University of Amsterdam

Masters Thesis

Mathematically Modeling the Interactions Between Phages, Bacteria, and the Environment

Author:
Victor Piaskowski

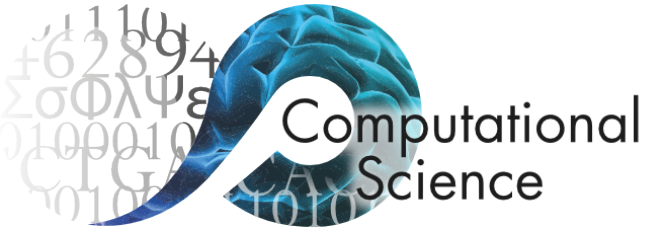
Examiner:
Dr. Jaap Kaandorp
Supervisor:
Dr. Matti Gralka
Assessor:
Dr. Yuval Mulla

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

in the

Computational Science Lab
Informatics Institute

June 2025



Declaration of Authorship

I, Victor Piaskowski, declare that this thesis, entitled 'Mathematically Modeling the Interactions Between Phages, Bacteria, and the Environment' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the University of Amsterdam.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date: June 30, 2025

"All models are wrong, but some are useful"

George E. P. Box

UNIVERSITY OF AMSTERDAM

Abstract

Faculty of Science
Informatics Institute

Master of Science in Computational Science

Mathematically Modeling the Interactions Between Phages, Bacteria, and the Environment

by Victor Piaskowski

In any given microbial ecosystem, hundreds of different phages and bacteria interact with one another in a complex system. These individual interactions have a significant impact on the bacterial community and the overall ecosystem, with lasting ecological consequences. They force bacteria to mutate and evolve, changing the community's fitness level. Phages can control bacterial populations, ensuring that dangerous bacteria do not over-reproduce. Phages control bacterial population with the "kill-the-winner" dynamic, preventing any one species from dominating the population. Upon lysis, bacteria release resources into the environment that other bacteria and plants can use.

Modeling these complex systems mathematically is crucial, as traditional laboratory experiments are often time-consuming and costly. While current models typically focus on one or two phage-bacteria pairs, classical models can be extended to accommodate multiple species. Such simulations can guide future laboratory work, particularly in addressing challenges such as antibiotic-resistant bacteria.

To address this, I developed a three-part program capable of modeling complex communities comprising of p phages, b bacteria, and r resources. The first part is a graphical user interface (GUI) tool that allows users to create and edit interaction networks between phages, bacteria, and resources. Nodes and edges within this network store interaction and environmental parameter values. The second part, the simulation framework, uses the user-defined network and ordinary differential equation (ODE) model to calculate population levels of phages, bacteria, and resources. The third part provides a custom-built dashboard for user interaction. Users can modify parameter values directly from the dashboard and immediately observe the effects of the simulation. The program includes five pre-built interactive plots, and users can download simulation results for custom analysis and further evaluation.

I applied this program to a model from Geng et al. [1], performing both qualitative and quantitative analyses. A Sobol analysis was used to quantify the variance in the model's output based on its inputs. Understanding phage restriction regimes and the factors that influence reaction rates is critical. By varying the initial uninfected bacterial population, the model goes from being adsorption limited to latency limited, two limiting processes.

It's vital to determine if a phage population will proliferate under constant washout conditions in a lab setting to prevent its accidental elimination. I analyzed phage proliferation based on initial conditions. Finally, I conducted a coexistence analysis on a large community of 20 phages, 20 bacteria, and 10 resources to assess phage-bacteria coexistence under various conditions.

In large communities, phages and bacteria can coexist. Introducing a term for debris that deactivates free phages increases the bacteria's survivability rate.

Researchers can utilize the flexible simulation framework to study the dynamics of complex communities, thereby deepening our understanding of phage-host dynamics in the future. Since it requires little to no programming skills on the part of the user, it can also serve as a valuable teaching tool.

Acknowledgements

I want to thank my parents for loving me despite some of my faults and for supporting me throughout my Bachelor's and Master's studies. Without them, I would not be where I am today, and I would certainly be a different person if it were not for them.

Thank you to Dr. Matti Gralka for the weekly meetings and for teaching me everything about phages and bacteria. Every meeting was always insightful, productive, engaging, and informative. I will forever be amazed at how he can recall which paper discusses which topic and how he always had a paper for every topic.

Thank you to Sofia Blaszczyk for finding the Master thesis opening and suggesting that I email Dr. Gralka for an introductory meeting. If she had not found this opening, I would not know what I would be doing for my thesis.

If I had not followed Dr. Rik Kaasschieter's and Dr. Martijn Anthonissen's courses, "Introduction Computational Sciences" and "Numerical Linear Algebra," in my Bachelor's program, I would not have been interested in Computational Sciences. I would not have found the MSc Computational Sciences program, as Computational Sciences fits my interests and skill sets better than any other program I could have taken. Rik and Martijn have forever altered my career trajectory.

Thank you to Sarah Flickinger for showing me the research she has been working on in the lab. She allowed me to connect my research and models to real life, reminding me that what I am doing has real-life applications rather than just being a purely theoretical or programming challenge.

Moreover, I want to thank my friends for keeping me sane and helping me throughout both of my programs.

Contents

List of Figures

List of Tables

Abbreviations

DDE	**D**elay **D**ifferential **E**quation
DNA	**D**eoxyribo**N**ucleic **A**cid
GUI	**G**raphical **U**ser **I**nterface
IC	**I**nitial **C**ondition
IVA	**I**nitial **V**alue **A**nalysis
OD	**O**ptical **D**ensity
ODE	**O**rdinary **D**ifferential **E**quation
PA	**P**arameter **A**nalysis
PDE	**P**artial **D**ifferential **E**quation
RNA	**R**ibo**N**ucleic **A**cid
ST	**S**erial **T**ransfer
UA	**U**ltimate **A**nalysis
UvA	**U**niversitiet **v**an **A**msterdam

Chapter 1

Introduction

Phages are small viruses, measuring 27-190 nm in diameter, that infect and lyse (kill) specific bacteria. Phages act as nature's antimicrobial defense, but they also impact bacterial evolution and resource turnover. There are various medical and industrial applications for phages to control bacterial growth; however, to realize these applications, it is essential to understand how phages interact with bacteria, enabling the implementation of a robust method to control bacterial growth.

1.1 Thesis Overview

I developed a simulation framework that can model and interact with any $p \times b \times r$ system, where p represents the number of phages, b represents the number of bacteria, and r represents the number of resources. This framework enables the mathematical description of these components and their interactions. Using the software, I answer how phages impact community dynamics in complex microbial communities.

First, there is a biological introduction to phages and bacteria. This introduction covers how phages infect bacteria, how bacteria defend against phages, how phages defeat bacterial defenses, and how phages defend against other phages. There is an introduction to different methods of modeling phage and bacteria dynamics. This thesis provides a brief overview of software that models phages, resources, bacteria, and their associated limitations. I present the simulation framework that I developed to support the research, demonstrating its capabilities using a representative model of phage, bacteria, and resource interactions. The section also provides an overview of its usage, including example outputs from demonstration runs. I use the software to analyze various scenarios, such as phage proliferation under a washout scenario, and analyze growth rate-limited regions.

1.2 The Environment

In ecosystems like the ocean, the gut, or soil, there are thousands of different microbes that interact with one another or with their surrounding environment. The interactions are complex, with many factors affecting the growth of bacteria and phages. The interactions between entities in the environment are often synergistic. When an animal dies, bacteria begin to digest and decompose the animal into simpler chemicals, such as carbon and nitrogen, which plants can use to grow. Animals then eat these plants, and the cycle continues. External factors, such as flooding, droughts, chemical spills, or the introduction of new entities, have a massive impact on the ecosystem. These events can add or remove resources from the system, alter environmental parameters such as the temperature and acidity of the soil, introduce competition, or create a population imbalance by killing microorganisms. These effects have a larger effect on the ecosystem and food chain as a whole, as bacteria are one of the fundamental foundations for resource recycling.

1.3 Biological Background

Phages are small viruses on the order of 27-190 nm (the average size of marine phages is 54 nm) that infect and lyse (kill) specific bacteria. The phage cycle process starts with a phage coming into contact with a bacterium Figure 1.1. Once it has identified an injection site, the phage can inject a strain of DNA into the bacteria. The DNA strand has two options.

The first option is that the phage can enter the lysogenic cycle by integrating its DNA into the host cell's DNA. Prophages are phages that have integrated with the host cell's DNA. The new cell will contain the prophage's DNA as the bacteria replicate. The phage will exit the lysogenic cycle and enter the lytic cycle after receiving a signal, such as cell damage, and hijack the DNA replication mechanism.

The second option is that the phage immediately enters the lytic cycle by hijacking the DNA replication process to create copies of its DNA. The phage will create multiple copies of itself using the host's transcription and translation machinery to replicate itself. The phages self-assemble inside the bacteria until they lyse the cell (e.g., through chemically induced rupture of the cell wall), releasing the phages into the environment [2].

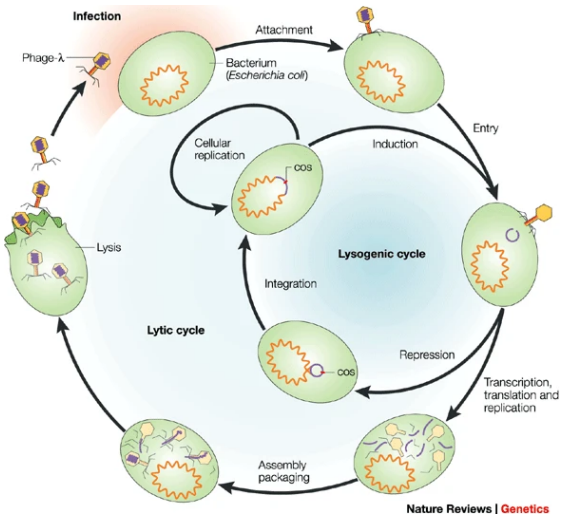


Figure 1.1: Life cycle of a phage, inside and outside a bacteria cell. Significant steps in the life cycle of a phage include the infection stage, integration, replication, and lysing process. Figure sourced from Campbell [2].

1.3.1 Phage's Role in the Environment

Phages play a significant role in the ecosystem. Phage-mediated death occurs when phages infect susceptible bacteria, resulting in cell lysis and the release of cellular contents and nutrients that other organisms, such as bacteria and plants, can utilize. This process not only reduces bacterial populations but also accelerates the turnover of resources, such as nitrogen and carbon, for other bacteria and plants to utilize. Phages also help mediate horizontal gene transfer, disperse pathogenic diseases, and spread antibiotic resistance [10]. Phages directly alter bacteria population diversity and population fitness by introducing new ways for bacteria to mutate [11].

There are about 10^6 bacteria cells/ml and 10^7 phages/ml of marine water. About 5% of any bacteria are currently infected, and phages account for about 15% of daily bacterial deaths [12]. Phage populations grow by infecting their hosts, but they can also be degraded, e.g., by UV radiation. UV is a significant factor in deactivating marine phages, causing up to a 5% reduction in phage infectivity per hour [12].

1.3.1.1 Phages and Controlling Bacterial Blooms

Phages can only kill a specific bacteria. The bacteria that a phage can kill depends on the receptors of the bacteria that the phage can detect. They have the potential to be used by humans to control bacterial infections in patients and large-scale environmental bioengineering efforts by controlling *Cyanobacteria* populations in lakes and rivers.

Cyanobacteria cause damage to aquatic life by consuming resources and oxygen, starving aquatic life, and negatively affecting human health. Scientists are investigating the potential use of phages to control cyanobacteria (also known as blue-green algae) blooms in the environment [13]. There is hope that phages can biologically control water quality in wastewater treatment plants and in the environment without the use of harsh chemical processes that would otherwise pose environmental and health hazards [4, 14]. Appendix B.3 contains more information about controlling *Cyanobacteria*.

1.4 Phage Cocktails and Human Health

There is particular interest in phage applications in human and animal health, called phage therapy. There are approximately 100 trillion microbes, comprising about 5,000 different types of bacterial strains, in the human gut. The phages will target the specific bacteria of interest, for example, *E. coli*, but they will not affect the other bacteria found in the human gut. Antibiotics indiscriminately affect any bacteria, disrupting the intricate ecosystem of the gut microbiome, acting as a scorched-earth mechanism. Antibiotics have also faced the challenge that bacteria are growing resistant to them, making the antibiotics less effective in the future [15, 16].

Phages, on the other hand, target a specific bacterial strain. Antibiotic-resistant bacteria are typically less resistant to phages. The bacteria cell typically has a trade-off between antibiotic resistance and phage resistance. So by designing phages to be highly infective, there is hope that the phage-resistant bacteria will lose the antibiotic resistance to counter the phages [17, 18]. Appendix B.2 offers a more in-depth look at how healthcare professionals can utilize phages.

1.5 Potential Industrial Applications of Phages

Phages have numerous applications in industrial settings. Phage therapies can prevent the spread of common bacteria in livestock by incorporating phage therapy into the animal feed. Farmers often raise livestock in cramped spaces with inadequate sanitation facilities, which increases the risk of disease spreading. Factories can utilize phages to control the growth of bacteria, such as *Salmonella* while producing food [7, 19]. Appendix B.1 in Appendix B goes into more detail about using phages to control foodborne bacteria.

1.6 Modelling Phages in a Complex Community

What we know about phages mechanistically often comes from well-researched bacteria in a laboratory, such as *E. coli* and its phages, and what we know about phages in the environment comes from metagenomic surveys. Making the connection between the mechanistic and metagenomic models, which would enable us to develop and test different models, is the challenging part. Because of this, we need mathematical models to help bridge the gap between the lab and the environment. There have been previous attempts to model the complex dynamics of populations between phages, bacteria, and resources within the environment using Ordinary Differential Equations (ODEs) and Delay Differential Equations (DDEs). Researchers cannot identify every interaction in the complex community, and even if they identify an interaction, they must experimentally derive the associated parameter values.

There are two primary methods for modeling phage-bacteria dynamics: spatial and non-spatial. In a spatial model, phages and bacteria can move through space and interact with their neighbors. Researchers have utilized partial differential equations (PDEs) and cellular agent-based models (ABMs) to model spatial interactions. Spatial models lead to more computationally complex models but can result in more biologically realistic results. By contrast, ODE and DDE models describe non-spatial models. In a non-spatial model, a well-mixed solution contains the bacteria and phages, and researchers make no distinctions regarding neighbors or distances to other entities. They simplify interactions with a probabilistic approach. At time step t, a percentage p of entities interact with each other.

Non-spatial models are easier to develop and understand and are more effective in modeling large populations, albeit at the cost of losing spatial information. For this thesis, the focus will be on modeling resource, phage, and bacteria interactions using an ODE model. I describe a phage-bacteria-resource system as a $p \times b \times r$ system. Current modeling methods have mainly stayed with $1 \times 1 \times 1$ models, meaning one phage, one bacteria, and one resource. This thesis aims to develop a simulation framework that can model any $p \times b \times r$ ODE system.

1.7 Software Overview

The project contains three parts, with an optional fourth part. In the first part, the user creates a network interaction. Here, the user can define the number of resources, phages, and bacteria that interact with each other, as well as the parameter names and values. See Section 3.1.1 for further information. In the second part (Section 3.1.2),

the user sends the network model and parameters to the framework. The third section (Section 3.1.3) allows the user to interact with the graph structure and Section 3.1.2 with a dashboard, but it is also possible to do this from a Python file, skipping the dashboard. The user sends the simulation parameter data to the solver and receives the time data and population data as an array as the output. The user can easily edit the parameter values of the network's edges and nodes and run more advanced visualizations, such as changing a parameter value to observe its impact on the population count from the dashboard.

Several plots are included out of the box, allowing the user to test them. The plots offered in Part 3 of the program provide interactivity, including the ability to hide and show lines and dots, zoom in and out, and hover over lines and dots to display more detailed data.

Finally, the user can optionally run multiple simulations and download the data to their disk to create their own visualizations using Section 3.1.4. The visualizations created in Section 3.1.3 can be recreated in Section 3.1.4. The user can choose the same parameter values used for a specific plot in Section 3.1.3, run the simulation (as described in Section 3.1.3.2), download the data, and reimplement the graphs.

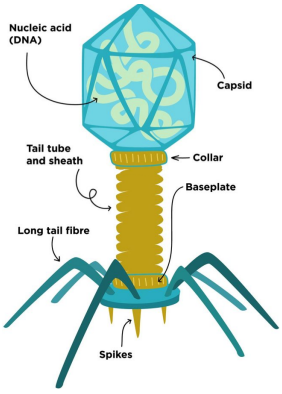
Chapter 2

Literature review

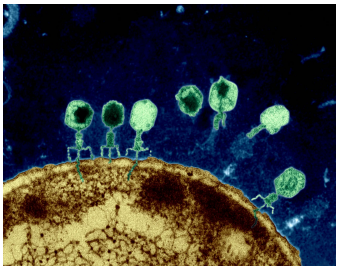
2.1 Phage Biology

2.1.1 What Are Phages?

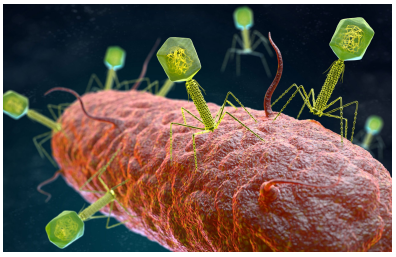
Phages are small bundles of proteins that contain viral DNA. Phages are composed of multiple parts, built like LEGO blocks, to complete the task of infecting a bacterium. Figure 2.1a shows the body parts of a phage. The phage aims to find a suitable bacterial host and infect the host with viral DNA. The DNA alters the host's metabolic pathways to its benefit and hijacks the cellular replication process to create new copies of the phage. Eventually, the cell lyses, releasing the newly created phages into the environment to infect more bacteria.



(a) Phage body structure. Image sourced from Fla [20].



(b) Phages infecting an *E. coli* bacteria. Image sourced from Twilley [21].



(c) Artist representation of phages infecting a bacterium. Image sourced from Vir [22]

Figure 2.1: Parts of a phage, a real-life picture of phages infecting an *E. coli* bacterium, and an artist's impression of phages infecting a bacterium.

2.1.2 How Does the Phage Cycle Work?

There are three main parts to the phage-bacteria host cycle: the infection stage, the lysogenic cycle, and the lytic cycle. Figure 1.1 shows an overview of the phage cycle.

In the infection stage, a phage attaches to the surface of a bacteria cell. The infection stage involves the phage searching for, detecting, and attaching to a bacterium, followed by the injection of its DNA into the bacterium. Detection and attachment occur via phage receptor-binding proteins located at the tip of the phage tail that recognize specific receptors on the bacterial cell wall, triggering conformational changes that enable DNA injection [23]. The success of this process depends on the specificity and density of both phage and bacterial receptors [24]. Once attached, the phage injects its DNA into the host cytoplasm, where it can replicate independently. Once injected, the phage-cell pair enters either the lysogenic cycle or the lytic cycle.

The lysogenic cycle involves phage DNA integrating into the bacterial genome as a prophage, where it is replicated along with the host cell without causing immediate lysis. The phage evades host defenses such as CBASS and CRISPR-Cas systems, which can initiate programmed cell death, preventing phage replication or detect and degrade foreign DNA [25, 26]. Programmed cell death helps recycle resources for other bacteria [27]. Once integrated, the prophage can alter host fitness and provide resistance to other phages. During cell division, the prophage is copied into daughter cells but remains at risk of being excised by restriction enzymes [28]. Under certain stress conditions, such as DNA damage or activation of the SOS response, the prophage induces itself to exit the genome and enter the lytic cycle [24, 29, 30].

