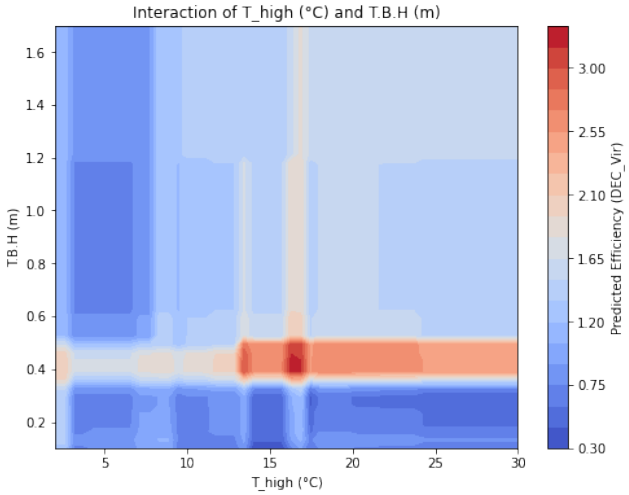

FIGURE G.2: Interaction of Temperature and Total Bed Height

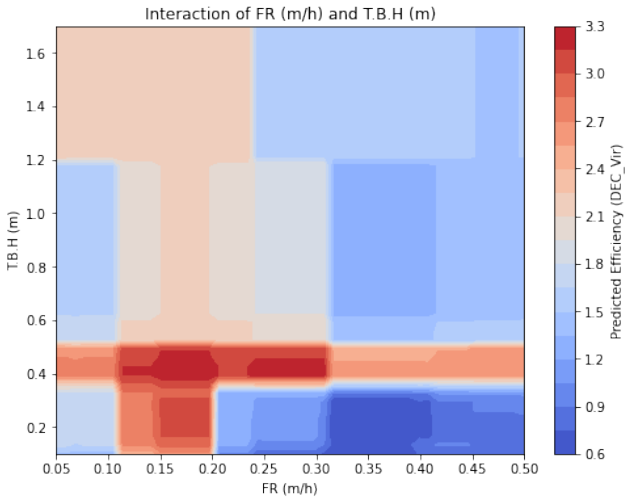

FIGURE G.3: Interaction of Filtration Rate and Total Bed Height

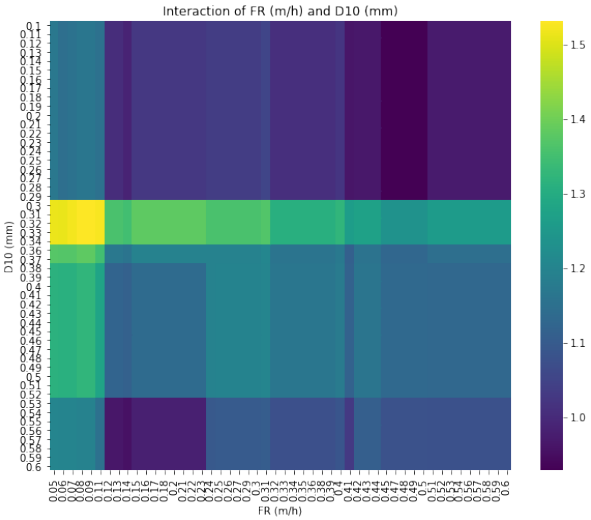

FIGURE G.4: Interaction of Filtration Rate and Effective Size on lab scale

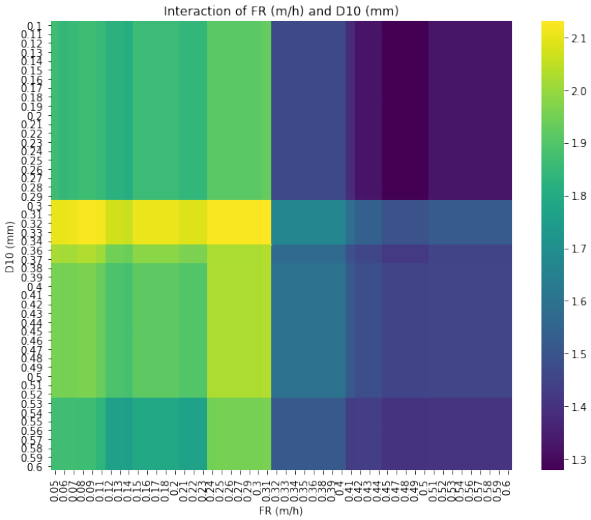

FIGURE G.5: Interaction of Filtration Rate and Effective Size on pilot/full scale

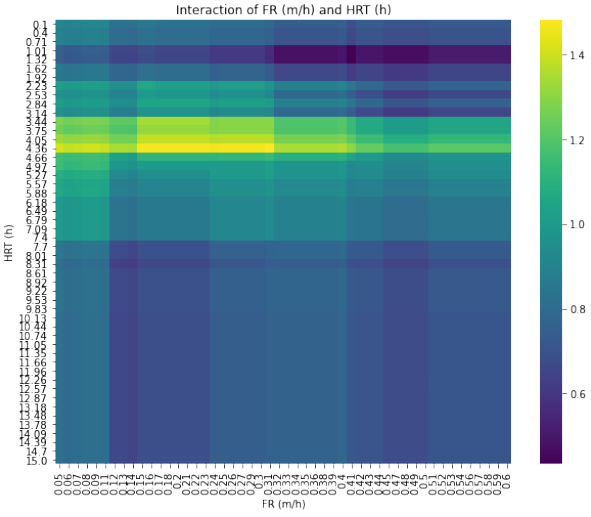

FIGURE G.6: Interaction of Filtration Rate and Hydraulic Retention Time on lab scale

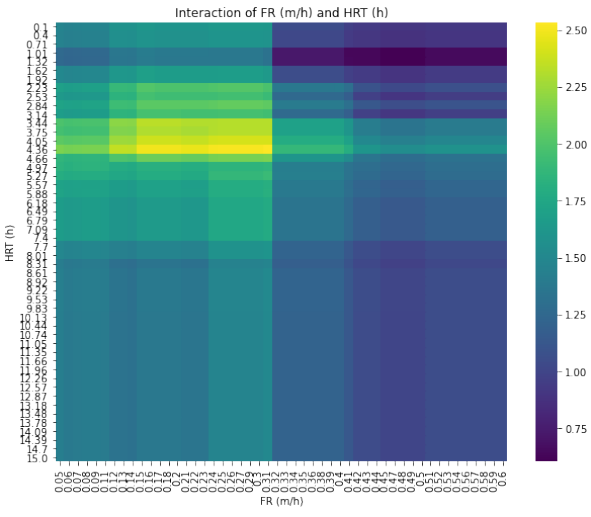

FIGURE G.7: Interaction of Filtration Rate and Hydraulic Retention Time on pilot/full scale

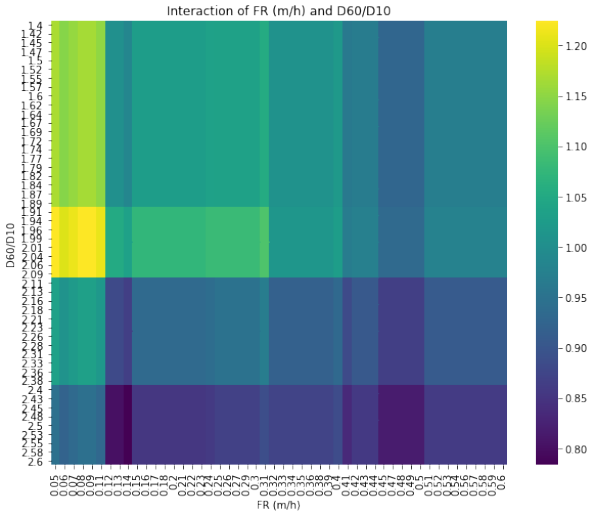

FIGURE G.8: Interaction of Filtration Rate and Uniformity Coefficient on lab scale