The lytic cycle is the process where a phage infects a bacterium, hijacks its replication machinery to produce new phage components, assembles these parts, and ultimately lyses the host cell to release new phages. This process involves hijacking the host's DNA replication to synthesize phage parts like the capsid, sheath, and tail (Figure 2.1a). The phage does this by redirecting resources from internal cellular functions towards viral replication [27]. The parts self-assemble via protein-protein and protein-nucleic acid interactions [31]. Phages induce bacterial lysis by producing holin proteins that disrupt the cell membrane, releasing the phages and resources [32].

2.2 Bacterial Defense Against Phages

There is a constant battle between phages and bacteria. The bacteria do not want to be killed by the phages, so they develop defenses such as thickening their cell walls or destroying the viral DNA.

2.2.1 Mutations in Bacterial DNA (Genetic (Co-)Evolution)

As bacteria cells grow and divide, random point mutations can occur in the DNA. These mutations can affect phage defenses, like thickening the cell wall or removing a receptor, making it harder for the phages to detect and infect the cell. Mutations can be partially effective if full effectiveness requires multiple steps to achieve. Random mutations can also fail to make the bacteria more resistant to phages by increasing phage susceptibility or by incurring a cost to the bacterial cell, such as losing receptors on the cell wall [33].

2.2.2 Horizontally Transferring DNA

Bacteria can horizontally transfer DNA to other bacteria on contact. A donor cell can donate DNA fragments using a mechanism called a pilus. The pilus acts as a tunnel between the donor cell and the recipient cell so that DNA can transfer from the donor cell to the receiver cell [34].

A phage can accidentally collect a piece of the host's DNA instead of its DNA during assembly. The phage, with the now dead host's DNA, can infect the next bacterium, injecting the new bacterium with the dead cell's DNA, thereby horizontally transferring the DNA [35, 36]. The transferred DNA can include natural phage defenses or alter the bacterium's genes and phenotype, making it undetectable to future phages.

2.2.3 Phage Inactivation and Decoys

Bacteria can further protect themselves by producing decoys that the phage will attach to rather than themselves. Freshly lysed bacteria may still have biomarkers that attract phages, leading phages to attach to non-viable cells where successful infection cannot occur. Bacteria can also produce proteolytic enzymes that will damage the proteins found in a phage [37]. Some bacteria can produce outer membrane vesicles that phages can absorb to, and later detach and float away with the phage [38]. These vesicles are suspected of having a minor impact as a sink [39]. Dey et al. [40] showed that these vesicles, a "debris" term for the phages, had a positive effect on coexistence in large phage-bacteria communities.

2.2.4 Phenotype Resistance

Not all new phenotypes arise from genetic mutations. Resistance can result from phenotypic variation within a genetically identical population, allowing bacteria to express

different resistance traits without altering their DNA. Gupta et al. [41] found that some *Bacteroides fragilis* bacteria were able to evade phage infection. The presence of combinatorial phenotypic states, where the differential expression of protective mechanisms created rare, super-resistant cells capable of withstanding phage attack. By acting together, these heterogeneously expressed anti-phage defense mechanisms created a phenotypic landscape where distinct protective combinations enabled the survival and regrowth of bacteria expressing these phenotypes without acquiring additional mutations [41].

2.2.5 Spatial Refuge/Biofilms

Bacteria can evade phages by forming spatial refuges, such as biofilms, or hide behind physical structures. Biofilms are dense microbial communities embedded in mucus, which impedes phage diffusion and protects resident bacteria [42]. On agar plates, spatial structure similarly limits phage spread [43]. While phages rely on passive diffusion, bacteria can actively move, further enhancing their survival [44].

2.3 Phage Counter Defense Against Bacteria

With some of the defenses that bacteria have developed, phages are constantly mutating to counter their defenses. If phages do not adapt to the ever-changing bacterial defenses, the phages will die out due to their inability to infect and multiply. It becomes an arms race, with each side trying to out-adapt the other. The bacteria and phages must have a delicate balance to ensure coexistence.

2.3.1 Genetic Mutations

Mutations in viral DNA will affect how the phage body parts are designed and built. These mutations will affect both external phage behavior, such as how it detects a bacterium, and internal behavior, including evading detection and integrating with the cell's DNA. The changes will lead to changes in overall phage fitness, i.e., the ability of the phage to infect, replicate, and lyse bacteria.

2.3.2 Viral Recombination

Multiple phages can infect a cell and replicate itself using the cell's internal replication process. Each phage has unique building blocks, including the capsid, tail, and sheath.

If the proteins that build the subparts of each phage have similar chemical properties, the phage parts can be swapped between phages [31]. The swapping of parts allows for biological diversity to spread throughout a phage population. Each phage body part can have unique characteristics, such as a higher attachment rate, a larger DNA storage capsule, or a better probability of injection.

2.4 Phage Defense Against Phages

Some phages can employ defenses against other phages from infecting the bacterial cell, ensuring that the host's resources are all for itself. The act of preventing a secondary infection from a similar or closely related phage is called superinfection exclusion (SIE) [45]. The following are several methods for preventing further infections.

2.4.1 Altering Cell Structure

The prophage can alter the surface receptors of the bacteria, making it harder for other phages to detect the bacteria, reducing the chance of attachment and injection by other phages [46].

2.4.2 Protein Creation

Other phages, like the T4 phage, can create proteins like the Spackle protein, which inhibits the lysozyme activity used in the process of DNA injection by other phages [46, 47]. Some prophages can encode proteins that will interfere with the replication process of other phages. For example, the SieA protein encoded by phage P22 blocks infection from other phages [48].

Tail Assembly Blocker (TAB) is an anti-phage defense mechanism encoded by a *Pseudomonas aeruginosa* prophage. While TAB permits the invading phage to replicate its genome, it inhibits the assembly of the phage tail, thereby preventing the production of infectious virions. The prophage that encodes TAB is not affected by this inhibition, as it also expresses a protein that neutralizes TAB's blocking activity. Although the host cell still undergoes lysis, no infectious phages are released.

2.5 Bacteria and Phages in the Lab

Researchers worldwide are conducting laboratory experiments to gain a deeper understanding of the interactions between phages and bacteria. The aim is to gain a better understanding of how phages work and interact with bacteria at the molecular, host, and population levels.

2.5.1 Running Experiments

Laboratory experiments are often conducted in a liquid medium containing the necessary resources for bacterial growth. This liquid medium, often referred to as broth, facilitates the cultivation of bacteria in a well-mixed environment, allowing researchers to monitor bacterial growth and phage infection dynamics over time. By adjusting parameters such as resource concentration, temperature, and pH, researchers can simulate different environmental conditions and observe their effects on phage-bacteria interactions.

Samples are collected at set time points to measure bacterial density, phage titer, and resource concentration. These data enable researchers to fit mathematical models, estimate growth rates, and determine phage parameters, such as latent time and burst size, from one-step growth curves [1, 6].

2.5.2 Chemostats

Commonly used setups include those containing liquids with phages, bacteria, and resources in a chemostat and in batch culture. Chemostats enable the continuous addition of resources and removal of waste, thereby maintaining steady-state conditions that are ideal for studying long-term dynamics.

2.5.3 Petri Dishes

Petri dishes are another common method used to grow bacterial colonies. Agar, a jelly-like substance derived from seaweed, is commonly added to create a solid growth medium in Petri dishes. Agar provides a stable surface for bacteria to grow on and form visible colonies. Clear zones, known as plaques, appear where phages have infected and lysed the bacteria, enabling the quantification and observation of phage activity. The phages can diffuse on the agar plate, infecting neighboring cells. Phage infection creates clear plaques (2-3 mm) where bacteria are absent. Figure 2.2 shows an example of a bacteria lawn with phage plaques.

2.5.4 Measuring Growth

Bacterial density in liquid media is usually measured by optical density (OD) using a spectrophotometer. OD measurements require calibration and are not directly comparable across devices [49]. Using cell counts (e.g., $cells/ml$) allows for more consistent comparisons across experiments and labs [50].

With Petri dishes, it is more challenging to measure bacterial growth and plaque size accurately. It may be possible to wash the bacteria off into a test tube with water to measure the optical density, but the results are inconsistent. It may be possible to quantify the change in plaque size using an image analysis program; however, the results may be inaccurate and sensitive to variations in lighting conditions.

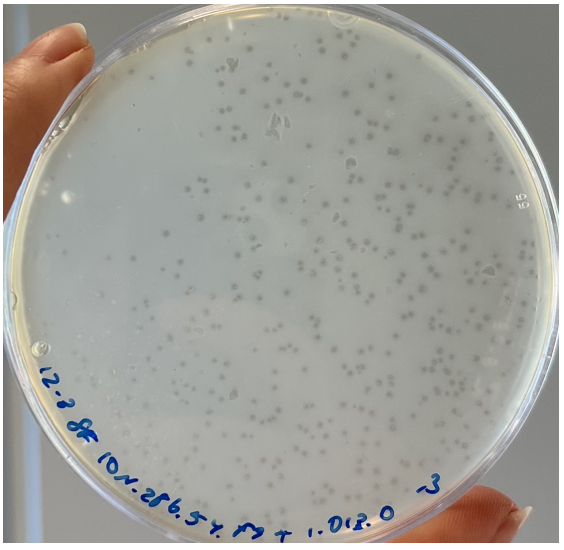


Figure 2.2: Bacteria lawn, the dots on the petri dish show no bacteria growth due to the presence of phages. Photo courtesy of S. Flickinger.

2.5.5 Serial Transfer

Serial transfer (ST) is a method employed by a bacteriologist, where, after a set amount of time, the bacteriologist pipettes medium containing phages, bacteria, and resources out of a test tube and adds the old medium to a new test tube with fresh medium. At this stage, the bacteriologist can add more bacteria or phages to the test tube. Only resources are typically added during the transfer process. Researchers can optically measure bacterial density using an optical density meter or employ a mass spectrometer to determine phage concentration at set time points during the experiment. As the bacteria grow, they consume the resources found in the medium. The resources will eventually run out, and the bacteria die out due to a lack of resources. By introducing

new resources at set time intervals, the bacteria can regrow and exhibit a semi-stationary behavior.

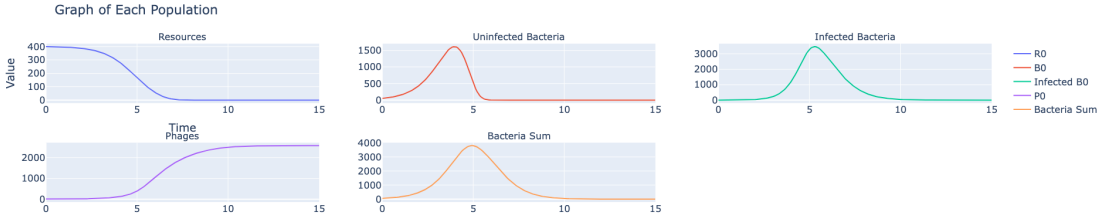
2.5.6 Growth Curves Typically Seen in a Lab

When choosing model parameter values, it is essential to select values that can realistically be found in real-life systems and be replicated in the lab. Various features characterize a growth curve produced in a lab that a model should aim to reproduce.

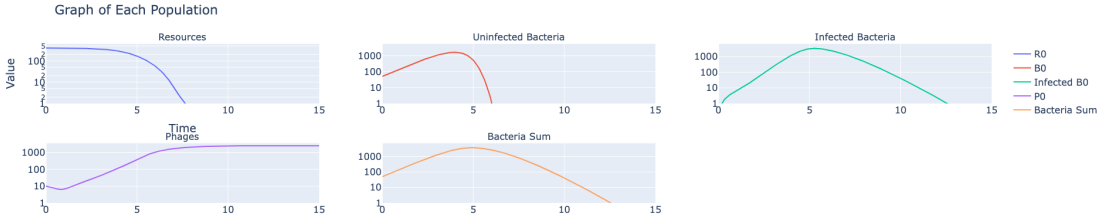
The idealized dynamics of bacterial populations undergoing phage infection have several phases. First, there is an apparent exponential rise in bacteria growth, growing 40-100x in population in the span of a few hours. At a certain point in time, the bacterial population starts decreasing almost as fast as it was growing.

Phage populations also exhibit exponential growth but with a delay in growth relative to the bacterial population. Initially, there was no growth in the phage population. After a set amount of time, the phage population will start to grow and peak a few hours after the bacteria population reaches its peak. If there is no phage death or removal, the phage population will eventually reach a plateau when every bacteria has died.

Figure 2.3a shows an example of a curve for a $1 \times 1 \times 1$ system that would typically be seen in a lab. Figure 2.3b is the same plot but with a logarithmic y-axis. These specific plots exhibit a clear growth, peak, delay, and death cycle.



(a) An example linear y-axis for a curve that researchers aim to replicate.



(b) The equivalent logarithmic y-axis plot for a curve that researchers aim to replicate.

Figure 2.3: Growth of a population in a $1 \times 1 \times 1$ system. The log plot allows us to visualize behavior at values approaching zero and to plot data on a logarithmic scale. The parameters used for this plot can be found in Table E.1.

2.6 Software Mathematically Modelling Phages, Bacteria, and Resources

Some software programs modeling phage-bacteria-resource interactions already exist. Here, I cover three software programs, Cocktail, PhageDyn, and PhiSite, that have been created to model phages and bacteria, as well as their limitations.

2.6.1 Cocktail

Nilsson [3] developed Cocktail to model phage-bacteria-resource kinetics in a chemostat. The model assumes that phage A, phage B, and the combination of both phages (phage AB) can simultaneously infect a single bacterial strain. The model simulates bacterial resistance to phage A and phage B, as well as a combination of phage A and B. The user can control parameter values, such as resistance rates to A, B, and AB, resource concentration and outflow, and phage adsorption rate. The user can also control model settings, such as if the model is deterministic or stochastic, and the step size [3]. Figure 2.4a shows four sample output plots.

2.6.2 PhageDyn

PhageDyn, developed by [4], is a Java applet that models phage dynamics in multi-reactor industrial wastewater treatment plant models. PhageDyn interacts with existing GPS-X [51] files to incorporate phage dynamics into models of industrial wastewater treatment plants. Krysiak-Baltyn et al. [4] developed PhageDyn to determine how phages can reduce foaming caused by bacteria in wastewater treatment plants, another real-life application of phages [52]. PhageDyn does not simulate phage dynamics on its own; instead, it manipulates existing files in GPS-X to incorporate phage dynamics into wastewater treatment plant models. Figure 2.4b shows the output that PhageDyn provides.

2.6.3 PhiSite

PhiSite is a website created by Beke et al. [5] to allow users to interact with the Schrag and Mittler [53] model. The website can be accessed here. It is possible to adjust various input parameters, such as the temperature and pH of the system, the infected bacteria count, washout, and more. It is possible to download the graphs and raw simulation data from the dashboard. See Section 2.7.3 for more details on the model. Figure 2.4c shows the output that PhiSite provides.

2.6.4 Cocktail, PhageDyn, and PhiSite Limitations

There are limitations to all three software. Cocktail can model up to a $2 \times 1 \times 1$ system and in a chemostat environment. Chemostats receive a constant influx of new resources and a constant removal of medium from the chemostat.

Cocktail's ODE model is not readily adaptable to other model equations. The ODE model accepts inputs from a hardcoded GUI front end. Any changes to the front end or the ODE model will require corresponding changes to both the ODE model and the front end to accommodate the new inputs and outputs. The code for Cocktail is open source, so adding new buttons and modifying the model should not pose a significant challenge, but it is still a considerable undertaking.

PhageDyn is used with GPS-X, a highly specialized wastewater treatment modeling software. PhageDyn is programmed for a specific task, with no flexibility in modifying the model or inputs. PhageDyn assumes biomass instead of individual bacteria populations. However, PhageDyn is no longer available for download.

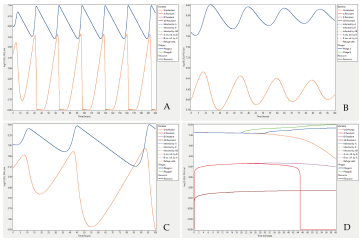
PhiSite faces similar limitations to Cocktail, where the model input is hard coded to the front end, making it difficult to change the ODE model inputs and challenging to run multiple simulations simultaneously to cover a wide range of parameter values. There is no built-in method to programmatically iterate over a set of values and download the data. The data needs to be downloaded individually and then combined into a graph.

2.7 Methods of Modeling Phages and Bacteria

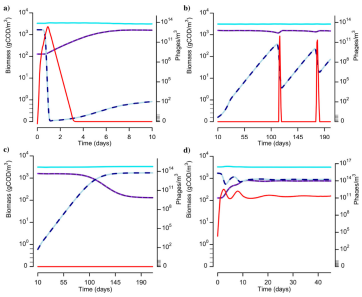
There are various ways to model phage, bacterial, and resource populations and their interactions; however, the most common approach is to use Ordinary Differential Equations (ODEs) or Delay Differential Equations (DDEs).

The ODE method is simple to understand and easy to set up, but it can only capture large population dynamics. Certain assumptions about community interactions also need to be made, such as the assumption that everything is probabilistic. For example, every timestep, α percentage of phages interact with bacteria DDEs are similar to ODEs, but DDEs incorporate time delays to account for processes that depend not only on the current state but also on past states, thereby incorporating behavior that has a delay, such as latent infection time.

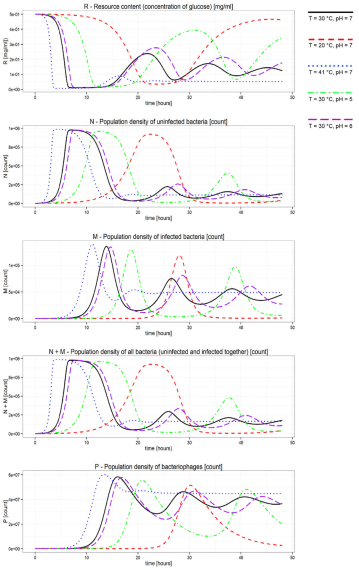
One way to introduce a delay in the infected bacteria releasing phages is to force the infected bacteria to undergo a series of stages. Once the infected bacterium has progressed



(a) Figure A) *E. coli* infected with phage T4 in a chemostat exhibiting an oscillating growth behavior, following the model of Bohannan and Lenski [54]. Figure B) Oscillations of bacteria and phages can exist at higher titers, dependent on low resource concentration, following the model of Lenski [55]. Figure C) As the concentration of resources change, this results in increasing oscillations, but not going extinct. Figure D) A system modeling the interactions with phage A and B. Figure and caption sourced from Nilsson [3].



(b) **Purple** is heterotrophic biomass, **Blue** is foaming biomass, **Red** is phages, **Light Blue** is total suspended solids. Figure A) Biomass concentration immediately post phage dosing. Figure B) Biomass concentration with low phage concentration and maintain low concentration post spike in population count. Figure C) Biomass concentration when phages are extinct. Figure D) Biomass concentration with a less virulent and low adsorption rate phage, indicating coexistence with biomass. A change in phage concentration shows a decrease in heterotrophic and foaming biomass [4]. Figure and caption sourced from [4]



(c) Running the simulations under five different conditions. The simulations have been run under different conditions to demonstrate the capabilities of the designed model. To set up the simulation the values of maximal, minimal and the optimal pH and temperature have to be experimentally stipulated. Then the model can calculate specific growth rate under different pH and temperature. Figure and caption sourced from Beke et al. [5]

Figure 2.4: Example output from Cocktail, PhageDyn, and PhiSite, respectively. For PhageDyn, the concentration of heterotrophic biomass in an aerobic plug flows across four situations. See Nilsson [3], Krysiak-Baltyn et al. [4], and Beke et al. [5] for more information on parameter values and supplementary resources.

through every stage of infection, it releases the phages into the system. For example, in the paper Geng et al. [1], infected bacteria go through M stages of infection before lysing. By decreasing τ (the latent period) in the model proposed by Geng et al. [1], more infected bacteria go from infected state i to infected state $i+1$ per timestep with rate $\frac{M}{\tau}$, causing the infected peak population count to peak earlier.

Each model can be further developed, for example, by adding temperature and pH dependence, bacteria releasing nutrients upon lysis, or phage resistance.

2.7.1 Campbell DDE Model for Chemostats

The Campbell [56] model is a DDE model that describes the growth of bacteria and phages in a chemostat, a bioreactor used to cultivate microorganisms. Fresh medium is

constantly added, and liquid culture is constantly removed at a constant rate to maintain a constant culture volume. The following equation describes the growth of bacteria B and phage P.

$$\frac{dB_i}{dt} = k_B B\left(1 - \frac{B}{L}\right) - \alpha B - k_A P B$$
$$\frac{dP}{dt} = k_A N\left[B(t-\tau)P(t-\tau)\right] - k_A B P - k_I P - \alpha P$$

α is the inflow and outflow rate, k_A is the adsorption rate, k_I is the rate of spontaneous inactivation of phages, k_B is the bacterial growth rate following a logistic growth rate, with max level L, N is the burst size after τ time units, and t is the time. $B(t-\tau)$ and $P(t-\tau)$ is the population of the bacteria and phages at time $t-\tau$.

Steady states occur when $\frac{dB}{dt} = \frac{d0}{dt} = 0$, and there are four steady-state solutions.

1. $B = 0, P = 0$: Both bacteria and phages are absent.
2. $B = 0, k_I + \alpha = 0, P = P_0$: Bacteria are absent, and phages persist only if the inactivation and outflow rates sum to zero.
3. $B = L\left(1 - \frac{\alpha}{k_B}\right), P = 0$: Bacteria reach their carrying capacity in the absence of phages.
4. $B = \frac{k_I+\alpha}{k_A(N-1)}$, $P = \frac{k_B}{Lk_A}\left[L\left(1 - \frac{\alpha}{k_B}\right) - \frac{k_I+\alpha}{k_A(N-1)}\right]$: Coexistence of bacteria and phages at steady state.

Solutions 3 and 4 happen when the initial populations are not zero and the flow rate or the spontaneous phage inactivation rate is greater than 0.

2.7.2 Weitz ODE Model for Coevolutionary Arms Race

Bacteria that adapt to avoid phage infection incur a cost in cellular functions, such as resource uptake. Phages that adapt to infect new bacteria may lose their ability to infect other bacteria. As phages and bacteria evolve, the phage's ability to adsorb to bacteria and the bacteria's ability to consume resources change. The key is this trade-off, which promotes diversification in phage and bacteria populations even in a single-resource, homogeneous environment [57]. The Weitz et al. [57] model describes how to model selective pressure and trait adaptations in phage and bacterial populations and how these adaptations affect the dynamics of phage and bacterial populations. The receptors on the bacteria's cell wall and phage's tail act as a key-lock model, so any

changes to the receptors will prevent detection and docking. The Weitz model, shown below, simplifies the traits into one number, x_i for bacteria i and y_j for phage j.

$$\begin{aligned}
\frac{dR}{dt} &= -\omega(R - R_0) - \sum_i \epsilon\gamma(x_i)\frac{RB_i}{R+K} \\
\frac{dB_i}{dt} &= -\omega B_i + \gamma(x_i)\frac{RB_i}{R+K} - \sum_j \phi(x_i, y_j)B_iP_j \\
\frac{dP_i}{dt} &= -\omega P_i + \sum_j \beta\phi(x_j, y_i)B_jP_i \\
\gamma(x_i) &= \gamma_0 e^{-\frac{(x-x_0)^2}{2\xi_n^2}} \\
\phi(x_i, y_j) &= \phi^{-\frac{(x_i-y_j)^2}{2\xi_v^2}}
\end{aligned}$$

ω is the washin rate, R_0 is the initial resource concentration, x_i is the trait value for bacteria i, and y_j is the trait value for phage j, $\gamma(x_i)$ is the resource consumption rate dependent on trait x_i, $\phi(x_i, y_j)$ is the adsorption rate between trait x_i and trait y_j. K is the half saturation Monod constant, β is the burst size of a phage, ξ_n is the stable uptake range of hosts, and ξ_v is the host range of phages.

Specifically, ξ_n is the range of possible host phenotypes whose maximal growth rate is within $e^{-\frac{1}{2}}$ of the maximum for all phenotypes. ξ_v is the range of possible host phenotypes for which any given phage has an adsorption rate within $e^{-\frac{1}{2}}$ of its maximal adsorption rate. K and β can also be considered traits of the bacteria and phage, but the authors decided to hold them constant for mathematical tractability. $\gamma(x_i)$ implies that there is an optimal configuration for maximal resource uptake when $x_i = x_0$ and an opportunity for a trade-off between resource uptake and phage avoidance. $\phi(x_i, y_j)$ suggests that for every bacteria trait x_i, there is a phage trait y_j that maximizes the adsorption rate.

2.7.3 Schrag DDE Model for a Chemostat Including Temperature and pH

The Beke et al. [5], Schrag and Mittler [53] model below describes a phage and bacteria population in a chemostat environment with dependence on temperature and pH. Temperature and pH affect the growth of phages and bacteria, influencing the dynamics of the system. Changing the temperature or pH of the system away from its optimal values will result in suboptimal growth. It should be noted that, for example, the T_{min} value

is not the lowest known temperature at which growth can occur but rather an extrapolated value. Likewise for $T_{max}, pH_{min}, pH_{max}$. When T or pH is near those values, the maximal growth rate μ_{max} will be close to zero. Thus, the system will show barely any growth [5].

$$
\begin{aligned}
\frac{dR}{dt} &= \omega(R_0 - R) - \frac{\varepsilon \mu_{\max} R(U+I)}{K+R} \\
\frac{dU}{dt} &= \frac{\mu_{\max} RU}{K+R} - \delta UP - \omega U \\
\frac{dI}{dt} &= \delta UP - \delta U(t-\tau)P(t-\tau)e^{-\omega\tau} - \omega I \\
\frac{dP}{dt} &= \beta\delta U(t-\tau)P(t-\tau)e^{-\omega\tau} - \delta UP - \delta IP - \omega P \\
\mu_{\max} &= \mu_{\text{opt}} \cdot \tau(T) \cdot \rho(pH) \\
\tau(T) &= \frac{(T - T_{\max})(T - T_{\min})^2}{(T_{\text{opt}} - T_{\min})\left[(T_{\text{opt}} - T_{\min})(T - T_{\text{opt}}) - (T_{\text{opt}} - T_{\max})\right]} \\
\rho(pH) &= \frac{(pH - pH \cdot T_{\min})(pH - pH \cdot T_{\max})}{(pH - pH_{\min})(pH - pH_{\max}) - (pH - pH_{\text{opt}})^2}
\end{aligned}
$$

R is the resource concentration, R_0 is the resource concentration in the reservoir, U is bacteria population density, I is the population density of infected bacteria, and P is phage population. t is the time of the simulation, ω is the flow rate, μ_{max} is the maximal growth rate, μ_{opt} is the optimal growth rate, ϵ is the growth efficiency, K is the Monod constant for half growth rate of μ_{max}, β is burst size of the phages, δ is the adsorption rate of phages to bacteria, τ is the latent period. Adding the environment parameters, T is the temperature of the system, $T_{opt}, T_{min}, T_{max}$ is the optimal, minimal, and maximal temperature of the bacteria, pH is the pH of the system, $pH_{opt}, pH_{min}, pH_{max}$ is the optimal, minimal, and maximal pH of the bacteria.

2.7.4 The Golding Model

The "Golding model" will be the default ODE model used in this report, as described by Geng et al. [1]. The model describes the interactions between resources, uninfected bacteria, infected bacteria, and phages.

2.7.4.1 The Original Golding Model

The original Golding model (Equation (2.1)) describes three biological processes: cell consumption of resources and growth, phage/cell encounters and infection, and cell lysis.

$$
\begin{aligned}
\frac{dR}{dt} &= -e \cdot g(R, v, K) \cdot (U + \sum_{k=1}^{M} I_k) \\
\frac{dU}{dt} &= g(R, v, K) \cdot U - r \cdot U \cdot P \\
\frac{dI_1}{dt} &= r \cdot U \cdot P - \frac{M}{\tau} \cdot I_1 \\
\frac{dI_k}{dt} &= \frac{M}{\tau}(I_{k-1} - I_k) \text{ for } k = 2, \ldots, M \\
\frac{dP}{dt} &= \beta \cdot \frac{M}{\tau} \cdot I_M - r \cdot (U + \sum_{k=1}^{M} I_k) \cdot P \\
g(R, v, K) &= \frac{v \cdot R}{R + K}
\end{aligned}
\tag{2.1}
$$

Once infected by a phage, the bacteria goes from U to I_1. The bacteria go through M stages of infection $I_1, \ldots, I_M$ before lysing, The bacteria goes from state I_k to state I_{k+1} with equal transition rate $\frac{M}{\tau}$. The infection rate of a cell is r. After a bacteria lyses after stage I_M, β phages are released, the burst size of the phage. $g(R, v, K)$ described the cell growth process, the instantaneous growth rate dependent on the Monod equation, where v is the maximal growth rate of the bacteria population, and K is the Monod constant. Bacteria consume a resource at a rate of e. The probability of phage p infecting bacteria b is r and is not to be confused with the resource concentration R. A summary of the parameter values can be found in Table A.1.

2.7.4.2 The Generalized Golding Model

The original Golding model simulates a $1 \times 1 \times 1$ system. To generalize this model to fit a $p \times b \times r$ model, it needs to be slightly modified by adding summation and index terms. Other changes can be made to the model, for example, by adding a washin rate ω_r^i, where resource r is constantly introduced, and a washout rate ω^o, where all phages, bacteria, and resources are washed out at a proportional rate. Equation (2.2) highlights the ω^i and ω^o additions in red. By default, ω^i and ω^o values are zero unless stated otherwise. When $\omega^i = 0$ and $\omega^o = 0$, and the system includes only a single phage, a single bacterial strain, and a single resource, the model reduces to the original Golding model. The probability of phage p infecting bacteria b is r_{pb} and is not to be confused with the resource concentration R_r. The interactions act independently of one another and are the sum of all interactions as they occur.

$$
\begin{aligned}
\frac{dR_r}{dt} &= -\sum_{b\in B} e_{br}\cdot g(R_r, v_{br}, K_{br})\cdot (U_b + \sum_{k=1}^{M} I_{b_k}) + \omega_r^i - \omega^o \cdot R_r \\
\frac{dU_b}{dt} &= U_b \cdot \sum_{r\in R} g(R_r, v_{br}, K_{br}) - U_b \cdot \sum_{p\in P} r_{pb}\cdot P_p - -\omega^o \cdot U_b \\
\frac{dI_{b_1}}{dt} &= U_b \cdot \sum_{p\in P} r_{pb}\cdot P_p - \frac{M}{\tau_b}\cdot I_{b_1} - \omega^o \cdot I_{b_1} \\
\frac{dI_{b_k}}{dt} &= \frac{M}{\tau_b}(I_{b_{k-1}} - I_{b_k}) - \omega^o \cdot I_{b_k} \text{ for } k = 2,\ldots,M \\
\frac{dP_p}{dt} &= \sum_{b\in B} \beta_{pb}\cdot \frac{M}{\tau_b}\cdot I_{b_M} - r_{pb}\cdot (U_b + \sum_{k=1}^{M} I_{b_k})\cdot P_p - \omega^o \cdot P_p \\
g(R_r, v_{br}, K_{br}) &= \frac{v_{br}\cdot R_r}{R_r + K_{br}}
\end{aligned}
\tag{2.2}
$$

2.7.5 Other Models

ODE and DDE methods are not the only methods to model phage-bacteria dynamics. Differential equations can also be used to model the molecular level, potentially to model individual bacteria and phages interacting with one another at an atomic level [58, 59]. Partial differential equations can be used to model the diffusion and interactions of bacteria, phages, and resources through the system [60, 61].

Chapter 3

Methods

3.1 Project Overview

To help complete this Master's thesis, I developed various tools to assist in creating the final model outputs. The project consists of four parts, with the final one being optional.

- A GUI network creation tool is used to define the number of phages, bacteria, and resources in the network and their interactions.
- The simulation framework handles the data, runs the ODE simulations, and sends the data back to the dashboard.
- A dashboard that the user can interact with to run simulations, edit parameter values, and plot and interact with the visualizations.
- Optionally download the simulation data to create your own visualizations and analyses.

Appendix C contains a flowchart illustrating the user-system interactions.

3.1.1 Network Creation Tool

The user uses the GUI network creation tool to create and edit the network interactions. There are three types of variables in the simulations: phages, bacteria, and resources. Every node in the network represents either a unique phage population, bacterial population, or resource. A bacterium can be further divided into uninfected and infected bacteria. An edge links two nodes together if there is an arbitrary interaction occurring between them. Phages infect bacteria and consume resources. The ability of a phage to

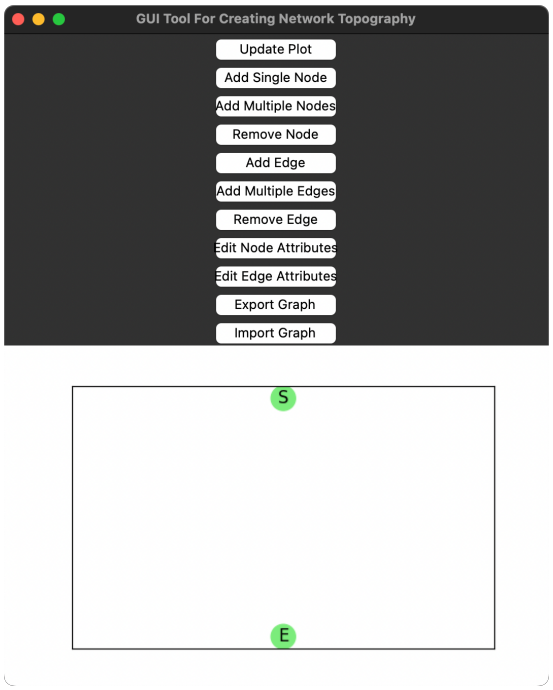
infect a specific bacterium and the resources each bacterium can utilize describe a network of interactions. Finally, the user can export (and later import to edit) the network representation for use in Section 3.1.2, Section 3.1.3, and Section 3.1.4. Figure 3.1 shows the layout of the GUI tool and example networks. This report will use these network structures.

Every node represents a unique entity, and each entity has its intrinsic properties. The user can intuitively define phages, bacteria, resources, their interactions, environmental data, and model settings using the GUI tool. This tool allows users to quickly and intuitively define entities and their attributes, entity interactions and their attributes, environmental data, and model settings. An edge links two entities together if there is an arbitrary interaction occurring between the entities, with the properties exhibited in the interaction dependent on the interacting entities. Self-interactions are allowed in the network. There is an environment node used to store global environmental data, such as the system's temperature and pH. The settings node holds information such as simulation length, max time step, and the type of ODE solver to use. The tool provides functionalities for adding, editing, and visualizing nodes and edges, as well as importing and exporting the network structure.

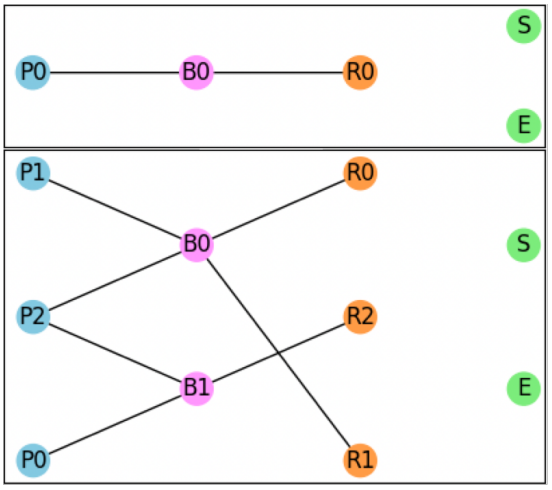
Once the user is happy with the graph shape, they can export the network representation for use in Section 3.1.2, Section 3.1.3, and Section 3.1.4. The most important part is that the user defines the shape and the attributes of the network, as these cannot be edited in part 2 onwards. It is possible to return to the network creation tool and upload the graph to edit the network representation and default parameter values.

The user can edit the values of the attributes in Section 3.1.3, so the parameter values do not have to be perfect. As such, the user does not need to keep on using the GUI tool to edit parameter values.

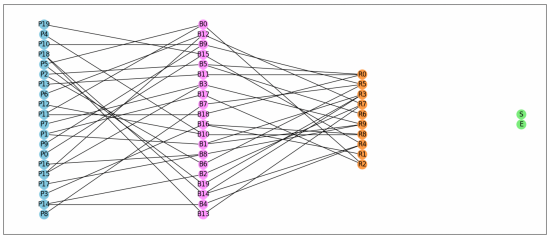
Figure 3.1a shows the layout of the GUI tool. Figure 3.1b shows an example network. Users can edit the graph, including adding or removing nodes and edges, as well as editing parameter values by using the various buttons. Manually adding nodes and edges can become tedious and repetitive for large graphs, so users can add multiple nodes and edges simultaneously. The user can self-determine the default attribute names and values to assign, as well as, if applicable, how the parameter values are randomized. By default, there is already an environment node, "E", and a setting node, "S". These nodes store data such as pH, temperature, or the simulation length. Nothing can interact with the environment and setting node, as they hold data about the environment and network solver.



(a) The GUI tool, as seen upon program startup.



(b) A $1 \times 1 \times 1$ and $2 \times 2 \times 3$ network.



(c) A $20 \times 20 \times 10$ network.

Figure 3.1: This network topography, along with a $1 \times 1 \times 1$ network, will be used in the Chapter 3 and Chapter 4 sections. The parameter values for the networks can be found at Table E.1, Table E.4 and Table E.3 Each node represents a phage, bacteria, or resource, with arbitrary interactions occurring between them. Although not shown and used here, edges between the same entity types and self-loops are allowed.

3.1.2 Simulation Framework

The user provides an ODE model and the network topography as input to the framework. The simulation framework handles the input, output, collection, and storage of the simulation input and output. The framework uses SciPy's [62] *solve_ivp()* numerical solver [63] to simulate the provided ODE equations and calculate the population levels through time. The user receives two outputs from the framework. The first output is an array of time values that the solver used to calculate the population count. The second output is an array containing the population count at each time step for every entity.