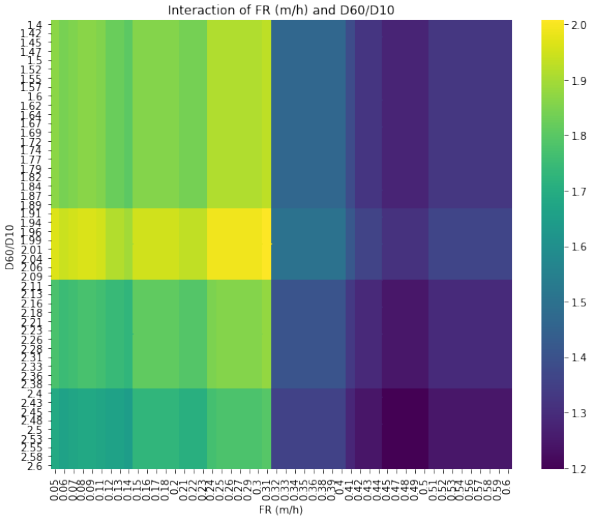

FIGURE G.9: Interaction of Filtration Rate and Uniformity Coefficient on pilot/full scale

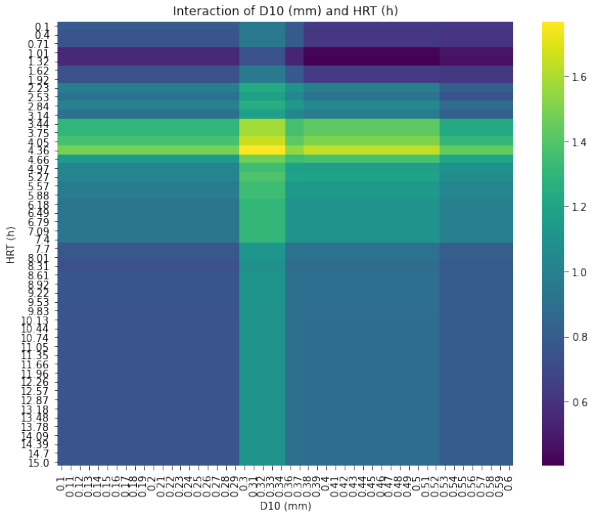

FIGURE G.10: Interaction of Effective Size and Hydraulic Retention Time on lab scale

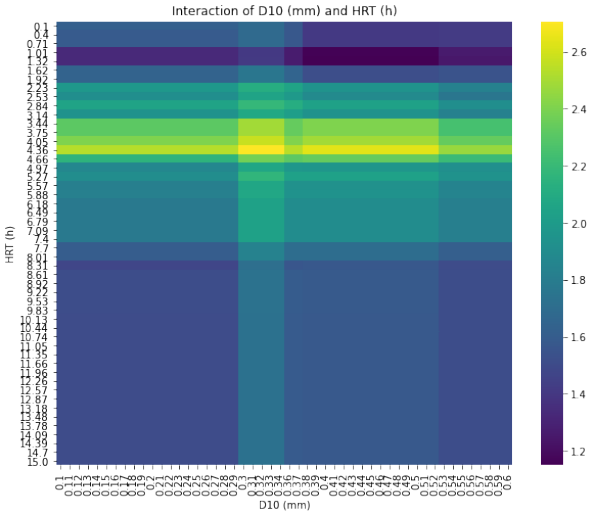

FIGURE G.11: Interaction of Effective Size and Hydraulic Retention Time on pilot/full scale

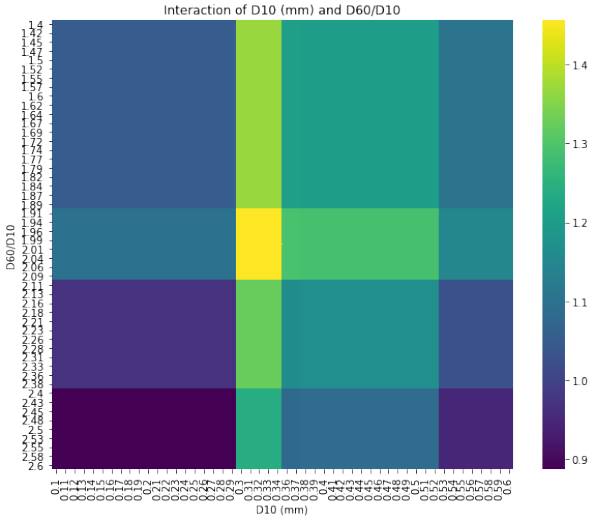

FIGURE G.12: Interaction of Effective Size and Uniformity Coefficient on lab scale

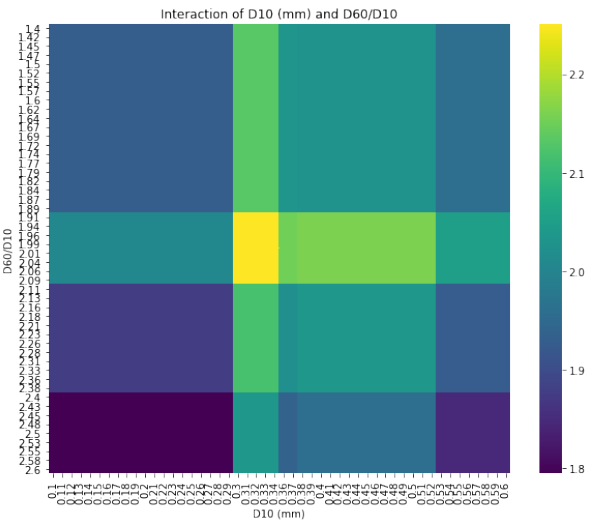

FIGURE G.13: Interaction of Effective Size and Uniformity Coefficient on pilot/full scale

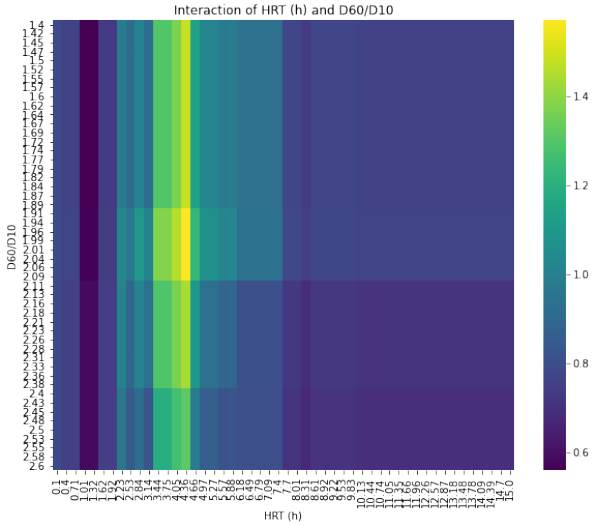

FIGURE G.14: Interaction of Hydraulic Retention Time and Uniformity Coefficient on lab scale

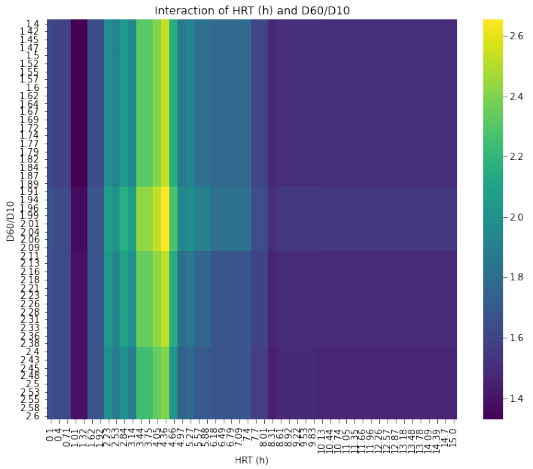

FIGURE G.15: Interaction of HRT and U.C. on pilot/full scale

Appendix H

Advanced Visualization of Multi-Parameter Interactions

This appendix showcases advanced visualizations representing three-dimensional interactions of key design parameters and operational conditions on their influence on bacterial and viral removal efficiencies. These plots help identify non-linear trends, parameter synergies, and regions of optimal performance.