To facilitate more complex model behavior, additional system variables can be added to the simulation. An example of this distinction is the difference between uninfected and infected bacteria. In the network model, a $3 \times 2 \times 3$ structure is explicitly defined,

representing three phages, two bacterial strains, and three resources. The solver is, however, instructed to incorporate $2 \cdot M$ supplementary state variables corresponding to the infected states. Therefore, the ODE solver would solve for three phages, two uninfected bacteria states, $2 \cdot M = 2 \cdot 4 = 8$ infected bacteria states, and three resources. Adding a resource reservoir to the model is straightforward. This involves introducing three additional resource variables into the solver, allowing the ODE system to model resource transfer from the reservoir to the simulation environment. The provided ODE model must accurately model and transfer resources from $R_{r_{\text{reservoir}}}$ to $R_{r_{\text{chemostat}}}$ correctly. The bacteria would only consume from $R_{r_{\text{chemostat}}}$.

The user's ODE model must accurately represent each (extra) phage, bacterium, and resource and correctly handle changes in states.

3.1.3 Visualization Dashboard

The third part involves analyzing and visualizing the simulation results on an interactive Dash Plotly [64] dashboard. The user can use a dashboard built using Plotly Dash to interact with the solver and network. The user can quickly change parameter, environment, and setting values with the dashboard. As output, the dashboard displays interactive plots, enabling the user to analyze the system.

The dashboard enables users to interact with the network, model, and prebuilt visualizations. The dashboard contains three separate sections. The first section enables the user to edit parameter values and solver settings on the fly, allowing for quick iteration through different conditions and fine-tuning of parameter selection without needing to rebuild the network using the GUI tool. The second section enables users to visualize how the population count evolves for a given IC and parameter values, allowing them to quickly test the network input. The final section enables the user to conduct more advanced analyses on the network, for example, by modifying multiple parameter values and visualizing the resulting output.

3.1.3.1 Editing Network and Parameter Values

The editing network and parameter value contain five separate sections.

Initial Condition The IC settings panel (Figure 3.2a) allows the user to edit the initial starting values of the entities. Each entity type has a table containing the initial population count.

Vector Data Data stored as a vector, which includes data on the nodes of phages, bacteria, and resources, is displayed in the Vector tab. This data is typically associated with node data. That would be the washin rate of resources or the bacteria's latent time Figure 3.2c.

Matrix Data Data stored as a matrix, typically representing the edges between phages, bacteria, and resources, is stored in the matrix tab, Figure 3.2b.

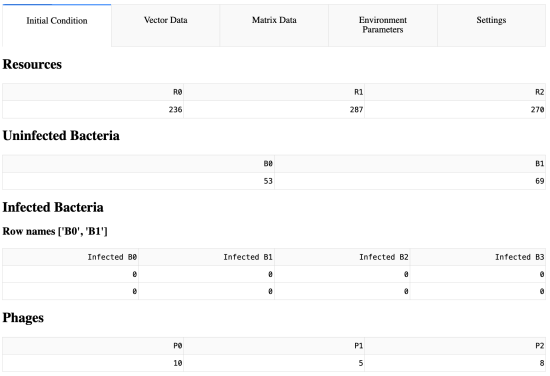
Environment and Settings The environment data and settings data also have their own tab, Figure 3.2d and Figure 3.2e, respectively. The data stored in the environment act as global variables, like the pH of the system or the constant washout rate ω^o. The settings node contains the solver and simulation settings, including the simulation length and minimum time step.

3.1.3.2 Visualization and Analysis

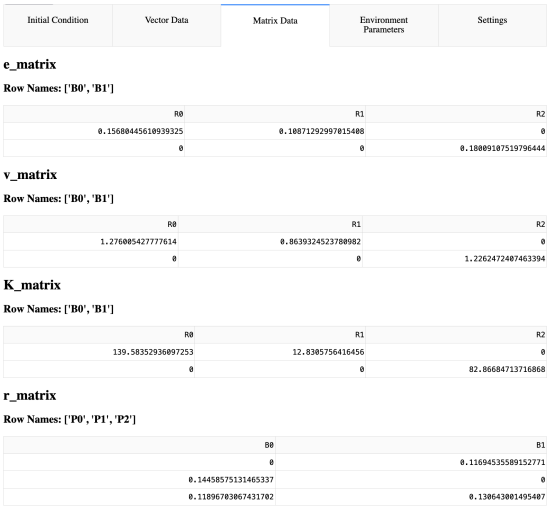
In the analysis section, the user can run different analysis methods to gain a greater understanding of the model. For simplicity, the visualizations on the dashboard only support a $1 \times 1 \times 1$ model. This makes it easier for the user to analyze the system. The goal of the dashboard is to investigate a simple system in order to gain a deeper understanding of the system. The next step is to simulate more complex models using the Ultimate Analysis section, where you can implement your own visualizations. There are five prebuilt visualizations described below. These five visualizations are called serial transfer, parameter analysis, initial value analysis, phase portrait, and Sobol analysis. These visualizations aim to investigate how a simple system responds to varying inputs before moving on to more complex models.

After the user has a deeper understanding of the system, they can run and download custom simulations to create their own custom visualizations in the Ultimate Analysis section. The saved simulation data is stored as a *.parquet* file, a tabular-like data format. Dask can query the simulation data, allowing users to find specific simulation results. Parquet with Dask offers superior performance and data storage solutions that Pandas does not offer.

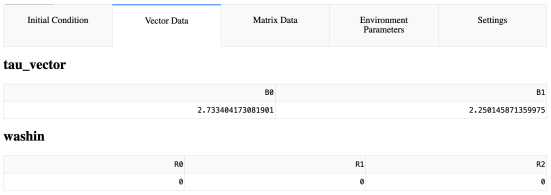
Serial Transfer On the dashboard, a user can select the dilution factor, which divides the phage, bacteria, and resource population count by that number (Figure 3.3a). Then, the program takes the IC values defined in Section 3.1.3.1 and adds those values to the respective entity.



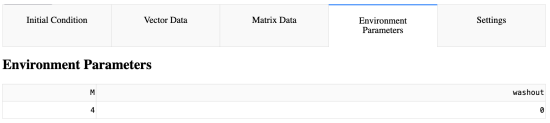
(a) The tab where users can edit the ICs of the entities.



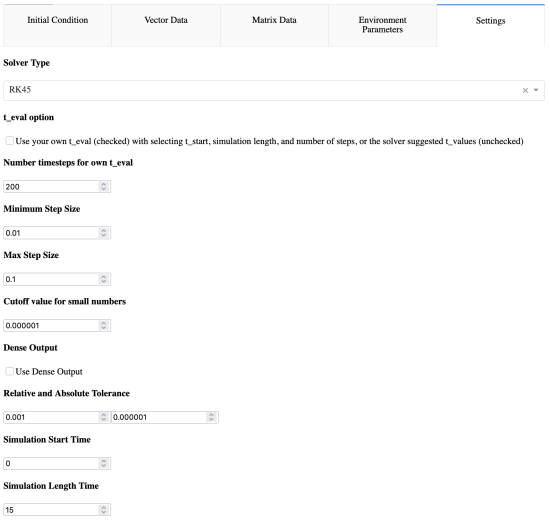
(b) The tab where the user can edit the matrix attribute values.



(c) The tab where users can edit vector attribute values.



(d) The tab where users can edit environment values.

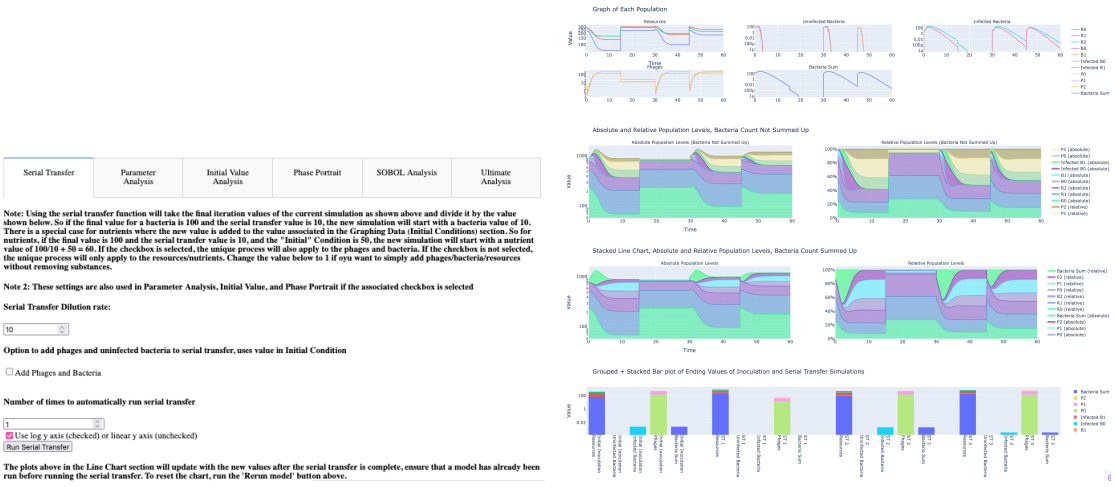


(e) The tab where a user can edit the settings of the solver and simulation.

Figure 3.2: The tabs where the user can edit the various parameter values and control the simulation parameters

As an example, if the simulation ended with 3500 phages, 100 (uninfected) bacteria, and 50 resources, the dilution factor is 10, and the user wants to add 350 new bacteria and 130 new resources for the following simulation, the new starting condition for the following simulation would be $\frac{3500}{10} = 350$ phages, $\frac{100}{10} + 350 = 660$ bacteria, and $\frac{50}{10} + 130 = 135$ resources.

As output, ST will display how the population evolves, as well as the final population value at the end of each serial transfer run. Figure 3.3b shows an example Sobol output.



(a) The section where the user can set up the ST. **(b)** The ST output plots.

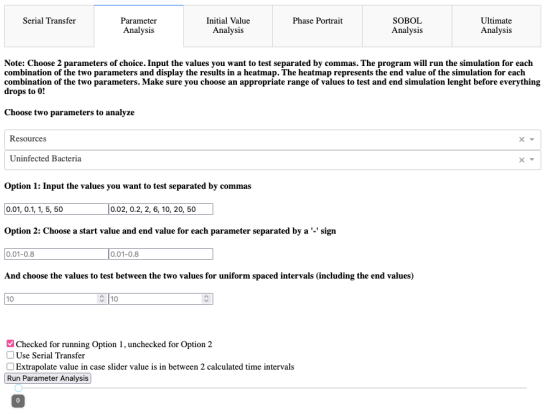
Figure 3.3: The ST settings and output.

Parameter Analysis The Parameter Analysis (PA) settings tab, as shown in Figure 3.4a, allows the user to select two parameters and individually run the model with varying input values. The values that can be tested and changed include all IC values, vector and matrix data, and environmental data. As input, the user can select two parameters of choice. The user can manually choose which parameter values they want to test or test a range of values equally spaced by selecting the number of values to test. Finally, the user can optionally run a ST, where the ST uses the settings found on the ST tab.

Figure 3.4b shows the heatmap that the user can expect, one heatmap for each entity type. Each heatmap cell represents the input of two unique parameter values and shows the population count for that parameter run at the time indicated by the slider. As the user slides the slider, the value inside the cell updates to correspond with the selected time. Note that the heatmap color range resets for each heatmap, so similar colors across heatmaps and across time will not correspond to the same values.

Initial Value Analysis The initial value analysis (IVA) settings tab, as shown in Figure 3.5a, allows the user to select a single parameter and adjust its value over a range of values, visualizing how a change in parameter value affects the population count of the entities.

Figure 3.5b shows the plots that the user receives. Three plots are created, one for each resource, (uninfected and infected) bacteria, and phage, along with a total bacteria sum



(a) The available PA settings.

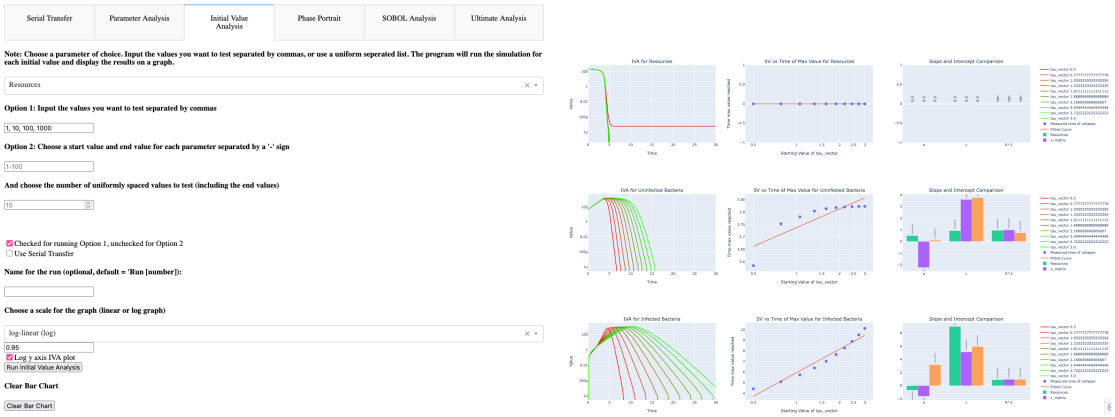
(b) The expected PA output. The left figure shows the PA output for the resource value at $t = 2.81$ for different initial numbers of uninfected bacteria and initial resource concentrations. The right figure shows the value of the uninfected population at $t = 2.81$.

Figure 3.4: The PA settings and output.

plot. The left plot shows the population count through time, one line for each parameter value submitted. The middle plot takes each run and calculates the "percentage from the max value" (default value of $0.95 \rightarrow 95\%$) reached from the peak. In short, the 95% rule identifies the maximum value reached during the simulation and then calculates 95% of that maximum. It determines the time point at which this 95% threshold is first reached. See Appendix F.2 for more information on what the 95% rule is. This value is considered the time of peak and is used to address some issues that can arise when the population plateaus or continues to rise. For example, suppose the phage population starts to plateau early in the simulation. In that case, the program can calculate that the "peak" of the phage population happened towards the beginning of the simulation.

We place the initial concentration value on the x-axis and plot the time of peak value on the y-axis. Using this data, we apply either a linear or logarithmic fit. In Figure 1 of Mulla et al. [6] (see Figure 3.6), the authors vary the initial bacterial concentration and measure the time until bacterial collapse. The observed logarithmic decrease suggests that the phage kinetics is adsorption-limited. Figure 4.2a replicates Figure 1 of Mulla et al. [6].

Using the IVA tool can help understand how a change in parameter value affects the time at which the population count reaches a maximum. The slope, intercept, and R^2 value are stored and saved in the third plot, a bar chart with an editable name. For every rerun of the IVA, the bar chart stores the slope, intercept, and R^2 value and displays the linear regression next to previous runs. When executed with multiple parameters, this enables comparison of high-level results across various parameters and experimental conditions.



(a) The settings for the IVA tab. **(b)** An example of an IVA output.

Figure 3.5: The IVA settings and output.

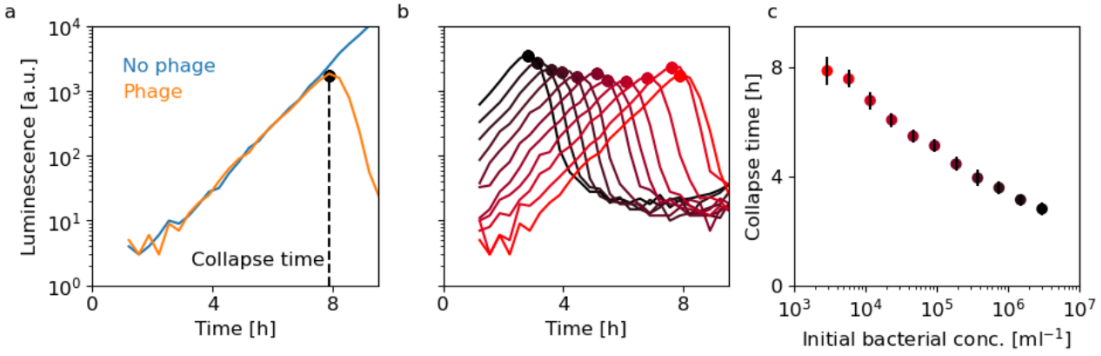
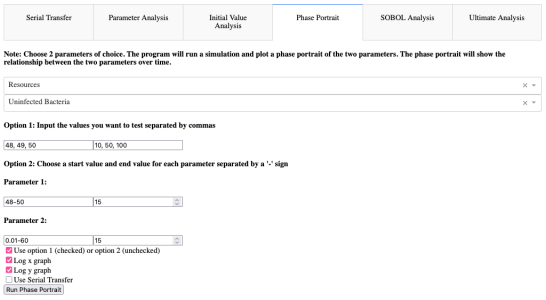


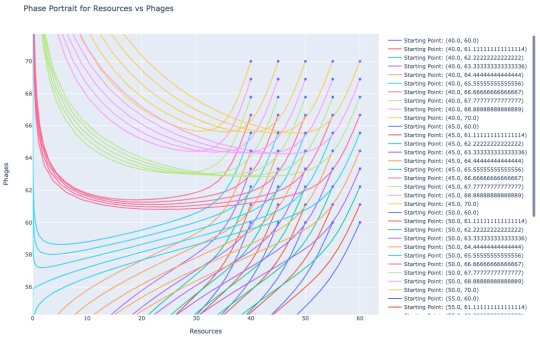
Figure 3.6: Phage-induced bacterial population collapse is limited by phage-bacteria adsorption. a) Exemplary growth curve of E. coli in the absence and presence of a phage (Bas04), measured by bioluminescence. b) Growth curves measured at different initial bacterial concentrations b_0 ($3 \times 10^3 - 3 \times 10^6 ml^{-1}$ from red to black; colors as in c) at a fixed multiplicity of infection (MOI, i.e., phage-to-bacteria ratio) of 0.4. c) Collapse times from b); the observed logarithmic decrease (Pearson's $\rho = 0.995, p = 10^{-10}$) suggests that the phage-amplification kinetics is adsorption-limited. Error bars show standard errors from bootstrapping the growth curves. Figure and caption sourced from Mulla et al. [6].

Phase Portrait The phase portrait plot enables users to analyze how a phage, bacterium, or resource population evolves in relation to one another. Phase portraits indicate how one population increases while the other decreases and vice versa. Steady states can be identified and classified as either stable, unstable, or saddle points. It is also possible to visually identify attractor and repeller points by observing where the population values tend to trend. By comparing different starting points, it is possible to see if the system is chaotic or not. Figure 3.7a and Figure 3.7b show the phase portrait setup and sample output.

Sobol Sensitivity Analysis It is essential to understand how a change in parameter value impacts the output of a model. Models will have parameters that are more



(a) The user can select two starting values for the IC. However, they cannot choose vector, matrix, or environment settings, as the plot displays the development of phage, bacteria, or resource populations against other phage, bacteria, or resource populations. As typical, the user can select their own values or auto-generate values between two values, as well as use an ST option. There is also an option to take the logarithm of the x and/or y-axis.



(b) An example phase portrait run.

Figure 3.7: Phase portrait settings and output.

important and have a greater impact on the model's output than other parameters.

Sobol analysis [65], a variance-based sensitivity analysis, is a method that enables users to quantify the importance of input parameters on a measured aspect of the output by varying the input parameter values of the model and measuring the resulting change in model output. Sobol can only measure a single univariate model output, for example, the final population value, the smallest or largest value reached, the time at which the largest value was reached, or any other univariate output. Sobol quantifies the variance in the output that can be attributed to a specific parameter and measures the effects of global/total (ST), first-order ($S1$), and second-order sensitivity ($S2$).