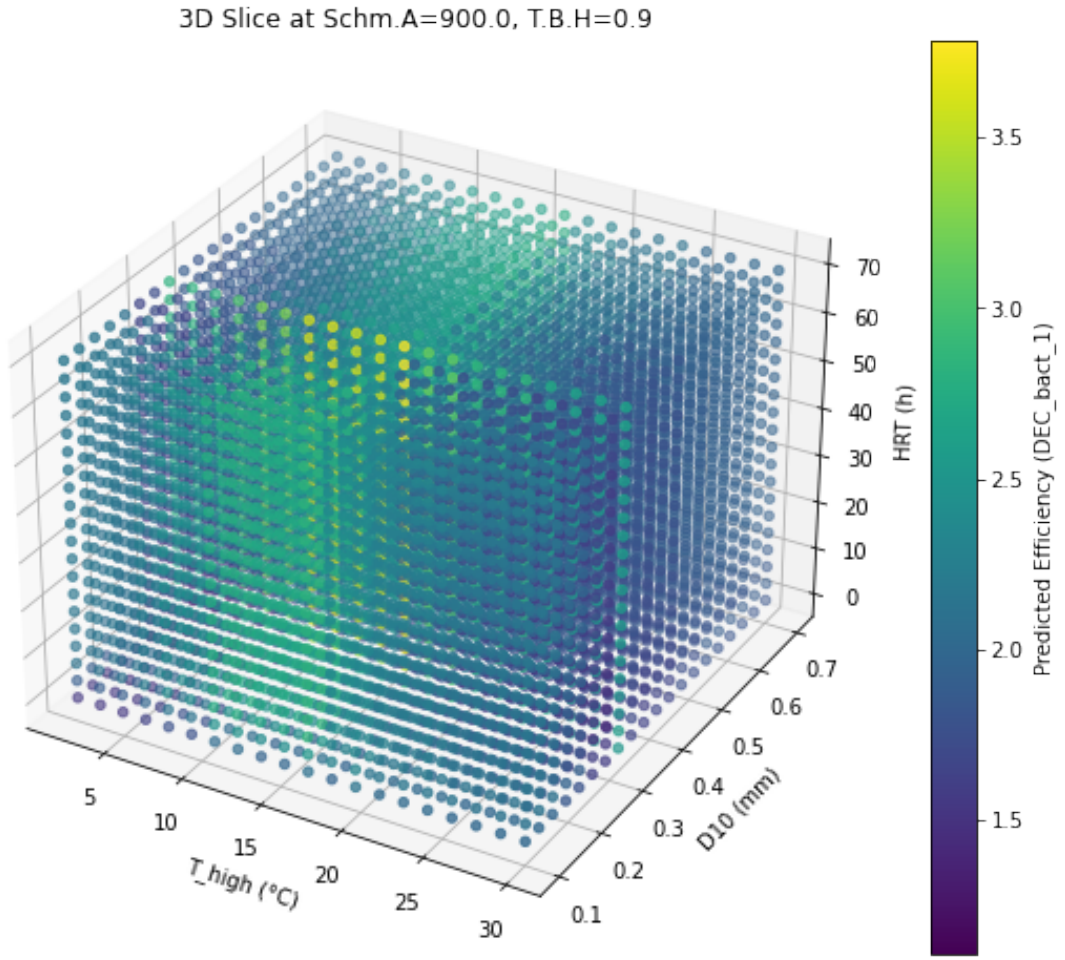

FIGURE H.1: 3D Slice at Schm.A = 900.0, T.B.H = 0.9 showing interactions between T_high (℃), D_{10} (mm), and HRT (h) for removal of bacteria.

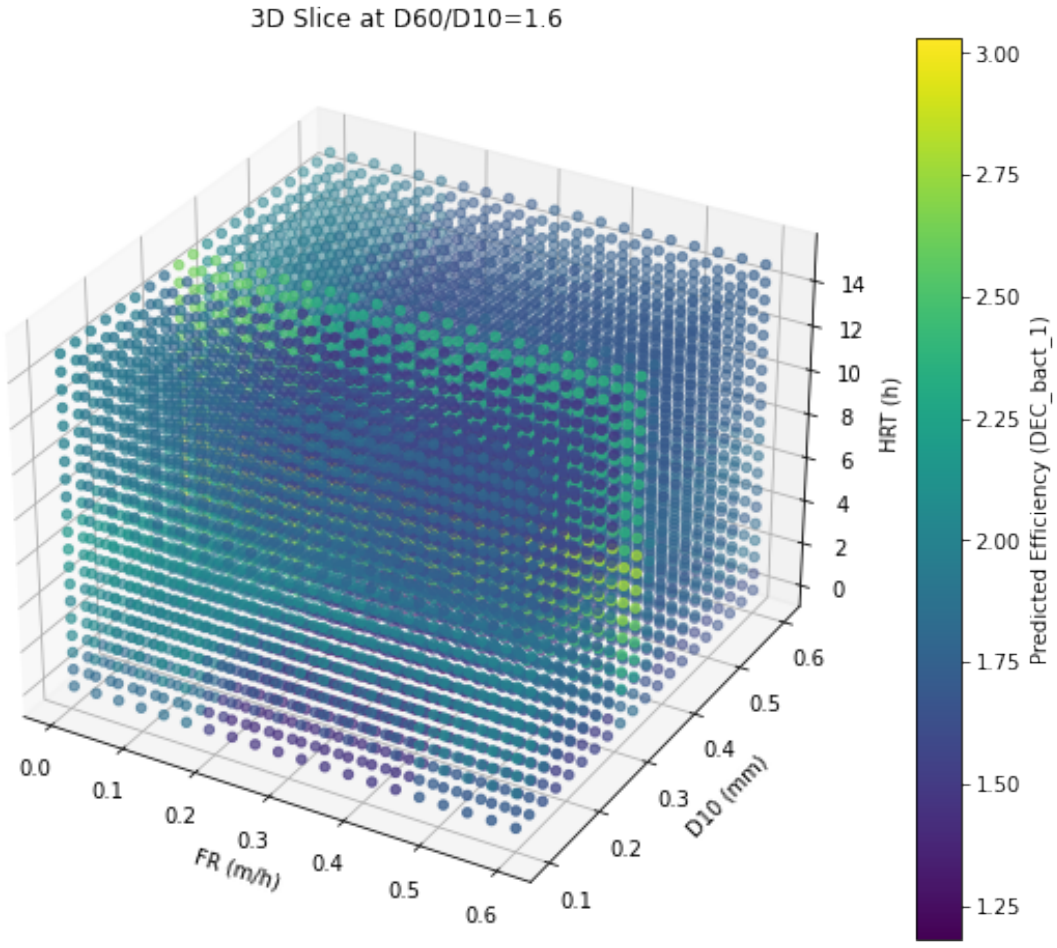

FIGURE H.2: 3D Slice at U.C. of 1.6, showing interactions between controllable parameters FR (m/h), D_{10} (mm), and HRT (h) for removal of bacteria.

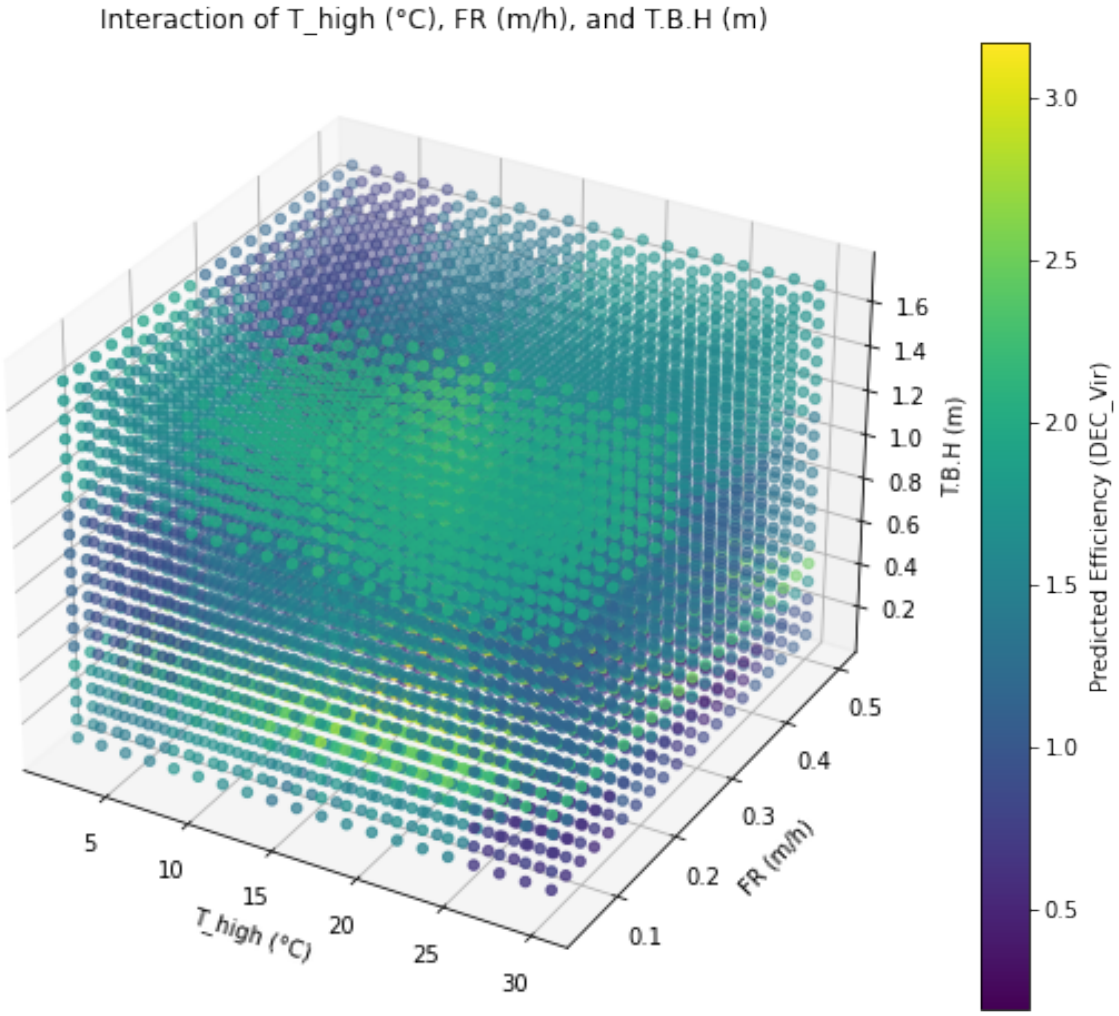

FIGURE H.3: Interaction of T_high (℃), FR (m/h), and T.B.H (m) showing removal efficiency predictions for virus removal.

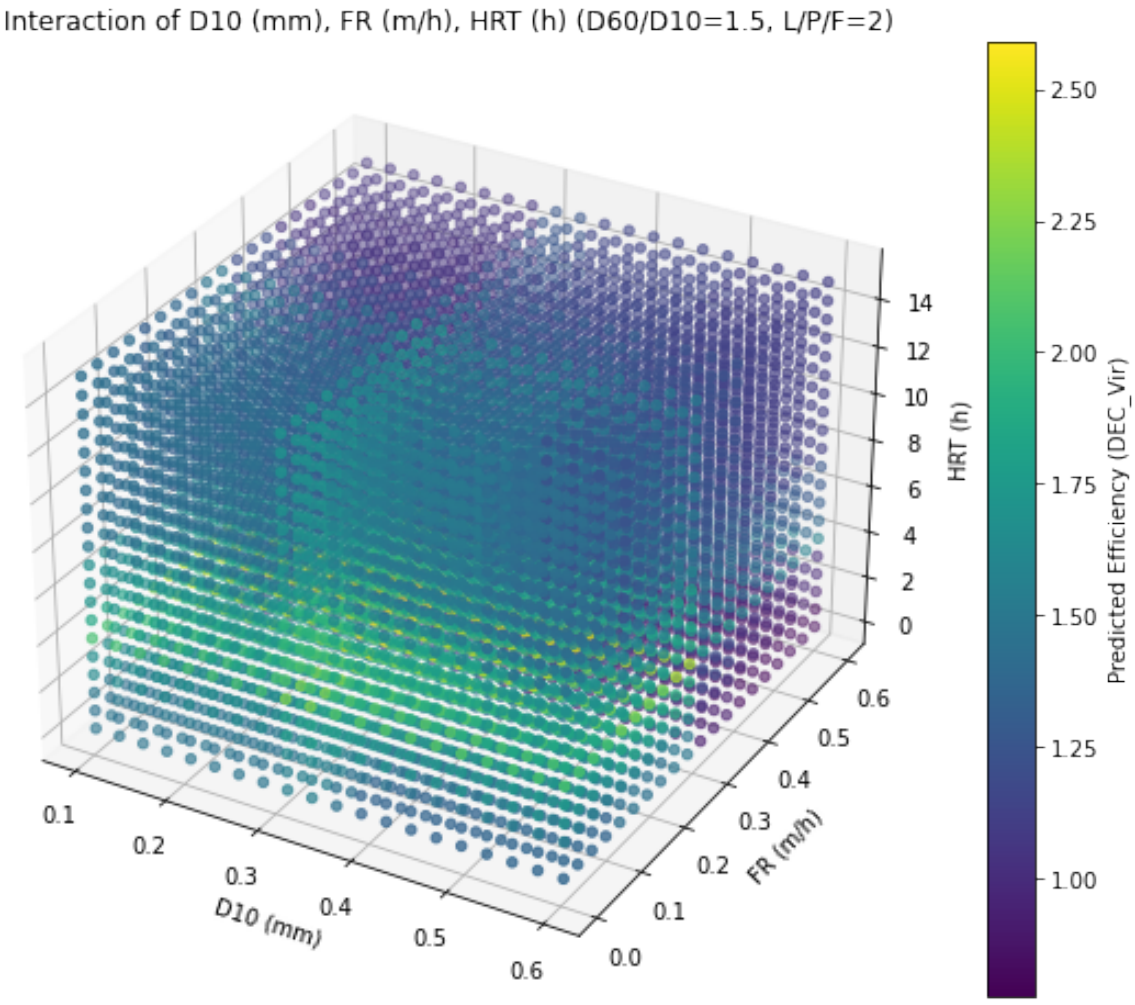

FIGURE H.4: 3D Slice at U.C. of 1.5 and pilot scale, showing interactions between controllable parameters FR (m/h), D_{10} (mm), HRT (h) for removal of virus.

Appendix I

Key Findings on the Influence of Design Parameters and/or Operational Conditions on Bacterial and Virus Removal

TABLE I.1: Key Findings on the Influence of Design Parameters and/or Operational Conditions on Bacterial Removal Efficiency, part 1

Key Findings	Observations
Optimal Temperature Range (15–20 °C)	The highest removal efficiencies for bacteria occur within a moderate temperature range of 15–20 °C, with performance declining noticeably below 10 °C and above 20 °C. However, temperatures between 10–20 °C still maintain removal efficiencies above 2 logs, showing consistent performance across various design parameters. The model indicates that removal efficiency consistently improves as temperature approaches the 15–20 °C range, regardless of interactions with other parameters.
Temperature in relation to D_{10}	Removal efficiencies remain low across all temperatures when D_{10} is between 0.15–0.30 mm. In contrast, sizes between 0.45–0.70 mm show improved removal efficiencies across a wide range of temperatures, demonstrating robust performance. At a very small D_{10} (<0.15 mm), removal efficiencies improve slightly when temperatures exceed 15 °C, while larger grain sizes (0.70 mm) perform adequately even at lower temperatures (10 °C).
Temperature in relation to HRT	Removal efficiencies are highest when the HRT is between 4 and 6 hours, particularly within a temperature range of 15–20 °C. Higher HRTs also perform well in this range. Interestingly, at temperatures above 20 °C, short HRTs still maintain high efficiency, but if the HRT increases to around 8 hours at, for example, 25 °C, removal efficiency drops significantly. This suggests that low HRTs are more effective at higher temperatures for removal of bacteria
Temperature in relation to *Schmutzdecke* Layer Age	The *Schmutzdecke* layer appears to be most effective when it is approximately 250 days old, as a wide range of temperatures (10 to 25 °C) perform well with a *Schmutzdecke* of this age. While both younger and older layers show reasonably good performance, the highest removal efficiencies are consistently observed around the 250-day mark. In contrast, *Schmutzdecke* layers younger than 50 days exhibit poor removal efficiencies across all temperature ranges.
Temperature in relation to T.B.H.	Interactions between T.B.H. and temperature reveal that at optimal temperatures (16–19 °C), lower bed heights (0.5–0.7 m) can still achieve moderate removal efficiencies. Higher bed heights (>1.45 m) are less temperature-sensitive, maintaining high removal efficiencies even at lower temperatures. This suggests that lower T.B.H. values require moderate temperatures for optimal performance, while higher T.B.H. values allow for greater operational flexibility across a broader temperature range.
Optimal Range of D_{10} (0.30 - 0.35 mm)	The highest removal efficiencies for bacteria are achieved when the D_{10} is around 0.30–0.35 mm, reaching approximately 4.05 logs, regardless of other design parameters. Interestingly, a D_{10} around 0.27 mm consistently exhibit poor performance across all parameter combinations, indicating a critical inefficiency threshold. While both higher and lower effective sizes can still yield reasonable removal efficiencies, their performance heavily depends on interactions with other design parameters.