First order $S1_i$, or local sensitivity, is the measurement of the effect that parameter i has on the variance of the output. The second order $S2_{ij}$ is the measurement of the interaction between parameter i and parameter j and how this interaction contributes to the output variance. Etc. for third order and higher. Global ST, also known as total sensitivity, is the sum of all interactions. If $ST_i \gg S1_i$, then parameter i depends on higher-order interactions with other parameters, while when $ST_i \approx S1_i$, then i does not interact much with and depend on other parameters. $ST_i \geq S1_i$ and ST_i can be greater than 1, while $S1_i \leq 1$.

When a model is treated as a black-box model, the model acts as a function $Y = f(X)$, where X is an input vector of d parameter values, and Y is a univariate model output. The first-order sensitivity measures the output variance resulting from the primary effect of parameter X_i. Measuring the effect of varying X_i averaged over other input parameters and standardized to provide a fractional contribution to the overall

output variance. The following equation describes the first-order sensitivity.

$$S1_i = \frac{V_i}{Var(Y)}$$

where $V_i = Var_{X_i}(\mathbb{E}_{X_{\sim i}}[Y|X_i])$ and where $X_{\sim i}$ represents all the parameters that are not X_i. All parameters are summarized in Table A.2

The second-order index measures the impact of the interaction between input X_i and X_j. For many inputs, this becomes unwieldy to analyze. The global sensitivity is used to analyze the global sensitivity without evaluating $2^d - 1$ indices. It measures the contribution to the output variance of X_i, including all variance due to X_i's interaction with other variables.

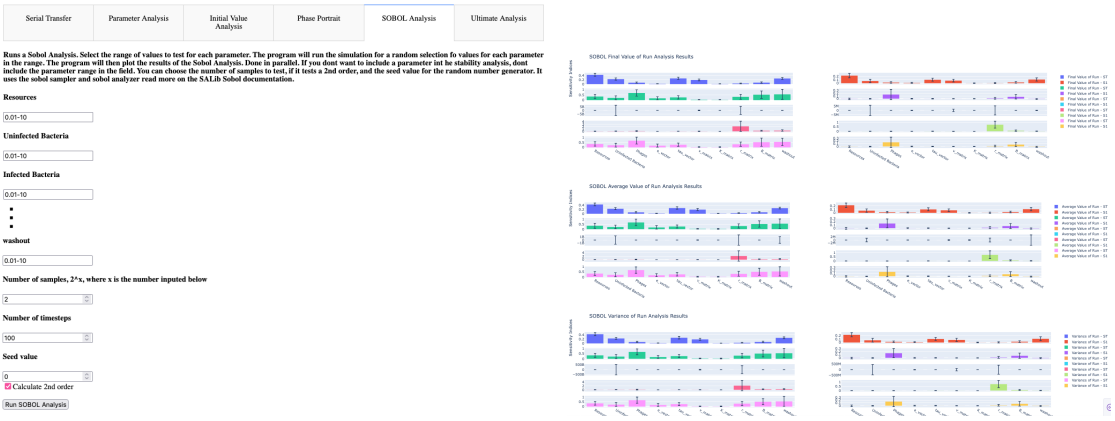
$$ST_i = \frac{\mathbb{E}_{X_{\sim i}}[Var_{X_i}(Y|X_{\sim i}))}{Var(Y)} = 1 - \frac{Var_{X_i}(\mathbb{E}_{X_i}[Y|X_{\sim i}])}{Var(Y)}$$

Sobol accepts a list of parameter names and an interval of values to sample from, which the user can input in the Sobol settings tab, Figure 3.8a. If no values are specified, the simulation excludes the parameter and uses the default value instead. The user then needs to select the number of samples to run, using the formula 2^x, where x is the number they input, and 2^x is the number of samples that Sobol will create and run. The larger x is, the more accurate the Sobol analysis results will be, but the more simulations we will need to run. When the second-order option is not selected, the model is executed $N(D + 2)$ times with randomly sampled input values, where N is a power of 2, and D is the number of input parameters. If the user wants to analyze the second-order interactions, then the model will run the system $N(2D + 2)$ times. Due to the randomness of the sampling method, the user can, but does not need to, submit a seed value.

Figure 3.8b shows a sample Sobol output. The global and first-order sensitivities are displayed side by side, with each sub-row in a plot representing a specific entity type. We can see the proportion of the global and local sensitivity for each entity type and each parameter.

Upon completion of a Sobol analysis, the original simulation data is saved to the disk as a *.pickle* file so that the user can reuse the data and run their own Sobol analyses.

Ultimate Analysis Creating a dashboard that can accommodate various inputs is a challenging task. Predicting the type of plots that a user might be interested in and the type of behavior they want to analyze is impossible. The Ultimate Analysis section does not produce any visualizations or analysis; instead, it allows the user to define which ICs



(a) The Sobol settings tab. **(b)** Sobol's expected output.

Figure 3.8: Sobol variance analysis settings and output.

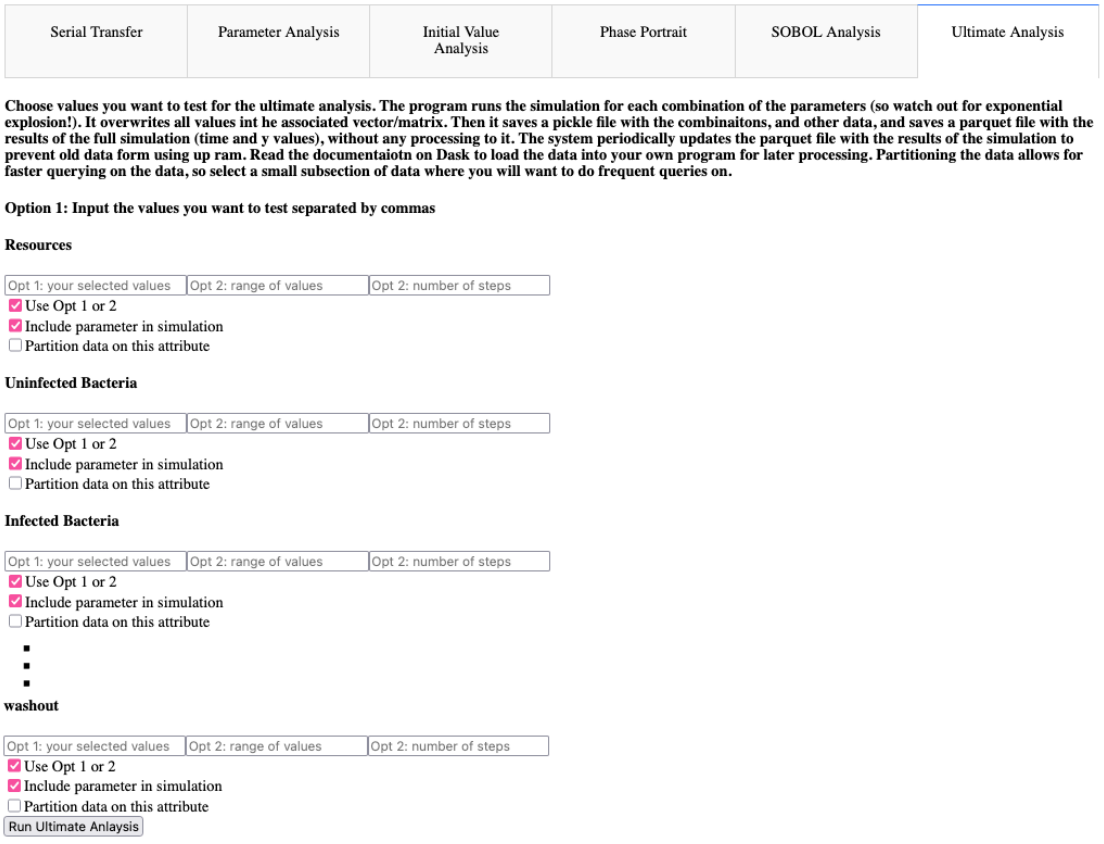


Figure 3.9: The Ultimate Analysis setup tab.

and parameter values they want to simulate (Figure 3.9). The solver will iterate over every possible parameter input and save the results in a *.parquet* file. Ultimate Analysis uses Joblib to parallelize the simulation runs [66].

Using Dask and the saved *.parquet* file, the user can query for specific runs, for example, runs where a parameter value was greater than 0.05, and use the simulation data to create their own plots.

3.1.4 Custom Visualizations and Analyses

The final part, an optional step, enables the user to define the parameters they want to simulate in the Ultimate Analysis section and download the resulting raw simulation data. The user can use this data to create their own custom visualizations without having to rerun the simulations, especially if there are many simulations. The data can be further processed and visualized as desired by the user. If the user partitioned the data, the querying for the data speeds up, and they can use Joblib to parallelize the data querying analysis.

As the dashboard cannot create a graph for every situation or be easily adapted to analyze every situation, Section 3.1.3.2 can be used to run and download the simulation data to the disk, allowing you to create your own custom visualizations later.

3.2 Software Used and Packages

I created the program with Python [67] and extensively use various packages, ranging from standard scientific packages such as NumPy [68] and SciPy to more niche packages, including pickle and SALib [69, 70].

The graphical tool utilizes Tkinter as the front end, handling user inputs, while NetworkX [71] stores the graph and contains the attribute data of the edges and nodes. The GUI tool displays a generated graphical representation of the graph. The creation of the picture utilizes Matplotlib [72] to generate the graph figure.

The simulation framework, which serves as the backend of the modeling, makes extensive use of SciPy's *solve_ivp()* to generate the ODE data. It also uses NetworkX to load the graph and parameter values.

The visualization component heavily utilizes Dash and Plotly. Dash hosts the web server and displays the HTML and visualizations while handling input and output. Upon choosing parameter values and clicking "Submit", Dash registers the activity and calls the function registered to the button, sending data such as parameter values and options, like "log x-axis" to the backend server. In the backend, various inputs are handled, such as converting the input string *"0.05, 0.1, 0.15, 0.2"* into an iterable list *[0.05, 0.1, 0.15, 0.2]* that the simulation framework can iterate over to vary the parameter value.

If there are many simulations to run, an intermediate call executes Joblib, a parallel computing program, as seen in Section 3.1.3.2 and Section 3.1.3.2. Joblib parallelizes computations across multiple CPUs to speed up computation time. The Ultimate Analysis uses Pandas to write the data to a *.parquet* file. Pandas Parquet offers efficient data

compression, efficient memory usage, and, when combined with Dask, efficient querying functionalities in a DataFrame format that many data scientists are familiar with.

Sobol uses the SALib library to sample and analyze the parameter input. Both Ultimate Analysis and Sobol save a *.pickle* file containing a dictionary with the parameter values tested, setting values, and other important information regarding the simulation.

Section 3.1.3.2 uses SciPy's *curve_fit()* function to curve fit the points in the middle plot (Figure 3.5b).

Other packages used in the project include collections, copy, warnings, itertools, os, datetime, json, gc, and time.

Chapter 4

Experiments and Results

In the following section, I will apply the software to demonstrate the predicted dynamics of the phages and bacteria under different conditions. I first begin with analyzing the model using a Sobol analysis on the simple Golding model, Equation (2.1). Using the most critical parameters identified from the Sobol analysis, I run multiple IVAs to describe the change in graph behavior for different inputs. I use the information gained from this to describe how the graphs change for a wide range of inputs. Next, I analyze if phages will proliferate or not depending on the initial phage, uninfected bacteria, and resource concentration. I finally extend the analyses to a large community and examine how changing parameter values affect the growth of phages and bacteria, as well as whether the phages and bacteria can survive and coexist or not.

4.1 A Realistic Growth Curve

As the bacterial population grows, resource consumption accelerates until, eventually, the resources start to deplete, leaving no resources available. The delay between the peaks of uninfected and infected bacteria is due to the infection stages and the latent period of phage infection. Each bacterium transitions from infection stage k to $k+1$ at a rate of $\frac{M}{\tau}$. Therefore, decreasing the number of infection steps, M, or increasing the latent period, τ, amplifies this delay. A more extended latent period means it takes more time for bacteria to progress through the infection stages.

Figure 2.3a shows how the infection rate surpasses the bacterial replication rate at $t = 4$, causing the bacterial population to decline despite the availability of resources. This moment coincides with the rise of the phage population. Observing the timing of these events and changes in the graph's behavior, as well as their relationships across different

graphs, helps clarify the complex population dynamics and the interdependence of the populations. The nonlinearity of the models makes the analysis harder to understand and conceptualize.

Understanding the system becomes more complicated as the model increases from a $1 \times 1 \times 1$ system to a $p \times b \times r$ system. Now, up to any number of phages can interact with any number of bacteria, and any number of bacteria can interact with any number of resources, each with its unique parameter values. These varying rates will significantly influence the system's dynamics, making it difficult to determine which event caused what due to the increase in the number of interactions. There are only two interactions that occur in a $1 \times 1 \times 1$ system. With a $p \times b \times r$ system, there are at most $p \cdot b + b \cdot r$ interactions that can occur. The simultaneous occurrence of multiple individual events makes it more challenging to identify the cause of the event and understand how the event's action will propagate through the network. A method to circumvent this issue is to knock out specific nodes or edges, rerun the simulation, and compare the results. The knocked-out node or edge will no longer contribute to the simulation, resulting in a cascading effect on the other populations. This is indeed observed in real microbial communities. For example, Hsu et al. [73] found that predating a specific bacteria strain in a mouse's gut microbiome with a phage had a cascading effect in other bacteria populations due to interbacterial interactions.

4.2 Sobol Sensitivity Analysis Results

The Sobol method is a global sensitivity analysis technique that quantifies the contribution of each input parameter, as well as their interactions, to the variance of a model's univariate output. It decomposes the output variance into fractions attributed to individual parameters and their combinations, providing first-order and total-order sensitivity indices. A Sobol analysis identifies the most influential parameters affecting model output. The insights from this analysis inform the selection of key parameters for subsequent simulations, ensuring that further investigations focus on those with the most significant impact.

Figure 4.1a shows the impact that the parameter had on the final value of the population at $t = 15$ for a $1 \times 1 \times 1$ system, on the original Golding model, Equation (2.1). Figure 4.1b shows the impact that the parameter had on the peak population count using the 95% rule. (see Section 3.1.3.2 and Appendix F.2 for an explanation on what the 95% rule is). Figure 4.1c shows the impact that the parameter had on the time of the peak, using the 95% rule.

The parameters tested include all those listed in the basic Golding model, except for the uninfected bacteria $(I_1, \ldots, I_4), M, \omega^i$, and ω^o. It did not make sense to include infected bacteria I_k at the start of the simulation. Infected bacteria are an internal model variables rather than primary outcomes of interest, and including them could make interpreting the outputs such as the final value or time of peak value more difficult. M, the number of stages that the infection goes through, cannot be tested with Sobol as M is an integer, while Sobol randomly chooses floating-point values. While testing, the washin rate ω^i and washout rate ω^o consistently had the most significant influence on the final peak value and the time of peak value, as determined by the 95% rule. Washin and washout are not part of the original model from Golding, and the addition of a washin and washout term significantly skews the results and analysis; therefore, it was excluded from the analysis. The results for a Sobol analysis with washin and washout can be found in Appendix F.1.

4.2.1 Resources

The final value for the resources depended heavily on the initial resource concentration. There were not many interactions with other parameters because $ST \gg S1$. e, the ability for the bacteria to consume resources, had little influence on the system despite e acting as the link between the resources and bacteria and directly controlling the rate of resource consumption. τ, the latent period, had a larger influence on the final resource value than e.

The peak value and time of peak value graphs for the Sobol indices of Resources are empty, as the resource concentration continuously decreases from different initial conditions. The initial resource always had a peak at $t = 0$.

4.2.2 Phages

The final phage population value depended most on r (the probability of a successful phage-bacteria infection), with β (the burst size) as the second most important parameter influencing the final population value. The other parameters had little to no influence on the final phage population levels.

The Sobol peak value plot is equivalent to the final population value. Similar to the final value, the phage max value is highly dependent on the value of r and β, and the other parameters had little to no influence on the peak value for the phages.

For the time of peak value, τ becomes the most crucial parameter for determining the time of peak value, while r is not as important anymore. The initial phage population

has a negligible influence on the final population value, which is about as crucial as r. β roughly maintains the same sensitivity value across the final, peak, and time of peak analyses.

4.2.3 Total Bacteria

The final total bacterial population mainly depended on β, the burst size of the phage, but through many secondary or higher-order interactions that occurred as $ST \gg S1$. The final population depended heavily on many higher-order interactions with the initial resource concentration, τ, and e.

β remains the most critical parameter to the model, but instead of ST and $S1$ being equal to 1 and 0.28, as in the final value, the sensitivity values are only 0.54 and 0.16, respectively. Every parameter has some influence on the output, but with higher-order interactions, as with all parameter inputs, $ST > S1$.

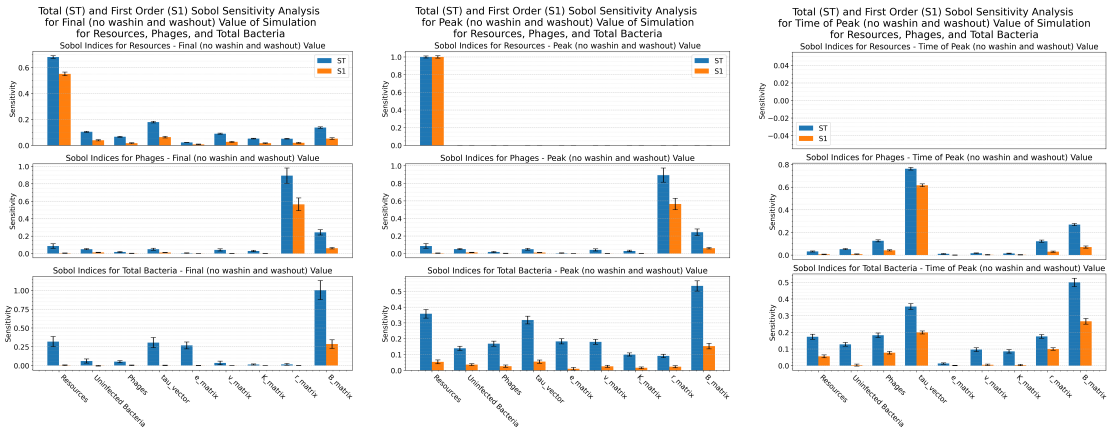
β and τ are the two most essential factors in determining the time of peak for the total bacteria. The only parameter that does not influence the time of peak in some manner is e. Otherwise, every parameter has some influence on the time at which the bacteria population peaks.

4.2.4 Summary of Sobol Results

Some results are surprising. e, v (max growth rate of a bacteria), and K (half saturation Monod constant) are consistently the least important factors in determining the final value, peak value, and time of peak. Changing the parameter values that directly affect bacterial growth would have a considerable impact on the resources and phages. More bacteria mean more resources are consumed, and more phages are created. Knowing that e, v, and K are relatively unimportant compared to a parameter like r or τ, future analyses do not have to focus on e, v, and K. As β, r, and τ are relatively important, future analyses could focus on how those parameters influence the growth of phages and bacteria.

4.3 Graph Behavior with IVA

Understanding how a change in parameter values across a wide range of values affects the curve is essential for comprehending how the model operates across a variety of values. The sections below provide both quantitative and qualitative elaborations on



(a) The final value, with no washin and washout.

(b) Peak population value, no washin, and washout.

(c) Time of peak value, no washin, and washout.

Figure 4.1: Sobol analyses for the final, peak, and time of peak value without a washin and washout rate. The input value ranges to test each parameter used for this Sobol test can be found in Table E.3, except that washin and washout is set to 0.

how changes in parameter values across multiple values alter the shape of the population curves. Only the most critical parameters identified in the Sobol section, along with the initial resource, uninfected bacteria, and phage value, will be analyzed. Initially, the resources, uninfected bacteria, phage, τ, r, and β will be analyzed. The default parameter values can be found in Table E.1 but are given in the text as well. The sections below use an IVA run to analyze the change in behavior across a wide range of values. Each figure from which the conclusions are made can be found in Appendix F.3. ω^i and ω^o are not included despite having a considerable impact as seen in Appendix F.1 because ω^i and ω^o are not a part of the original Golding model. Some of the results are unrealistic and do not accurately replicate real-life growth curves.

4.3.1 Initial Resource R

The initial resource R concentration values were tested across a range from 1 resource to 500 resources, with a default value of 200 resources. The IVA results can be found at Figure F.3. All resource values are depleted at around the same time, within 1 time unit of each other. As the resource concentration increases, the time it takes for the uninfected bacteria to reach its peak population increases from 0.5-time units to 4.12-time units. The infected bacteria face the opposite, whereas the resource concentration increases, and the time to peak decreases from 9.5-time units to 4.9-time units. The same behavior is exhibited for the phages, except that the time duration changes from 14-time units to 6.3-time units. The phage population showed a significant variation in maximum population, ranging from a maximum of 464 phages to 17,200 phages, a

37-fold difference. The maximum bacterial population sum varied from 55 to 3106 total bacteria, representing a 56.5-fold difference.

4.3.2 Initial Uninfected Bacteria U

The initial uninfected bacteria U population values were tested across a range of 1 to 100 uninfected bacteria, with a default value of 20 uninfected bacteria. The IVA results can be found at Figure F.4. As the initial uninfected bacterial population increases, resources are consumed at a faster rate, and the bacterial populations grow more rapidly and peak earlier. The phages were able to grow faster. The phages peaked at 5.7-time units for 100 initial uninfected bacteria, whereas for small initial uninfected populations, the phages peaked at 10.3-time units. Varying the number of uninfected bacteria had little influence on the maximum value reached. There is a difference of 600 phages between the phages that started with 1 uninfected and 100 uninfected bacteria. The difference in total bacteria peak values is 173. Changing the uninfected bacteria had little influence on the peak values but had a significant impact on the time at which the peak value occurred.

4.3.3 Initial Phage P

The initial phage P population values were tested across a range from 1 phage to 50 phages, with a default value of 10 phages. The IVA results can be found at Figure F.5. The differing phage values had almost no influence on resource consumption and a limited impact on the peak value and time to peak value for both the bacteria and phages. The difference between the peak and time of peak values is minor. The final phage population at $t = 15$ was 9,436, for an initial phage value of 50. In contrast, for an initial phage value of 1, the final phage value reached 10,676. The time of peak difference is just 1.44-time units. As the phage value increases, the time required for the uninfected, infected, and phage populations to reach their peak values decreases.

4.3.4 Latency Period τ

The latency period τ parameter values were tested across a range from 0.5 to 3.5, with a default parameter value of 0.7. The IVA results can be found at Figure F.6. τ does not influence how fast the resources are being consumed. However, as τ increased, the time for the infected bacteria population to peak went from 4.43 to 11.21-time units, and the time it took for the phages to peak went from 5.49 to 14.70-time units, 2.53 times, and 2.68-times increase in time length. Smaller τ values resulted in larger final

phage populations and smaller total bacteria populations. The peak value and time of peak value for varying τ values show little difference for the bacteria sum, ranging from a peak of 1,444 to 1,702 total bacteria and a time of the peak of 3.74 to 3.82-time units, a difference of 0.08-time units. As τ increases, the infection process will take longer, and it will take longer for the bacteria to die. Dying later means the phage's population experiences a delay in growth, taking longer to reach its peak population value. Meanwhile, more bacteria can grow and consume resources. The bacteria will take longer to peak as there is less initial pressure from the phages.

4.3.5 Adsorption Rate r

The adsorption rate r parameter values were tested across a range from 0.001 to 0.2, with a default parameter value of 0.001. The IVA results can be found at Figure F.7. For small r values, all resources are consumed by $t = 5$, while for large r values, only 13.12 resources were consumed as not enough bacteria were created throughout the simulation to consume all the resources. As r increases, the time to reach the max value decreases for the uninfected and infected bacteria and phages. The delay in phage value decreased from 6.32 to 2.35-time units, and the total bacteria decreased from 3.81 to 0.59-time units. However, for large r values, there was little growth of phage and bacteria. For large $r = 0.2$ values, the maximum number of phages and bacteria was 154 and 79, while for small $r = 0.001$ values, the maximum number of phages and total bacteria was 10,464 and 1,588, respectively. As r increases, the probability of a successful infection increases, thereby enhancing the adsorption rate of phages and causing them to grow and peak earlier. This higher adsorption rate causes the bacteria to become infected more quickly, resulting in their faster death. Lower r values allow the bacteria to grow for an extended period, providing more bacteria for the phages to infect, which in turn leads to higher final phage populations.

4.3.6 Burst Size β

The burst size β parameter values were tested across a range from 1 to 100, with a default parameter value of 10. The IVA results can be found at Figure F.8. For large β values, not every resource was consumed, similar to large r values, where for $\beta = 100$, 89 resources were consumed. As β increases, the time to peak for the infected bacteria and phages temporarily increases before decreasing. For the uninfected bacteria, the time to peak (does not change for $\beta = 1$ to $\beta = 10$) takes 3.8-time units, but for larger β values, the time to peak decreases to a minimum of 1.94-time units. There was a significant difference in phage population values, ranging from 4.54 final phages to 42,570 phages,

representing a 9,376-fold increase in phage value. As β increases in value, more phages are being created upon lysis, which infects the bacteria. For β less than 10, bacterial growth is restricted by latency, which determines how fast a bacterium can complete the infection process. At the same time, for β greater than 10, the system transitions to an adsorption-limited regime, where phages cannot adsorb to the bacteria quickly enough, allowing the bacteria to grow more rapidly and reach a peak earlier. As phage production increases, more bacteria become infected, causing the bacterial population to decline earlier and leading to reduced bacterial reproduction. With fewer bacteria present, resource consumption also decreases.

4.4 Initial Value Analysis Results

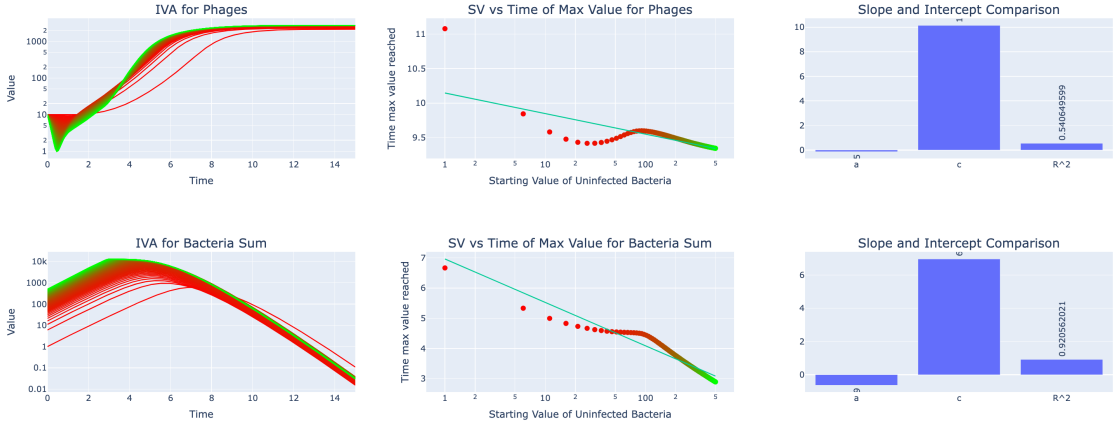
Figure 4.2a and Figure 4.2b illustrate how varying the initial uninfected bacteria population from 1 to 500 (using 100 different starting values) affects the dynamics and time of peak population of phage and total bacteria populations using the 95% rule.

Figure 4.2a perfectly replicates Figure 1 of Mulla et al. [6] (Figure 3.6). As the initial bacteria population increases, the time to reach the phage and bacteria sum peak decreases, following $y = -0.8648 \cdot ln(x) + 9.7911$ and $y = -1.0056 \cdot ln(x) + 7.7626$, with $R^2 = 0.9800, 0.9988$ respectively.

Figure 4.2b on the other hand shows different behavior. As the initial bacteria population decreases from 500 to 100, Figure 4.2a exhibits the same behavior. There is a change in behavior at 100 and less initial uninfected bacteria. Instead of following the predicted line like in Figure 4.2a, the curve for the phages decreases non-monotonically. The bacteria plateau before starting to increase again. The fitted linear regression curves follow $y = -0.1292 \cdot ln(x) + 10.1462$ and $y = -0.6234 \cdot ln(x) + 6.9602$, with $R^2 = 0.5406, 0.9206$ respectively.

The slope tells us how well the uninfected bacteria influence the time of peak value. The larger (positive or negative) the slope is, the greater the impact the uninfected bacteria have on the time of peak value. The closer the R^2 value is to 1, the greater the proportion of the variance in the time to peak value is explained by the uninfected bacteria. The linear regression curve can not explain the change in limiting regions. For the first curve, the uninfected bacteria had a small influence on the time of peak value. In contrast, for the second curve, changing the number of uninfected bacteria had a more significant impact on the time of peak and is considerably more important in explaining the time of peak value.

(a) IVA for Table E.2. Replicates Figure 1 of Mulla et al. [6]. The system is adsorption limited [6].



(b) IVA for Table E.1. For initial uninfected bacteria of less than 100, the phage-bacteria interaction is burst-limited, whereas for initial uninfected bacteria of more than 100, the phage-bacteria interaction is adsorption-limited.

Figure 4.2: Varying the initial uninfected bacteria concentration, from 50 to 500, with 30 unique values tested. Varying the default parameter values can have a significant impact on how changing the initial bacterial concentration affects the system's dynamics. The default values for Figures a) and b) can be found at Table E.1 and Table E.2.

4.5 Phage Proliferation

Understanding how phage behavior varies under different conditions is essential. Phages can experience chemical or biological deactivation or, if the replication process is not fast enough, be removed from the system through washout. It is very easy for a researcher to vary the initial resource concentration, phage, or resource concentration. One of the first experiments that a researcher might conduct in a lab is to investigate how changing the initial population or concentration value of phages, bacteria, and resources affects the survival of phages or bacteria and by how much.

4.5.1 Phase Portrait

Comparing a phage, bacteria, and resource value against one another can reveal how one population evolves in relation to another over time. It is easier to understand how one population evolves compared to a change in another population. It also illustrates how initial population values can influence the evolution of populations in relation to one another.

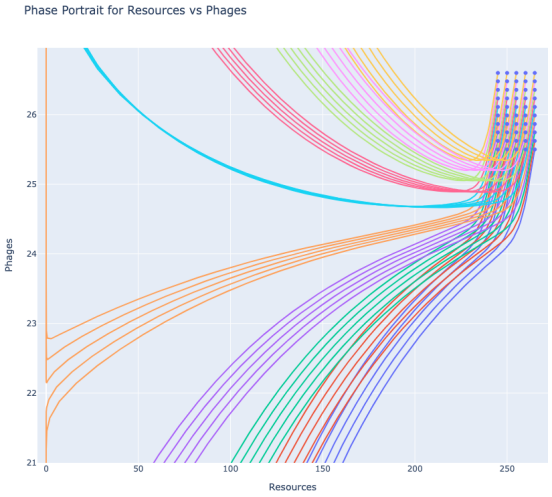
Figure 4.3a shows a phase portrait varying the initial resource and phage concentration. The exact initial phage values have the same color for the line. For phages that start at a population above 25.98, the phage population can proliferate until the washout eventually removes the phages. For phage populations that start below 25.98, the washout removes the phages before they have time to infect and kill the bacteria. Both regions of phages exhibit consistent behavior, either going to 0 or proliferating. If the phage population started at precisely 25.98, if the initial resources were 260 or above, the phages died out. If the initial resource value was 255 or below, the phages proliferated.

4.5.2 An Initial Resource and Phage Value Analysis for Phage Proliferation

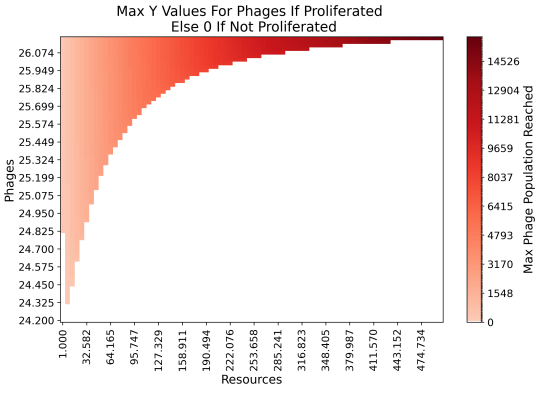
Figure 4.3b expands on the phase portrait by simulating more values and coloring the square depending on whether the phages proliferated or not. The initial resource values span from 1 to 500, and the initial phage values range from 25.5 to 26.5, each with 100 unique values sampled.

Figure 4.3 zooms into the range $(1 - 40, 24.2 - 25)$ for a highly detailed view of the behavior happening around initial resources of 10. From 1 to around 7 initial resources, fewer phages are needed to ensure proliferation. At 7 initial resources, there is a minimum in the phage proliferation boundary. From 7 initial resources and upwards, as more resources are added to the system, more phages are needed to ensure proliferation. Despite this, the change is tiny, a difference of about two phages. Considering the range of possible initial phage populations, the phage proliferation boundary is flat. Under these parameter values, choosing an initial phage population of 27 or higher will ensure phage proliferation. The higher the initial resource concentration, the more final phages will appear. As there are more resources, more bacteria can grow from the resources, which in turn allows more phages to grow.

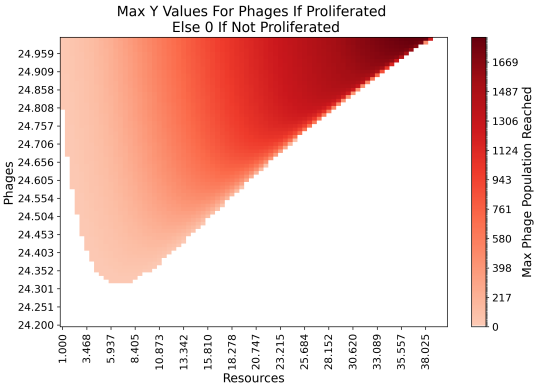
When washout is higher, the system exhibits behavior similar to that shown in Figure 4.3b, but it requires more phages to achieve proliferation. Increasing K shifts the



(a) The zoomed-in plot of a phase portrait with varying resource and phage population from 245-265 and 25.5-26.5, respectively. Each row has its own line color. Diverging behavior is observed for the orange lines (when $P = 25.98$).



(b) Phage population proliferation as a function of initial resource and phage concentrations. Although the color appears uniform along the vertical axis, each cell has a slightly different value. The phage-resource proliferation boundary follows a fitted Monod equation.



(c) Zoomed in to analyze the regime of behavior change near $R = 10$.

Figure 4.3: Varying initial resources and initial phages and the resulting proliferation and fitted proliferation curve. The box is colored red if the phages proliferated for that condition and white if the phages died out. Phages proliferated if they reached twice their initial population at any point in time during the simulation. This simulation used the values from Table E.2 but with washout set to 0.02 instead of 0.

minimum point of the proliferation boundary to the right. In both cases, the phage proliferation boundary is still relatively flat.

4.5.3 An Initial Resource, Phage, and Bacteria Value Analysis for Phage Proliferation

The initial resource concentration had some, but minimal, impact on whether the initial phage concentration would affect phage proliferation. Within the context of the basic Golding model, the initial uninfected bacteria population is one of three parameters

that a researcher can easily control, with the other two being the initial resource and initial phage. For low initial populations of uninfected bacteria, it will be harder for the phages to proliferate. There are not enough bacteria to infect before the washout will remove the phages from the system. The washout ω^o removes a percentage ω^o of phages, bacteria, and resources from the system at each time step.

For large initial uninfected bacterial populations, it will be easier for the phages to proliferate, such that the washout will not immediately eliminate the phages. I extend the initial resource and phage population analysis by adding a third dimension: the initial uninfected bacteria population. The aim of adding the uninfected bacteria is to see how the initial uninfected bacteria will: 1) affect if the phage can proliferate and 2) affect the maximum population that the phages can reach.

Figure 4.4 has three axes: the initial phage, resource, and bacteria population value. The initial uninfected bacteria did not have a significant impact on two aspects: 1) whether the phages would proliferate and 2) how much the phages would proliferate. If sliced along the bacteria axis, there is little difference in the shape of the curve. It was expected that as the uninfected bacterial count increased, fewer phages would be needed to ensure phage proliferation. However, there is no significant change in whether the phages proliferated at all and by how much.

This finding suggests that, under a washout situation, ensuring there are enough phages is the most critical factor among the initial phage, bacteria, and resource values to prevent the phages from being washed out. Conversely, changing the initial resource concentration determines the number of created phages.

4.6 Plotting Parameter Change – $3 \times 2 \times 3$ Model

Now that we have identified the most important parameters in the Golding model, we can analyze how the curve shapes change across a range of parameters for a larger model. Understanding how multiple parameters affect the output is crucial. As the complexity of the model input increases, interactions and their corresponding parameter values play a crucial role in determining which phage or bacteria can survive. The interactions (or lack thereof) will influence the output of the graphs.

The larger model will exhibit similar but slightly different behavior than a $1 \times 1 \times 1$ model due to the multiple interactions occurring simultaneously. A combination of interactions now explains the population curves. The differing parameter values across each interaction will influence how fast each population can grow and decline, which will have cascading effects on the values of other populations.

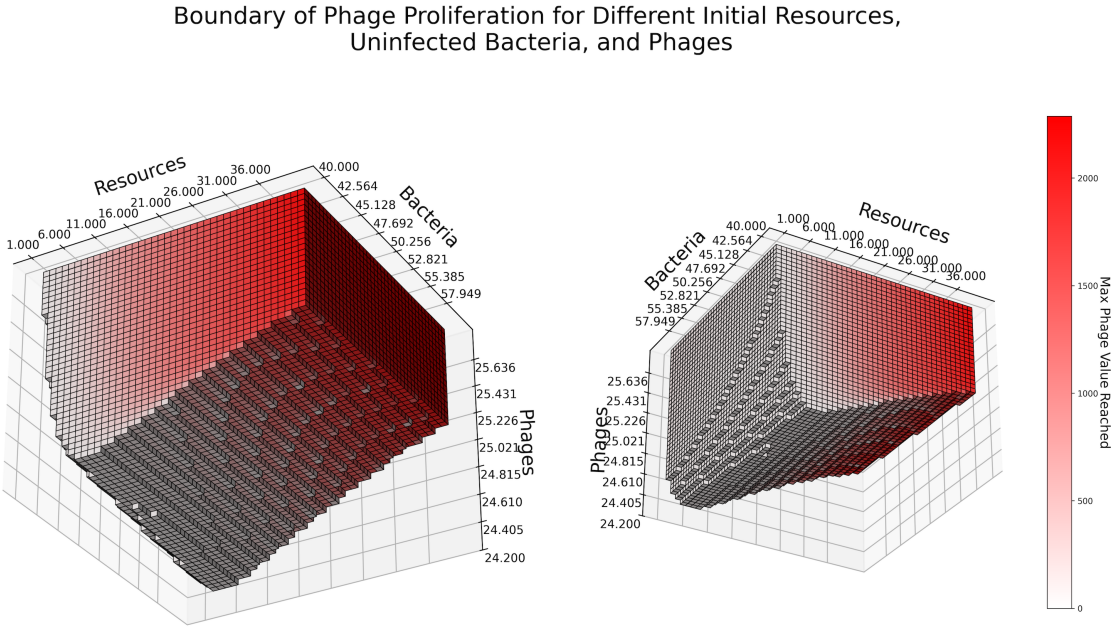


Figure 4.4: A 3D plot of phage proliferation, dependent on the initial resource, uninfected bacteria, and phage population. Color scaling from white to red; the color is dependent on the maximum phage population reached. Zoomed into a small area of interest, zooming out does not offer much more information. In the grand scheme of things, the initial presence of bacteria and resources has little effect on whether phages can proliferate. On a larger initial resource and uninfected bacteria axis range, the boundary becomes essentially flat.