TABLE I.2: Key Findings on the Influence of Design Parameters and/or Operational Conditions on Bacterial Removal Efficiency, part 2

Finding	Observations
D_{10} in relation to *Schmutzdecke* Layer Age	As previously mentioned, the D_{10} performs best within a range of approximately 0.30 to 0.35 mm, achieving the highest bacterial removal efficiencies. Optimal performance is also closely linked to the age of the *Schmutzdecke*. When the *Schmutzdecke* is between 80 and 300 days old, removal efficiency remains consistently high at this effective size range. However, as the *Schmutzdecke* ages beyond 300 days, a slight decline in removal efficiency becomes noticeable at the 0.30–0.35 mm range. Interestingly, during the 80 to 250-day window, a wider range of effective sizes appears effective for bacterial removal. In contrast, when the *Schmutzdecke* ages between 300 and 1200 days, the range of 0.35 to 0.45 mm seems to lose its effectiveness, showing reduced bacterial removal performance. Beyond this point, larger effective sizes (greater than 0.45 mm) appear to outperform the 0.35–0.45 mm range in older *Schmutzdecke* systems.
D_{10} in relation to T.B.H.	Higher T.B.H.s offer greater flexibility, as a range of D_{10} can still achieve high removal efficiencies under these conditions. In contrast, at lower bed heights, maintaining an effective size of approximately 0.30 to 0.35 mm proves to be the most effective, consistently delivering the highest removal efficiencies. The absolute peak removal efficiency is observed at a T.B.H. of around 1.5 m, combined with an effective size of 0.35 mm, representing an optimal configuration for bacterial removal. Interestingly, as the D_{10} increases, the required T.B.H. can be reduced while still maintaining high removal efficiencies. This suggests that a larger D_{10} allows for shorter T.B.H.s without significantly compromising performance.
HRT in relation to T.B.H.	A HRT between 4 and 6 hours, combined with a T.B.H. between 1 and 1.6 m, appears to be the most effective configuration for bacterial removal. Higher HRTs within this T.B.H. range also maintain strong performance. Interestingly, the model indicates that a T.B.H. between 0.8 and 1 m, when combined with HRTs exceeding 6 hours, results in poor removal efficiency. In contrast, T.B.H. values below 0.8 m, paired with HRTs of 4 hours or more, still achieve effective bacterial removal. However, performance tends to decline when the HRT increases to between 8 and 10 hours within this lower T.B.H. range.

TABLE I.3: Key Findings on the Influence of Design Parameters and/or Operational Conditions on Virus Removal Efficiency

Finding	Observations
Optimal Temperature Range (10–20 °C)	The model suggests that the optimal temperature for virus removal is between 10 and 20 °C. However, temperatures outside this range, both higher and lower, can still achieve effective virus removal depending on other design parameters. Virus removal appears to be less sensitive to temperature variations compared to bacterial removal. Notably, higher temperatures remain effective for virus removal, whereas bacterial removal efficiency tends to decline more sharply outside the optimal temperature range.
Temperature in relation to the FR	The highest virus removal efficiency occurs at moderate temperatures (16–18 °C) with lower filtration rates (0.10–0.30 m/h). Below 8 °C, efficiency drops significantly, especially at higher FR values (0.33–0.50 m/h). FR values between 0.30 and 0.40 m/h generally show low efficiency across most temperatures, with slight improvement around 15 °C. Interestingly, when FR exceeds 0.40 m/h, a broader temperature range can still maintain effective removal, indicating greater flexibility at higher flow rates. When temperatures fluctuate, an FR between 0.30 and 0.40 m/h should either increase to 0.40–0.50 m/h or decrease below 0.30 m/h for optimal results. During low temperatures, FR values around 0.15–0.20 m/h are the most effective, while higher FR values significantly reduce removal efficiency.
Temperature in relation to T.B.H.	The highest virus removal efficiencies are observed at moderate temperatures (14–18 °C) combined with a T.B.H. of approximately 0.4–0.6 m. At lower bed heights (<0.3 m), removal efficiency remains consistently low across most temperature ranges, with a slight improvement noticeable around 18 °C. This pattern suggests that a T.B.H. range of 0.4–0.6 m is optimal for virus removal. Outside this range, the influence of temperature becomes less pronounced, and efficiency stabilizes at generally lower levels. Interestingly, at lower temperatures (around 14 °C), relatively high removal efficiencies are maintained up to a T.B.H. of 1.2 m. Beyond this point, efficiency begins to diminish, likely due to hydraulic limitations or reduced biological activity. At 18 °C, removal efficiency remains consistently high across the entire observed bed height range, indicating a more stable and predictable performance at this temperature, regardless of bed height variations. This suggests that 18 °C acts as a stabilizing temperature point, enabling efficient virus removal across a wider range of T.B.H. values.
FR in relation to T.B.H.	The model indicates that the optimal FR range is 0.12–0.30 m/h when the T.B.H. is 0.4–0.5 m. Within this range, FR can be increased up to 0.50 m/h while still achieving better removal efficiency than lower FR values (e.g., 0.25 m/h) combined with significantly taller bed heights. When T.B.H. increases to 0.5–1.2 m, the effective FR range narrows to 0.15–0.20 m/h. Outside this range, efficiency declines. At even greater bed heights (>1.2 m), lower FR values (<0.10 m/h) show slight improvements, but overall efficiency remains limited. Interestingly, FR values between 0.12 and 0.20 m/h allow lower bed heights (<0.4 m) to maintain high efficiencies. However, FR values exceeding 0.20 m/h or dropping below 0.12 m/h result in a sharp decline in removal efficiency.

Appendix J

Key Findings on the Influence of Controllable Design Parameters on Bacterial and Virus Removal

The previously showcased key findings in tables I.1, I.2 and I.3 highlighted the design parameters and operational conditions that were most influential in the model's prediction performance for the removal of bacteria and viruses. These critical features were further visualized in figures 4.4 and 4.8, offering insights into how these parameters contribute to the removal efficiencies. However, as previously mentioned, it is equally important to focus specifically on the controllable design parameters, which are illustrated in figures 4.15 and 4.16, as focusing on these allows users to practically optimize filter design and operation. It is important to note, however, that the XGBoost model's performance in predicting bacterial and viral removal was lower when using only controllable design parameters. As a result, the key findings derived from these parameters may have reduced accuracy and should be interpreted with caution.

The key findings on controllable design parameters for bacterial removal, derived from the 2D heatmaps, are summarized in Table J.1. Similarly, the key findings for virus removal, focusing on controllable design parameters, are presented in Table J.2.

TABLE J.1: Key Findings on the Influence of Controllable Design Parameters on Bacterial Removal Efficiency

Key Findings	Observations
FR in relation to D_{10}	The highest bacterial removal efficiencies are achieved when the FR is between 0.05 and 0.16 m/h and the D_{10} ranges from 0.29 to 0.34 mm. Within this D_{10} range, higher FR values can still maintain reasonable efficiency, but performance tends to decline as FR increases further. Interestingly, when the D_{10} exceeds 0.35 mm or drops below 0.29 mm, higher FR values lead to a significant reduction in removal efficiency. In such cases, it is more effective to opt for a lower FR to ensure optimal bacterial removal performance.
FR in relation to HRT	As previously mentioned, the highest bacterial removal efficiencies are achieved when the HRT is between 4 to 8 hours. Combining this range with a low FR appears to be the most effective strategy. However, higher FRs within this optimal HRT range can also yield good removal performance. Interestingly, when the FR drops below 0.12 m/h, removal efficiency remains consistently high, regardless of the HRT value. This suggests that very low flow rates compensate for shorter contact times in the filter. A notable exception is observed in the heatmap, where removal efficiencies are significantly low in the range of 0.17–0.44 m/h for FR combined with an HRT of 0.1–4 hours. This region highlights a clear mismatch between flow rate and retention time, limiting bacterial removal performance. Lastly, when the HRT exceeds 8 hours, efficiency begins to decline for FR values above 0.17 m/h, suggesting diminishing returns from extended retention times at higher flow rates.
FR in relation to U.C.	As stated previously, the highest bacterial removal efficiencies are achieved when the U.C. is between 1.4 and 2, combined with a FR, ranging from 0.05 to 0.12 m/h. Within this optimal range, the pore structure remains consistent, allowing for effective filtration and bacterial retention. A higher FR values can also maintain effective removal efficiencies, provided the U.C. remains within a narrower range of approximately 1.62 to 1.9.
D_{10} in relation to U.C.	As previously mentioned, the optimal D_{10} for bacterial removal lies between 0.30 and 0.35 mm, while the U.C. is most effective between 1.4 and 2. Interestingly, higher U.C. values can also maintain effective removal efficiencies, provided that the effective size remains within the optimal range of 0.30 to 0.35mm. This suggests that a stable D_{10} can compensate for increased grain size variability, preserving the consistency of the pore structure and supporting reliable filtration performance.