A $3 \times 2 \times 3$ model was chosen because it is situated on the boundary between adding too many phages, bacteria, or resources that would otherwise clutter the plot with lines while still offering behavior that can be compared against one another. The $3 \times 2 \times 3$ graph network used can be found in Figure 3.1b, with the default parameter values listed in Table E.4. B_0 is infected by P_1 and P_2, and consumes R_0 and R_1. B_1 is infected by P_0 and P_2, while consuming R_2.

Figure 4.5a and Figure 4.5b show a 7×7 matrix of subfigures between washout rates of 0 and 0.02. Each subfigure uses a different combination of r and β parameter values. The adsorption rate and burst size were chosen because they had the most significant effect on the phage population. These results demonstrate that even when controlling the two most important parameters that drive phage growth, secondary, less important parameters and interactions still influence the outcome.

All initial phage values started at 10. This was specifically chosen to illustrate how, although the phage values all start the same, the different parameter values and interactions ultimately influence population growth. It was selected to provide context and illustrate how parameter values and interactions ultimately affect the progression of a population.

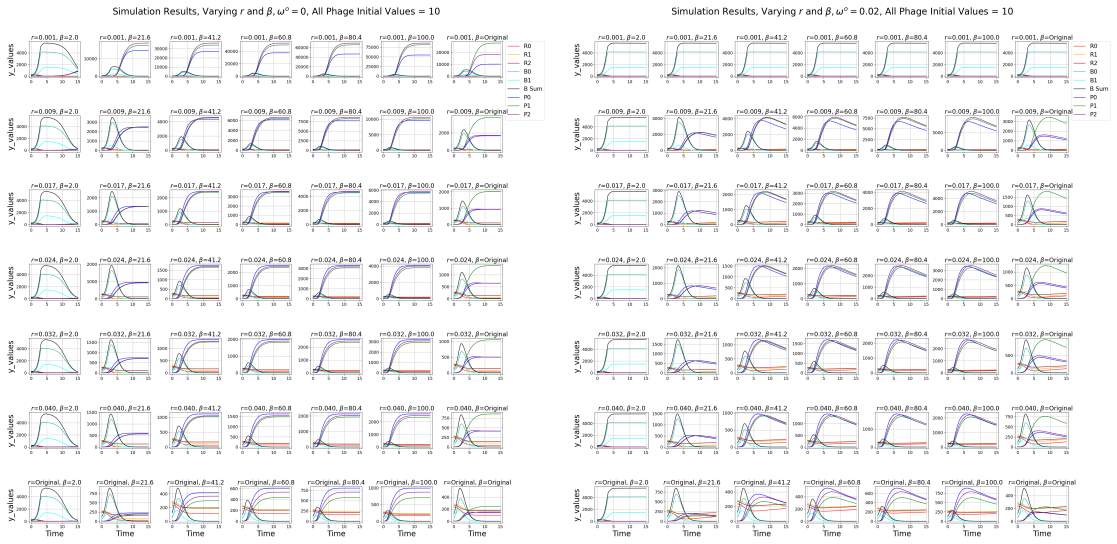
If r or β is equal to Original, then the simulation uses the original parameter values as defined in Table E.4, otherwise each r and each β parameter interaction has the value listed in the subfigure title.

The columns and rows of each figure illustrate how a change in parameter value affects the curve while keeping the other parameters constant. A visualization of which phages can infect which bacteria and which bacteria can consume which resources can be seen in the $3 \times 2 \times 3$ model in Figure 3.1b. P_0 infects B_1, P_1 infects B_0, and P_2 infects B_0 and B_1. B_0 consumes R_0 and R_1, while B_1 consumes R_2. Washin and washout are included, where washin ω_r^i is set such that $\omega_r^i = \omega^o \cdot R_r$, where R_r is the initial resource concentration for resource r.

In $r, \beta, \omega^o = \text{Original}, \text{Original}, 0$, the graph shows how the different parameter values for each interaction uniquely affect the growth rate of each entity, especially the phage population (P_0=blue, P_1=green, and P_2=purple). All phages start at the same population level of 10. P_2 has the fastest initial growth rate until $t = 4$, at which P_1 overtakes P_2. P_2 reaches its peak population count before P_0 or P_1, but despite the slower initial growth, P_0 and P_1 eventually overtake P_2 in total phage population. P_2 also reaches its peak before decreasing in population. The complete extinction of the bacteria has been delayed long enough that, at trace amounts, phage reduction still occurs despite the bacteria's continued existence. The time of peak values for P_0, P_1, and P_2 are $t = 6.33, 7.99, 4.52$, a difference of 3.47 time units. This illustrates how the other parameter values influence the growth and competition of phages for bacteria, as well as the impact these parameters have on the phage populations.

Contrast $r, \beta, \omega^o = \text{Original}, \text{Original}, 0$ (the previous paragraph) with the phage population dynamics of $r, \beta, \omega^o = \text{Original}, 100, 0$, the phage populations show less notable dynamics. The time of peak values are more similar and consistent with one another ($t = 5.50, 7.01, 6.78$, a maximum difference of 1.51-time units). By decreasing the variability of the β values, the system's dynamics have undergone significant changes. Sobol suggested that changing β will not have a large effect on the time of peak values, but when combined with other interactions, the parameter effect can propagate through the network.

The phage population curve all appears the same, with slightly slower growth rates compared to $r, \beta, \omega^o = \text{Original}, \text{Original}, 0$. There is no crossing of phage population count, unlike with $r, \beta, \omega^o = \text{Original}, \text{Original}, 0$. The crossing of phage populations highlights how one phage population performs better than the other, especially when compared to different situations where the cross does not occur.



(a) Washout $\omega^o = 0$.

(b) Washout $\omega^o = 0.02$.

Figure 4.5: Varying adsorption rate r, β, ω^o and ω^i. The default values for the parameters can be found in Table E.4. All initial phage population values were set to 10. Washout values are set such that $\omega_r^i = \omega^o \cdot R_r$.

As the washout term changes, the choice of parameters becomes more critical. The top row of Figure 4.5b ($\omega^o = 0.02$) shows how the phages and resources died out relative to the top row of Figure 4.5a ($\omega^o = 0$). Even with a high burst value, the phages were unable to overcome the pressure from the washout. The phages were not able to proliferate under these conditions.

However, by changing the r value from $r = 0.001$ to $r = 0.009$, with $\beta \geq 21.6$, the phages were able to proliferate at a higher washout rate. Small changes in parameter values can have a drastic impact on outcomes.

The highlighted examples demonstrated the dynamics and influence that multiple agents can have on the final output. These graphs still show some realistic growth curves that can be directly compared against Figure 2.3a. However, with the addition of more interactions, phage and bacterial growth differ, where some phages and bacteria die out faster than others.

4.7 Phage and Bacteria Survivability Analysis For A $20 \times 20 \times 10$ System

In the context of large communities, it is essential to understand how different conditions will influence whether a phage or bacterial population will survive. Different interactions and environmental factors will influence the community's evolution. Sometimes,

populations of phages or bacteria will die out due to external factors, such as a lack of resources or bacteria to consume or infect, natural degradation, or exposure to external factors like UV light.

Using the simulation framework, I created and analyzed a $20 \times 20 \times 10$ system using the adapted Golding model. I selected two parameters, τ and β, to run a survivability analysis using the adapted Golding model.

A phage is considered to have survived if its final population exceeds 1 at the end of the simulation. A bacterial population likewise is considered to have survived if the final population of uninfected bacteria is greater than 1. Washout removes phages and bacteria from the system over time unless a delicate balance of coexistence is maintained. Washin provides a continuous supply of new resources to support bacterial growth, which in turn supports phage infection. Meanwhile, washout prevents the phage population from growing too large and infecting all bacteria.

Each phage is guaranteed to interact with at least one bacterium but no more than two bacteria. Each bacterium interacts with at least one phage and one resource but not more than two phages and two resources. Every resource interacts with at least one bacterium and, at most, three bacteria. The parameter values were randomly selected from a uniform distribution within the Sobol analysis value ranges (Table E.3). Figure 3.1c shows the network interactions.

Figure 4.6 shows the phage and bacteria survivability matrix. It appears that there is an inverse relationship between phage and bacterial survivability. If the phages survived, the bacteria died out. If the bacteria survived, the phages did not survive. The phages only died out with small burst sizes, β, while otherwise would consistently survive. The small burst size ensured that the phages could not proliferate in time. Once the burst size became large enough, around $\beta = 3$, more than half of the phages were able to survive. The bacteria only survived with small burst sizes β. For large burst sizes, the phages would grow too fast and infect the bacteria.

τ had less of an influence in survivability than β did, except for when the β values were less than 20. As τ increases, fewer phages survive, and more bacteria survive. τ determines how fast a bacterium goes through the infection process, so as τ increases, fewer phages will survive. The bacterium takes longer for the infection process to complete, so there is a greater chance for the phages and infected bacteria to be washed out before more phages can be created.

For the most part, the competitive exclusion principle is respected. The majority of the simulations resulted in less than 10 uninfected bacteria surviving. For small burst sizes, there were more than 10 surviving bacteria. However, this is a consequence of

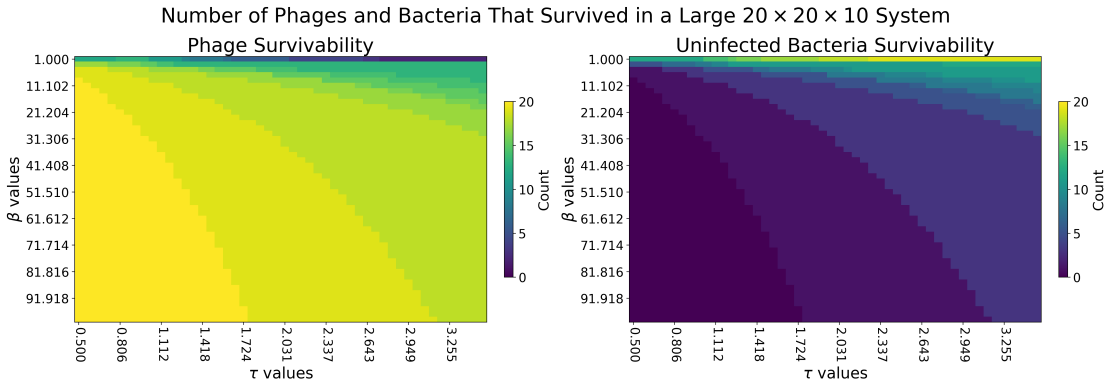


Figure 4.6: The survivability matrix for phages and bacteria by changing τ and β in a $20 \times 20 \times 10$ network. The output graph for the default parameter values for an extensive $20 \times 20 \times 10$ network. Parameter values were randomly chosen in the interval given by Table E.3.

the limited simulation length. I expect more bacterial populations to die out as the simulation length increases.

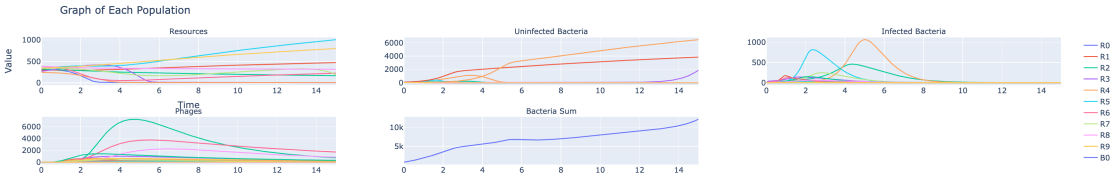
4.7.1 Debris Term

A recently published paper Dey et al. [40] utilizes the Golding model, incorporating a new term: debris. While running their simulations and experiments, the model's results would diverge from their experimental work. They hypothesized that freshly lysed bacteria still have biomarkers that phages can detect and attach to. Incorporating the debris term, which acts as an additional death term for phages, improved the model's alignment with the experimental data. The debris term can also encompass bacterial phage defenses (Section 2.2). The ODE equation from the generalized Golding model that describes the phage behavior can be rewritten as:

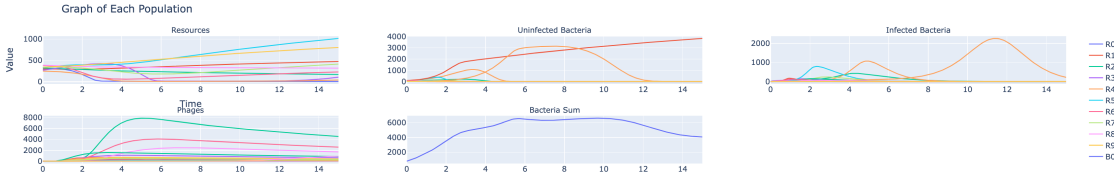
$$\frac{dP_p}{dt} = \sum_{b \in B} \beta_{pb} \cdot \frac{M}{\tau_b} \cdot I_{b_M} - r_{pb} \cdot (U_b + \sum_{k=1}^{M} I_{b_k}) \cdot P_p - w^o \cdot P_p - d_{pb} \cdot P_p$$

Figure 4.7a shows how debris d affected the growth curves of the phages and bacteria in comparison to no debris term, Figure 4.7b. Without the debris term, only one uninfected bacteria survived, and every phage survived except for one phage that died due to the washout. Three uninfected bacteria species reached more than 1,000 uninfected bacteria at any point in the simulation time, and only two infected bacteria species reached more than 1,000 infected. The maximum total bacterial population reached was 6,620 bacteria.

With the debris term included, different results are apparent. Three uninfected bacteria survived, and four different uninfected bacteria strains ever reached a population value of



(a) The $20 \times 20 \times 10$ model with a debris factor included.



(b) The $20 \times 20 \times 10$ model without the debris factor included.

Figure 4.7: A large $20 \times 20 \times 10$ model with a debris parameter added. The debris parameter values were randomly and uniformly selected between 0.01 and 0.2 for each phage-bacteria interaction. Washin of resources and washout is included.

greater than 1,000 at any point in time during the simulation. Only one bacteria strain ever reached more than 1,000 infected bacteria, The ending bacteria value is 12,000, almost double the maximum bacteria population reached without debris. On average, there were significantly fewer phages at the end of the simulation than in the control without debris. No phage population exceeded 2,000 with debris, whereas two phage populations reached more than 2,000 phages without debris.

The debris term more accurately models real communities as phages can randomly deactivate, for example, by UV radiation or phage defense mechanisms.

Chapter 5

Discussion

This section presents an analysis and discussion of the results.

5.1 Graph Behavior

The behaviors shown in Section 4.3 's subsections and the graphs shown in Appendix F.3 represent typical trends observed when varying the parameter values. The graph behavior should be interpreted in conjunction with the local $S1$ sensitivities from Section 4.2 to gain a better understanding of how sensitive the output is to specific parameters and the potential impact of variations in these parameters.

A researcher would examine the growth curve and conclude by visually analyzing the curves and reasoning through the ODE model to explain the behavior. They can further supplement this analysis with a Sobol analysis to quantify the significance of the change and compare the importance of a parameter with that of other parameters. By analyzing the S and $S1$ values, they can also determine whether the parameter acts independently of other parameters or interacts with them. If the parameter interacts with other parameters, then those parameters must be adjusted to observe a significant difference in the output.

5.2 Realistic Growth Curves

As the complexity of the network increases, biologically accurate curves become more complicated. There is not just an exponential and death cycle anymore, but rather a combination of multiple phases for an individual species in a complex community.

This can be observed in real-world complex communities, where multiple phases occur simultaneously.

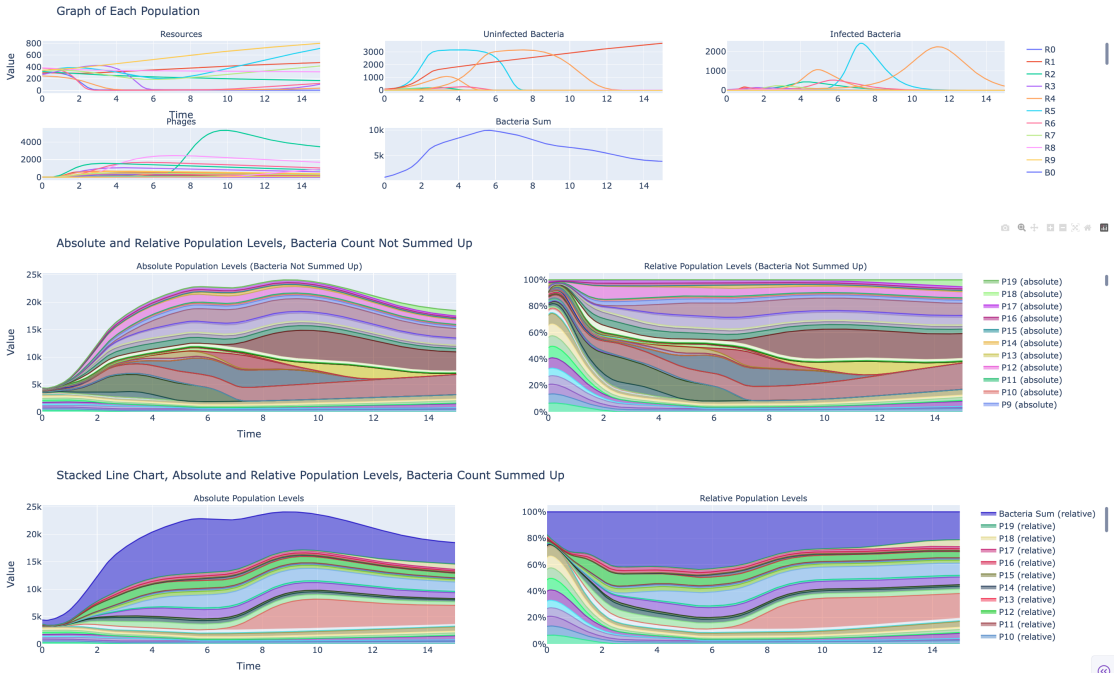
In a $1 \times 1 \times 1$ system, like that in Figure 2.3, there is a clear exponential growth, peak, and death cycle for the bacteria. The phages experience a delay in growth but also exhibit exponential growth. These behaviors can disappear as the community size grows to a $p \times b \times r$ system. Figure 5.1a shows how these behaviors disappear. Some, but not every phage or bacteria in Figure 5.1a exhibits a clear growth and death cycle. Bacteria face increased competition for resources, and phages face increased competition for bacteria. This increased competition decreases the pool of available resources for the bacteria and phages to grow. There is an external threat known as the washout, which cancels out bacterial and phage growth. If the growth is not fast enough, it can even be eliminated.