TABLE J.2: Key Findings on the Influence of Controllable Design Parameters on Virus Removal Efficiency

Key Findings	Observations
FR in relation to D_{10}	The same findings apply to virus removal as those observed for bacterial removal, as summarized in table J.1, specifically for lab-scale experiments. The optimal range of FR is slightly lower for virus removal, namely between 0.05 and 0.11 m/h. However, higher FRs can still achieve effective removal, provided the effective size remains within the optimal range of 0.30–0.35 mm. Interestingly, when the perspective shifts from lab-scale to pilot- or full-scale experiments, higher FR values—up to 0.3 m/h—are also effective. The optimal range for effective size (0.30–0.35 mm) remains consistent across scales, but both lower and higher effective sizes can still yield good removal efficiencies when paired with FR values below 0.31 m/h.
FR in relation to HRT	At both lab-scale and pilot/full-scale operations, the optimal HRT ranges again from 4 to 8 hours, particularly when combined with FRs below 0.3 m/h. This combination consistently yields high removal efficiencies across different experimental scales. However, at pilot/full-scale operations, FR values exceeding 0.30 m/h within the 4–8 hour HRT range demonstrate reduced performance compared to lab-scale experiments.
FR in relation to U.C.	The same findings apply to virus removal as those observed for bacterial removal, as summarized in table J.1. In pilot/full-scale operations, higher FRs, up to 0.31 m/h, can still achieve good removal efficiencies. Additionally, both low U.C. values, around 1.4, and higher U.C., values up to 2.6, can yield effective results at pilot/full-scale operations. This suggests that, at larger scales, there is greater flexibility in optimizing FR and U.C. combinations, allowing for consistent virus removal performance across a broader range of operational conditions. However, highest removal efficiencies at full scale are found when the value of the U.C. is around 1.90 to 2.1.
D_{10} in relation to HRT	The same findings apply to virus removal as those observed for bacterial removal, as summarized in table J.1. This consistency holds true across both lab-scale and pilot/full-scale operations.

Bibliography

[1] John K. Maiyo, Sruthi Dasika, and Chad T. Jafvert. “Slow Sand Filters for the 21st Century: A Review”. In: *International Journal of Environmental Research and Public Health* 20.2 (2023). ISSN: 1660-4601. DOI: 10.3390/ijerph20021019. URL: https://www.mdpi.com/1660-4601/20/2/1019.

[2] Geneva WHO et al. “Guidelines for drinking-water quality”. In: *World health organization* 216 (2011), pp. 303–304.

[3] AM de Roda Husman and G Medema. *Inspectierichtlijn, Analyse microbiologische veiligheid drinkwater, VROM-inspectie, 34 pp*. 2004.

[4] Jack F. Schijven et al. “A mathematical model for removal of human pathogenic viruses and bacteria by slow sand filtration under variable operational conditions”. In: *Water Research* 47.7 (2013), pp. 2592–2602. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2013.02.027. URL: https://www.sciencedirect.com/science/article/pii/S0043135413001310.

[5] Hannah Ritchie and Max Roser. *Clean Water and Sanitation*. Our World in Data. 2021. URL: https://ourworldindata.org/clean-water-sanitation.

[6] Jan van der Hoek, Doris van Halem, and Hauke Smidt. *Slow Sand Filtration for the Next Century*. 2020.

[7] Sarah-Jane Haig et al. “Biological aspects of slow sand filtration: Past, present and future”. In: *Water Science Technology* 11 (Sept. 2011). DOI: 10.2166/ws.2011.076.

[8] William Bellamy, David Hendricks, and Gary Logsdon. “Slow Sand Filtration: Influences of Selected Process Variables”. In: *Journal American Water Works Association - J AMER WATER WORK ASSN* 77 (Dec. 1985), pp. 62–66. DOI: 10.1002/j.1551-8833.1985.tb05659.x.

[9] Ephrem Guchi. “Review on Slow Sand Filtration in Removing Microbial Contamination and Particles from Drinking Water”. In: *American Journal of Food and Nutrition* 3.2 (2015), pp. 47–55. ISSN: 2374-1163. DOI: 10.12691/ajfn-3-2-3. URL: http://pubs.sciepub.com/ajfn/3/2/3.

[10] J. J. Troyan and S. P. Hansen. *Treatment of Microbial Contaminants in Potable Water Supplies*. Park Ridge, N. J.: Noyes Data Corporation, 1989, pp. 5–54.

[11] Jianan Li, Qizhi Zhou, and Luiza C. Campos. “The application of GAC sandwich slow sand filtration to remove pharmaceutical and personal care products”. In: *Science of The Total Environment* 635 (2018), pp. 1182–1190. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2018.04.198. URL: https://www.sciencedirect.com/science/article/pii/S0048969718313767.

[12] Jack F. Schijven, S.Majid Hassanizadeh, and Ria H.A.M. de Bruin. “Two-site kinetic modeling of bacteriophages transport through columns of saturated dune sand”. In: *Journal of Contaminant Hydrology* 57.3 (2002), pp. 259–279. ISSN: 0169-7722. DOI: https://doi.org/10.1016/S0169-7722(01)00215-7. URL: https://www.sciencedirect.com/science/article/pii/S0169772201002157.

[13] Jack Schijven and S. Hassanizadeh. “Removal of Viruses by Soil Passage: Overview of Modeling, Processes, and Parameters”. In: *Critical Reviews in Environmental Science and Technology - CRIT REV ENVIRON SCI TECHNOL* 30 (Jan. 2000), pp. 49–127. DOI: 10.1080/10643380091184174.

[14] Ephrem Guchi. “Review on Slow Sand Filtration in Removing Microbial Contamination and Particles from Drinking Water”. In: *American Journal of Food and Nutrition* 3.2 (2015), pp. 47–55. ISSN: 2374-1163. DOI: 10.12691/ajfn-3-2-3. URL: http://pubs.sciepub.com/ajfn/3/2/3.

[15] Prem Ranjan and Manjeet Prem. “Schmutzdecke- A Filtration Layer of Slow Sand Filter”. In: *International Journal of Current Microbiology and Applied Sciences* 7 (July 2018), pp. 637–645. DOI: 10.20546/ijcmas.2018.707.077.

[16] Hemant Arora. “Optimising the Ripening Period of Slow Sand Filters”. PhD thesis. Delft University of Technology, 2017.

[17] Shreya Ajith Trikannad et al. “The contribution of deeper layers in slow sand filters to pathogens removal”. In: *Water Research* 237 (2023), p. 119994. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2023.119994. URL: https://www.sciencedirect.com/science/article/pii/S004313542300430X.

[18] Frances C. Pick et al. “Application of enhanced assimilable organic carbon method across operational drinking water systems”. In: *PLOS ONE* 14.12 (Dec. 2019), pp. 1–24. DOI: 10.1371/journal.pone.0225477. URL: https://doi.org/10.1371/journal.pone.0225477.

[19] Marion W. Jenkins, Sangam K. Tiwari, and Jeannie Darby. “Bacterial, viral and turbidity removal by intermittent slow sand filtration for household use in developing countries: Experimental investigation and modeling”. In: *Water Research* 45.18 (2011), pp. 6227–6239. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2011.09.022. URL: https://www.sciencedirect.com/science/article/pii/S0043135411005410.

[20] Andrew J Logan et al. “Transport and fate of Cryptosporidium parvum oocysts in intermittent sand filters”. In: *Water Research* 35.18 (2001), pp. 4359–4369. ISSN: 0043-1354. DOI: https://doi.org/10.1016/S0043-1354(01)00181-6. URL: https://www.sciencedirect.com/science/article/pii/S0043135401001816.

[21] Jules J. Berman. “Chapter 4 - Understanding Your Data”. In: *Data Simplification*. Ed. by Jules J. Berman. Boston: Morgan Kaufmann, 2016, pp. 135–187. ISBN: 978-0-12-803781-2. DOI: https://doi.org/10.1016/B978-0-12-803781-2.00004-7. URL: https://www.sciencedirect.com/science/article/pii/B9780128037812000047.

[22] CourseHorse. *Visualizing Relationships with Scatter Plots and Trend Lines*. Accessed: 2025-01-24. n.d. URL: https://coursehorse.com/blog/learn/data-analytics/visualizing-relationships-with-scatter-plots-and-trend-lines.

[23] I. H. Sarker. “Machine Learning: Algorithms, Real-World Applications and Research Directions”. In: *SN Computer Science* 2.3 (2021), p. 160. DOI: 10.1007/s42979-021-00592-x.

[24] Arthur Graus. *Classification Models - XGBoost*. Accessed: 2024-12-22. URL: https://www.arthurgraus.nl/xgboost.html.

[25] Vinod Chugani. *Navigating Missing Data Challenges with XGBoost*. Accessed: 2024-12-22. 2024. URL: https://machinelearningmastery.com/navigating-missing-data-challenges-with-xgboost/.

[26] UW-Madison Data Science. *XGBoost*. Accessed: 2024-12-22. 2024. URL: https://uw-madison-datascience.github.io/ML-X-Nexus/Toolbox/Models/XGBoost.html.

[27] “Avoiding Overfitting of Decision Trees”. In: *Principles of Data Mining*. London: Springer London, 2007, pp. 119–134. ISBN: 978-1-84628-766-4. DOI: 10.1007/978-1-84628-766-4_8. URL: https://doi.org/10.1007/978-1-84628-766-4_8.

[28] Lior Rokach and Oded Maimon. “Decision Trees”. In: *Data Mining and Knowledge Discovery Handbook*. Ed. by Oded Maimon and Lior Rokach. Boston, MA: Springer US, 2005, pp. 165–192. ISBN: 978-0-387-25465-4. DOI: 10.1007/0-387-25465-X_9. URL: https://doi.org/10.1007/0-387-25465-X_9.

[29] J. R. Quinlan. “Induction of Decision Trees”. In: *Mach. Learn.* 1.1 (Mar. 1986), 81–106. ISSN: 0885-6125. DOI: 10.1023/A:1022643204877. URL: https://doi.org/10.1023/A:1022643204877.

[30] DataCamp. *Decision Tree Classification in Python: Tutorial*. Accessed: January 26, 2025. n.d. URL: https://www.datacamp.com/tutorial/decision-tree-classification-python.

[31] Christoph Molnar. *Interpretable Machine Learning: A Guide For Making Black Box Models Explainable*. Independently published, 2022. ISBN: 979-8411463330. URL: https://www.amazon.com/dp/B09TMWHVB4.

[32] upGrad. *Understanding Gini Index for Decision Trees*. Accessed: January 26, 2025. n.d. URL: https://www.upgrad.com/blog/gini-index-for-decision-trees/.

[33] Jason Brownlee. *Train-Test Split for Evaluating Machine Learning Algorithms*. Accessed: January 26, 2025. n.d. URL: https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/.

[34] Scikit learn Developers. *sklearn.tree.DecisionTreeRegressor*. Accessed: January 26, 2025. n.d. URL: https://scikit-learn.org/dev/modules/generated/sklearn.tree.DecisionTreeRegressor.html.

[35] Scikit learn Developers. *sklearn.metrics.mean_squared_error*. Accessed: January 26, 2025. n.d. URL: https://scikit-learn.org/1.5/modules/generated/sklearn.metrics.mean_squared_error.html.

[36] Scikit learn Developers. *sklearn.metrics.r2_score*. Accessed: January 26, 2025. n.d. URL: https://scikit-learn.org/1.6/modules/generated/sklearn.metrics.r2_score.html.

[37] Scikit-learn developers. *Decision Trees — scikit-learn 1.3.0 documentation*. Accessed: [Date]. 2024. URL: https://scikit-learn.org/stable/modules/tree.html#tree.

[38] NVIDIA. *What is XGBoost? | NVIDIA Glossary*. Accessed: January 26, 2025. n.d. URL: https://www.nvidia.com/en-us/glossary/xgboost/.

[39] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145/2939672.2939785.

[40] Majid Niazkar et al. "Applications of XGBoost in water resources engineering: A systematic literature review (Dec 2018–May 2023)". In: *Environmental Modelling Software* 174 (2024), p. 105971. ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2024.105971. URL: https://www.sciencedirect.com/science/article/pii/S136481522400032X.

[41] LazyProgrammer. *XGBoost*. Accessed: 2024-12-22. 2024. URL: https://lazyprogrammer.me/mlcompendium/ensemble/xgboost.html.

[42] XGBoost Contributors. *XGBoost Documentation: Model Training Tutorial*. Accessed: January 26, 2025. n.d. URL: https://xgboost.readthedocs.io/en/stable/tutorials/model.html.

[43] Wikipedia contributors. *Taylor series — Wikipedia, The Free Encyclopedia*. Accessed: 2025-01-27. 2025. URL: https://en.wikipedia.org/wiki/Taylor_series.

[44] Plain English. *The Complete XGBoost Therapy with Python*. Accessed: 2025-01-28. 2025. URL: https://python.plainenglish.io/the-complete-xgboost-therapy-with-python-87c8cffcb71f.

[45] Gabriel Tseng. *Gradient Boosting and XGBoost*. Accessed: 2025-01-28. 2018. URL: https://gabrieltseng.github.io/posts/2018-02-25-XGB/.

[46] *XGBoost Python Package Documentation*. Accessed: 2025-01-28. 2025. URL: https://xgboost.readthedocs.io/en/stable/python/index.html.

[47] Jason Brownlee. *XGBoost for Regression*. 2021. URL: https://machinelearningmastery.com/xgboost-for-regression/.

[48] Rahul Shah. "Tune Hyperparameters with GridSearchCV". In: *Analytics Vidhya* (2021). URL: https://www.analyticsvidhya.com/blog/2021/06/tune-hyperparameters-with-gridsearchcv/.

[49] Rick Sun. *XGBoost StratifiedKFold for Beginner*. Accessed: 2024-01-29. 2021. URL: https://www.kaggle.com/code/ricksun/xgboost-stratifiedkfold-for-beginner.

[50] XGBoost Documentation. *XGBoost Parameters*. Accessed: 2024-01-28. 2024. URL: https://xgboost.readthedocs.io/en/stable/parameter.html.

[51] Maros von Sperling and Silvia M.A.C. Oliveira. "Loading rates applied to treatment units". In: *Assessment of Treatment Plant Performance and Water Quality Data: A Guide for Students, Researchers and Practitioners*. Ed. by Matthew E.Editor Verbyla. IWA Publishing, 2020.

[52] L Huisman and W. E Wood. *Slow sand filtration / L. Huisman, W. E. Wood*. 1974.

[53] Wim Hijnen et al. "Elimination of viruses, bacteria and protozoan oocysts by slow sand filtration". In: *Water science and technology : a journal of the International Association on Water Pollution Research* 50 (July 2004), pp. 147–54. DOI: 10.2166/wst.2004.0044.

[54] Caleb White et al. "Effect of contaminated filtration sand on performance of household biosand filters". In: *2013 IEEE Global Humanitarian Technology Conference (GHTC)*. 2013, pp. 243–247. DOI: 10.1109/GHTC.2013.6713688.

[55] Qaisar Mahmood et al. "Development of low cost household drinking water treatment system for the earthquake affected communities in Northern Pakistan". In: *Desalination* 273.2 (2011), pp. 316–320. ISSN: 0011-9164. DOI: https://doi.org/10.1016/j.desal.2011.01.052. URL: https://www.sciencedirect.com/science/article/pii/S0011916411000634.

[56] Candice Young-Rojanschi and Chandra Madramootoo. "Intermittent versus continuous operation of biosand filters". In: *Water Research* 49 (2014), pp. 1–10. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2013.11.011. URL: https://www.sciencedirect.com/science/article/pii/S0043135413009226.

[57] Tommy Ngai et al. "Global review of the adoption, use and performance of the biosand filter". In: June 2014, pp. 309–317. ISBN: 9781780406374.

[58] Peter F. Schuler, Mriganka M. Ghosh, and Prasad Gopalan. “Slow sand and diatomaceous earth filtration of cysts and other particulates”. In: *Water Research* 25.8 (1991), pp. 995–1005. ISSN: 0043-1354. DOI: https://doi.org/10.1016/0043-1354(91)90149-K. URL: https://www.sciencedirect.com/science/article/pii/004313549190149K.

[59] Rosalie Bauer et al. “Removal of bacterial fecal indicators, coliphages and enteric adenoviruses from waters with high fecal pollution by slow sand filtration”. In: *Water Research* 45.2 (2011), pp. 439–452. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2010.08.047. URL: https://www.sciencedirect.com/science/article/pii/S0043135410006111.

[60] Caleb White et al. “Effect of contaminated filtration sand on performance of household biosand filters”. In: Oct. 2013, pp. 243–247. ISBN: 978-1-4799-2401-1. DOI: 10.1109/GHTC.2013.6713688.

[61] Silvano Pereira et al. “Removal of cyanobacteria by slow sand filtration for drinking water”. In: *Journal of Water, Sanitation and Hygiene for Development* 2 (Sept. 2012), p. 133. DOI: 10.2166/washdev.2012.047.

[62] National Drinking Water Clearinghouse (U.S.) *Slow Sand Filtration. Tech Brief Fourteen*. 2000.

[63] Kaldibek Abdiyev et al. “Review of Slow Sand Filtration for Raw Water Treatment with Potential Application in Less-Developed Countries”. In: *Water* 15.11 (2023). ISSN: 2073-4441. DOI: 10.3390/w15112007. URL: https://www.mdpi.com/2073-4441/15/11/2007.

[64] Nur Muhammad et al. “Optimization of slow sand filtration”. In: 1996. URL: https://api.semanticscholar.org/CorpusID:55128129.

[65] Rehan Sadiq et al. “Performance evaluation of slow sand filters using fuzzy rule-based modelling”. In: *Environmental Modelling Software* 19.5 (2004), pp. 507–515. ISSN: 1364-8152. DOI: https://doi.org/10.1016/S1364-8152(03)00165-8. URL: https://www.sciencedirect.com/science/article/pii/S1364815203001658.

[66] William Anderson et al. “Influence of Design and Operating Conditions on the Removal of MS2 Bacteriophage by Pilot-scale Multistage Slow Sand Filtration.” In: *Journal of Water Supply Research and Technology-Aqua* 58 (Jan. 2009), p. 450.

[67] B. Lesikar. *Sand Filters for Home Use—Texas Agricultural Extension Service*. 2023. URL: http://www.scribd.com/doc/34621075/Sand-filters-for-home-use-Texas-Agricultural-Extension-Service.

[68] Doug Fogel et al. “Removing Giardia and Cryptosporidium by Slow Sand Filtration”. In: *Journal AWWA* 85.11 (1993), pp. 77–84. DOI: https://doi.org/10.1002/j.1551-8833.1993.tb06105.x. eprint: https://awwa.onlinelibrary.wiley.com/doi/pdf/10.1002/j.1551-8833.1993.tb06105.x. URL: https://awwa.onlinelibrary.wiley.com/doi/abs/10.1002/j.1551-8833.1993.tb06105.x.

[69] M.A. Boller and M.C. Kavanaugh. “Particle characteristics and headloss increase in granular media filtration”. In: *Water Research* 29.4 (1995), pp. 1139–1149. ISSN: 0043-1354. DOI: https://doi.org/10.1016/0043-1354(94)00256-7. URL: https://www.sciencedirect.com/science/article/pii/0043135494002567.

[70] P.G. Williams. “A study of bacteria reduction by slow sand filtration”. In: *Paper Presented at the 1987 IWPC Biennial Conference, Port Elizabeth, South Africa* (1987, National Institute for Water Research), pp. 12–15.

[71] K. V. Ellis and W. E. Wood. “Slow sand filtration”. In: *Critical Reviews in Environmental Control* 15.4 (1985), pp. 315–354. DOI: 10.1080/10643388509381736. eprint: https://doi.org/10.1080/10643388509381736. URL: https://doi.org/10.1080/10643388509381736.

[72] Aloo Becky Nancy, M Madu Josephine, and Mwamburi Lizzy. “Slow Sand Filtration of Secondary Sewage-Effluent: Effect of Sand Bed Depth on FilterPerformance”. In: *International Journal of Innovative Research in Science, Engineering and Technology* 3 (2014), pp. 15090–15099. URL: https://api.semanticscholar.org/CorpusID:54645543.

[73] Marion Jenkins et al. “The BioSand Filter for Improved Drinking Water Quality in High Risk Communities in the Njoro Watershed, Kenya”. In: *SUMAWA Research Brief* (July 2009).

[74] Matteo D’Alessio et al. “A low-cost water-treatment system for potable water supplies in developing countries and after a natural disaster: Ability to remove total coliforms and E. coli”. In: *Clean Technologies and Environmental Policy* 18 (Mar. 2016). DOI: 10.1007/s10098-015-1074-y.

[75] Hanting Wang et al. “MS2 Bacteriophage Reduction and Microbial Communities in Biosand Filters”. In: *Environmental Science & Technology* 48.12 (2014). PMID: 24857308, pp. 6702–6709. DOI: 10.1021/es500494s. eprint: https://doi.org/10.1021/es500494s. URL: https://doi.org/10.1021/es500494s.

[76] Gary Amy et al. “Integrated comparison of biofiltration in engineered versus natural systems”. In: *Recent progress in slow sand and alternative biofiltration processes* 1 (2006), pp. 3–11.

[77] MA Elliott, FA DiGiano, and MD Sobsey. “Virus attenuation by microbial mechanisms during the idle time of a household slow sand filter”. In: *Water research* 45.14 (2011), pp. 4092–4102.

[78] G. Grützmacher et al. “Removal of microcystins by slow sand filtration”. In: *Environmental toxicology* 17 (Aug. 2002), pp. 386–94. DOI: 10.1002/tox.10062.

[79] New England Water Treatment Technology Assistance Center. *Assessing Temperature Influences on Slow Sand Filtration Treatment Performance*. Accessed on 11 november 2023. Accessed 2023. URL: https://www.unh.edu/wttac/Project_Summaries/assessing_temperature_slow_sand.pdf.

[80] John K. Maiyo, Sruthi Dasika, and Chad T. Jafvert. “Slow Sand Filters for the 21st Century: A Review”. In: *International Journal of Environmental Research and Public Health* 20.2 (2023). ISSN: 1660-4601. DOI: 10.3390/ijerph20021019. URL: https://www.mdpi.com/1660-4601/20/2/1019.

[81] Doug Fogel et al. “Removing Giardia and Cryptosporidium by Slow Sand Filtration”. In: *Journal American Water Works Association - J AMER WATER WORK ASSN* 85 (Nov. 1993), pp. 77–84. DOI: 10.1002/j.1551-8833.1993.tb06105.x.

[82] Wikipedia contributors. *Bayesian information criterion — Wikipedia, The Free Encyclopedia*. [Online; accessed 23-December-2024]. 2024. URL: https://en.wikipedia.org/w/index.php?title=Bayesian_information_criterion&oldid=1260190481.

[83] Scott Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. 2017. arXiv: 1705.07874 [cs.AI]. URL: https://arxiv.org/abs/1705.07874.

[84] GeeksforGeeks. *ML | Bias-Variance Tradeoff*. Accessed: 2024-06-10. 2024. URL: https://www.geeksforgeeks.org/ml-bias-variance-trade-off/?utm_source=chatgpt.com.