There are only two interactions that occur in a $1 \times 1 \times 1$ system. With a $p \times b \times r$ system, there are at most $p \cdot b + b \cdot r$ interactions that can occur. Any number of phages can interact with any number of bacteria, and any number of bacteria can interact with any number of resources, each with its unique parameter values. These varying rates will influence the system's dynamics, making it difficult to determine which event caused what due to the increased complexity of the network. With so many individual events co-occurring, it becomes more challenging to identify the cause of the event and how its action will propagate through the network.

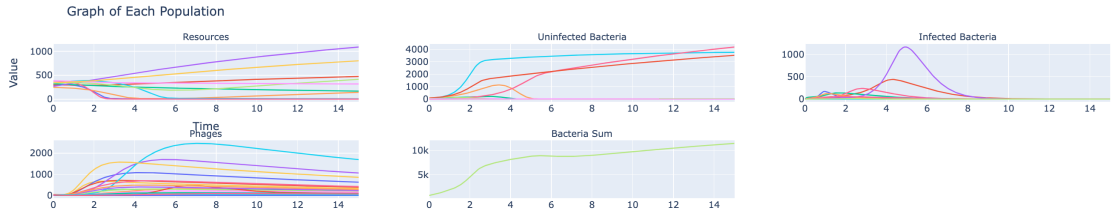
There are ways to circumvent this issue. Simulations offer researchers the ability to experiment with systems that might not be feasible otherwise. Knocking out specific nodes or edges and rerunning the simulation would enable the researcher to observe how the removal of the knocked-out node or edge affected the simulation. The knocked-out node or edge will no longer contribute to the simulation, resulting in an impact on the population count. The larger the difference, the more critical that node or edge was in the simulation.

5.2.1 Knockout

Figure 5.1a (original plot) and Figure 5.1b (knocked out plot) show the difference that knockout has on the output of the graphs. In the original plot, there was more competition for resources and a greater number of phage populations infecting the bacteria. The bacteria died out earlier, so fewer resources were consumed. The washin continued to add more resources to the system, which caused the resource concentration to increase. P_2 and B_{14} were knocked out of the network, using Figure 3.1c as the base network and the parameter values were sampled in the ranges found in Table E.3. In the knocked-out graph, more bacteria were able to survive, and in total, there were more bacteria in the



(a) The output graph for the default parameter values for a $20 \times 20 \times 10$ network. Parameter values were randomly chosen in the interval given by Table E.3. The washin rate continued to add more resources to the system.



(b) The large $20 \times 20 \times 10$ network, with phage P_2 and bacteria B_{14} knocked out from the system.

Figure 5.1: Comparing the effect of knocking P_2 and B_{14} out from the system. Figure a) has P_2 and B_{14} hidden from the system, while Figure b) has the phage and bacteria knocked out from the system. The most important phage and bacteria, as indicated by the growth of bacteria, were knocked out. Figure 3.1c is used as the reference network model. The colors between a) and b) cannot be directly compared, as the color is not mapped to a specific phage or bacterium.

system, as evidenced by the bacteria sum. There are also fewer infected bacteria. With the phages, there is less of a noticeable difference in behavior, but in Figure 5.1a, P_{18}, pink, suddenly increases in concentration at $t = 10$ until the end of the simulation, while this does not happen in Figure 5.1a. P_{18} infects B_{14} (as well as B_13), so by removing B_{14}, P_{18} was negatively affected, and explains why P_{18} couldn't grow.

By removing P_2 and B_{14}, it was demonstrated how the resource, bacterial, and phage dynamics were permanently altered. Figure 5.1b showed how knockouts have a lasting impact on the dynamics of the system, changing which phages and bacteria survive.

5.3 Sobol Sensitivity

A researcher may have an intuitive understanding of how a model works but may be uncertain about the significance of the model's parameters to its results. A researcher can use Sobol to quantify the importance of a change in parameter value in the context of the simulation results. By testing and measuring hundreds of simulation inputs and outputs, Sobol understands how changing a parameter's value influences its output. By identifying the critical parameters, a researcher could use that to their benefit in their research. Researchers can prioritize adjusting the most influential parameters in their simulations or consider engineering phages and bacteria to achieve the desired parameter values.

This section provides a more in-depth discussion of the Sobol results from Section 4.2.

5.3.1 Resources

Every parameter affects how the bacterial and phage populations grow, which in turn influences the consumption rate of resources and ultimately determines the final resource value. Greater resource consumption leads to a lower final resource level. As most resources are typically consumed by the end of the simulation, the final resource value will be 0. However, some parameters, such as τ and β, have a more significant influence on the depletion rate of resources than others. Of all parameters, τ has the most significant influence on the final resource value. τ determines how fast the bacteria will go through the infection process. The longer it takes (i.e., the bigger the value of τ is), the longer it takes for the phage population to grow, allowing more bacteria to grow and consume resources. For smaller τ values, the phages can quickly grow, infect, and kill the bacteria. For large τ values, the bacteria will die early and not consume any resources anymore. There are some combinations of parameters that, when chosen, will result in not all the resources being used up. For example, with Table E.2, if $r = 0.2$, not all the resources have been used up. r defines the probability of a phage infecting a resource. r being so high will cause the phages to infect many bacteria initially. However, due to the high infection rate, most bacteria are immediately infected, preventing population growth and dying out quickly. Therefore, the final resource value will still be relatively high because the bacteria did not have the opportunity to consume the resources.

5.3.2 Phages

r, the adsorption rate of phages to bacteria, is the probability of a successful infection. The smaller the value, the less likely it is to result in a successful infection. With a

larger r value, the likelihood of an infection occurring increases. r is the most important value for determining the final phage value. β, the burst size, influences the final phage population, as the infected bacterium will release β phages into the system. The larger β is, the more phages can be created. The more phages that are created for every lysed bacterium, the more phages are available in the system to infect the bacteria.

The barplot values for the peak value of phages using the 95% rule match the final value plot for the phages. This makes sense as there is no removal of phages from the system, so any phages created by the lysis that are heavily influenced by r and β will stay in the system. As the population continues to increase, the final and peak values will occur near one another (due to the 95% rule) and are closely associated with each other.

τ, the latent infection period, emerges as the most critical parameter, setting the time of peak for the phage population. The τ parameter determines the rate at which an infected bacterium progresses through the infection process. Decreasing tau increases the speed of lysis, allowing more phages to be produced faster. More phages earlier in the system will allow more infections to occur, thereby decreasing the pool of available bacteria. This, in turn, will increase the phage population more rapidly than other parameters, resulting in the phages reaching their peak population faster. r and β have similar effects, but they affect the final and peak phage population rather than the amount of time it takes to reach the peak value. Changing r and β has less impact on the time to peak compared to τ since τ shortens the infection period, directly reducing the time until new phages are created and thereby accelerating phage production.

5.3.3 Total Bacteria

Resources, τ, e, and β all play a critical role in the final population value of bacteria. That being said, more resources allow bacteria to grow for a more extended period, enabling the creation of more bacteria. β plays a crucial role in determining the final bacterial population, whereas r has no significant impact on the final bacterial population. e determines the resource consumption rate. The larger e is, the faster the resource's depletion rate. So, by lowering e, more bacteria will be created, and the time at which the bacteria peak occurs later.

The total bacteria peak population is much more sensitive to the different parameter values. The bacteria act as a link between the phages and resources. Any changes in the resource consumption rate or initial condition will affect the bacteria population. Likewise, any change in phage adsorption rate will affect the bacteria growth, which in turn will affect the resource consumption rate. The bacteria can dampen the effect of specific parameters. For example, the initial resource value affects the final resource

value. The initial resource value partially affects the final bacteria value, but it does not affect the phages. The parameters lose strength as they propagate through the system. Since the bacteria exist between the phages and resources, they are exposed to various parameters that ultimately influence the peak value. Many of these parameters interact with one another; hence, $ST > S1$ is true for many of the inputs.

Similar to the peak value, bacteria interact with multiple parameters that influence other parameters. The peak value, on the other hand, has no limit on the peak value, with a peak value occurring anywhere between 0 and ∞. The time of peak value depends on higher-order interactions for e and K as the bacteria grow at the Monod equation rate. Therefore, e and K (the Monod constant) are only important when combined with changing other parameter values. To find these critical parameters, the third, fourth, fifth, and subsequent orders will need to be calculated to determine which pairs of parameters interact with e and/or K and have the most significant impact on variance in the output.

5.4 Initial Value Analysis

Mulla et al. [6] identify two key processes that dictate the growth of bacteria. Phages adsorb to bacteria, and bacteria lyse bacteria after a time delay. The authors investigate which of the two steps limits the infection cycle rate. They find that there is an analytical solution to the time of peak value in an adsorption limited regime for the bacteria population is $t_{\text{col}} = \frac{1}{r_{\text{bac}}} \log\left(1 + \frac{r_{\text{bac}}}{r_{\text{pha}} b_0} \log\left(\frac{p_\infty}{p_0}\right)\right)$, where p_0 is the initial phage value, p_∞ is the final phage value, $r_{\text{pha}} = n_{\text{burst}} k_{\text{absorb}}, n_{\text{burst}}$ is the burst size, k_{absorb} is the absorption rate, r_{bac} is the bacteria growth rate, b_0 is the initial bacteria concentration. They demonstrate that the time of peak value is primarily limited by phage-bacteria adsorption and that the adsorption rate depends on the bacterial concentration, whereas the burst rate does not. The region is adsorption limited when $k_{absorb} b(t) \ll k_{burst}$. If the system is burst limited, the time of collapse can be calculated as follows: $t_{\text{col}} = \frac{\log\left(\frac{r_{\text{bac}}}{k_{\text{burst}}}\left(\frac{1}{n_{\text{burst}}} + \frac{1}{\text{MOI}_0} - \frac{r_{\text{bac}}}{k_{\text{burst}}}\right)\right)}{k_{\text{burst}} n_{\text{burst}} - r_{\text{bac}}}$ [6].

The behavior between Figure 4.2a and Figure 4.2b should be similar. However, the change in parameter values altered the simulation to introduce a region in behavior change. It would be expected that for 100 initial uninfected bacteria and fewer, the bacteria sum time of peak would follow the linear regression line; however, at around 100 uninfected bacteria, the peak curve deviates from the linear expression.

Between 100 and 500 uninfected bacteria, the system is adsorption limited. The curve follows what was proposed by Mulla et al. [6]. The adsorption of phages to bacteria

depends on the initial bacterial concentration [6]. For larger uninfected bacteria populations, the infection process speeds up as a larger proportion of bacteria can be infected $r \cdot U \cdot P$, where r is the successful phage-bacteria adsorption rate.

The bacteria can grow without immediate pressure from the phages. For small initial resource concentrations and initial bacterial populations, the resources will quickly become depleted. This severely limits the bacteria's ability to grow and artificially caps the population.

The system is latency-limited between 25 and 100 uninfected bacteria. As the initial bacteria decreases, we transition out of the $k_{\text{absorb}}b \ll k_{\text{burst}}$ equality as identified as a condition for the adsorption-limited regime. The phage-to-bacteria ratio increases, allowing them to infect the bacteria more quickly. Therefore, the time to phage peak also decreases as the initial number of uninfected bacteria decreases.

As the uninfected bacteria decreases from 25 to 1, it takes longer for the bacteria to grow and reach their peak population count. At these initial bacterial concentration levels, there are sufficient resources to sustain the bacteria throughout the entire simulation. For uninfected bacteria with fewer than 25, the system enters a new type of restriction, where it experiences a delay in infection due to low encounter rates.

The transition rate from uninfected bacteria U to the first infected stage I_1 is proportional to $U \cdot P$. Fewer bacteria are initially infected when the starting number of bacteria is low. If the phages cannot infect the bacteria, the later infection stages are delayed due to the slow infection process. There could be a threshold for phage to uninfected bacteria where a specific dilution rate will significantly affect the time of peak. The bacteria have a more extended period to grow, as there is a low infection rate and ample resources to consume. This means that it takes longer for the bacteria to grow, as noticed by the increase in time of peak relative to larger initial uninfected bacteria populations. As the bacteria population drives the phage population, an increase in bacteria time of peak causes an increase in phage time of peak.

5.5 Phage Proliferation

If the adsorption rate r is too large and/or the burst size β is too small, the phages will be washed out. Without phage washout, the phages will eventually take over the bacteria, no matter the initial conditions. Ensuring that there are initially sufficient phages in an experiment, as well as an appropriate r and β value to prevent them from disappearing from the system, is crucial. Given the context of the Golding model, the

easiest parameters that a researcher can control are the initial concentrations of phage, bacteria, and resources.

Assuming appropriate r and β (and other) values, researchers must understand how changes in the initial values of phages, uninfected bacteria, and resources influence phage proliferation. If the initial resources or bacterial counts are too low, bacteria will deplete the resources, and the remaining bacteria will consume what remains, limiting bacterial growth. Limited bacterial growth, in turn, limits phage growth, and the system washes the phages away.

5.5.1 Phase Portrait

There is a non-linear trade-off between initial resources and initial phages when a washout is included. The washout non-linearly affects whether the phages proliferate or not. The larger the washout value, the more difficult it is for phages and bacteria to proliferate. The bacteria couple the phage populations to resources, so changes in initial resource concentration will affect the final phage value. Increasing the initial resource concentration leads to an increase in bacteria, while decreasing it results in fewer bacteria. More bacteria ultimately produce more phages, while fewer bacteria ultimately result in fewer phages. While Sobol with washin and washout, Figure F.1b, showed that the final value for phages due to changes in initial resource input values is limited (sensitivity ≈ 0.1), it still has an impact.

For low initial resource concentration values, those below 10, the Monod curve is below the half-growth rate constant (where the growth rate v is 1 and K is 10). The lack of resources restricts the bacteria's growth, and the bacteria will grow at a rate below their maximum growth rate as the available resources limit their growth. As the resource concentration increases towards $K = 10$, the bacteria can grow faster. Since K is small, a slight change in R causes a relatively significant change in the Monod rate.

At around the minimum in the phage proliferation boundary, at around $R = 7$, the behavior changes. As R increases from K, the bacteria are no longer significantly limited by the nutrient concentration and can grow at their maximum rate. However, as R increases beyond K, each additional unit of R results in a diminishing rise in the Monod rate, which asymptotically approaches its maximum growth rate v.

Phage proliferation becomes a race against time under external pressure. The phages will not proliferate if the phage growth rate is not fast enough to overcome the washout removal rate initially or if the infected bacteria are washed out before they lyse.

5.5.2 3D Plot

It is not easy to see inside the matrix, but using the color on the outside can give some insights into the behavior happening inside the matrix. The behavior inside the matrix is uniform with the behavior exhibited on the outside of the matrix. Even with the added bacteria, the phage proliferation boundary remains heavily dependent on the initial phages and initial resources rather than on the bacteria.

5.5.3 Ecology

With the competitive exclusion principle in ecology, there should be at most 10 surviving bacteria in Figure 4.6 [74]. Two or more species competing for the same resources can not coexist indefinitely. The R* resource-ratio hypothesis states that a species must be the best at consuming resources to persist. Populations that require high resource availability will need to either adapt or die out when competing against a population with lower resource consumption. A species can only survive in a steady state at the lowest level of limiting resource R^*, excludes all other species [75]. The species with the lowest R* for a shared limiting resource will outcompete the others and dominate in the long term. The growth rate of a species is proportional to the resources consumed with an energy factor minus a maintenance and death rate. The bigger the resource overlap between competing bacteria, the smaller the realm of coexistence [76]. Populations that are slow at consuming resources reproduce at a slower rate and thus will eventually die out. For a species that consumes multiple resources, the growth of the species is defined by the largest R*, as that resource is limiting the growth.

The Zero Net Growth Isoclines (ZNGI) graph shows how a species transitions between being limited by R_1 to being limited by R_2. The isocline defines the optimal supply ratio at which a bacteria transitions from being limited by R_1 to being limited by R_2. The isocline represents the point where cell growth equals cell death. With multiple isoclines, with each isocline representing a bacteria strain, it is possible to tell which strain will outcompete and proliferate and under which initial resource concentrations coexistence can occur [77].

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Phages have traditionally been viewed as the enemy of bacteria. But with the advent of new transcriptomic, proteomic, genomic, and metabolomic methods, we have gained a better understanding of how phages influence bacterial populations in ecosystems, as well as in laboratories. Data from transcriptomics, proteomics, and metabolomics studies have shown how bacterial responses to phages vary and alter the phenotype of the bacterial host. Data suggests that phages have a net positive effect on bacterial populations. More research is needed to understand how phages and bacteria can coexist, as well as how the removal of a phage, bacteria, or their interactions can influence the dynamics of microbial communities [78].

Understanding the relationship between phages and bacteria, as well as their interaction with the environment, is complex. For a single phage-bacteria pair growing on one resource, there are seven biological parameters $(e, v, K, r, \beta, M, \tau)$ inputs for the Golding model, and six non-biological parameters $(R, U, I, P, \omega^o, \omega^o)$. Although relatively small in comparison to other models, analyzing 13 unique parameters and their interactions takes time and requires an intricate understanding of the model. Finding parameter values that result in high-quality and noteworthy graphs is not an easy task, although knowing the expected values and their biological relevance makes the task easier. Finding a set of parameters that yields behavior worth analyzing takes time.

6.1.1 Simulation Framework

To help aid myself with the task, I created a simulation framework that anyone can use to analyze their own custom model and interaction network. Users can visually create

and edit their interaction networks. Using the dashboard, they can edit the parameter values and run their simulations. The dashboard includes prebuilt visualization tools that enable users to interact with the simulation results. The tool enables users to quickly iterate over and modify parameter values to observe their impact on simulation results. Although the visualizations on the dashboard are designed explicitly for $1 \times 1 \times 1$ models, users can download the complete simulation data and implement their visualization methods to visualize the data over a $p \times b \times r$ system. I use this tool to significant effect in introducing visualization methods, such as the IVA, to identify growth bottlenecks. I take the analysis further by introducing new visualizations, such as a phage proliferation analysis and a survivability matrix.

Using the default interactive graphs on the dashboard and their custom visualizations, users can explore the parameter space to identify notable model behaviors and gain a deeper understanding of the system's dynamics. As evident by Figure 4.2b and Figure 4.2a, different parameter values will lead to contrasting behavior, even if both parameter models replicate realistic growth curves.

6.1.2 Larger Systems

Trying to expand an analysis into a $p \times b \times r$ system becomes even more complicated due to the interconnected nature of the entities. With increasingly larger systems, small changes in a single parameter will often have a minimal influence on the final output. If the parameter value has a substantial influence, it can have a cascading effect on the entire network. As an example, with a $2 \times 2 \times 1$ system, where P_1 infects B_1, and P_2 infects B_1 and B_2, and both bacteria consume R_1. Increasing the infection rate of P_1 will slow the growth of B_1 as B_1 is infected more rapidly. With slower B_1 growth and less uninfected B_1, P_2 is affected as there are fewer B_1 to infect. With a lower P_2 count, B_2 can grow, using more resources. As there are now fewer resources, it is harder for B_1 to grow, so B_2 can grow. Eventually, P_2 starts to infect B_2, so B_2 starts to die, which gives B_1 a chance to consume resources and grow. A self-reinforcing feedback loop starts, where a change in the infection rate has a cascading effect on the rest of the network. The coupled interactions will form a feedback loop, resulting in non-obvious behavior.

6.1.3 Parameterizing Matrices

Parameterizing complex systems is a challenging task. Parameterizing high-dimensional systems can be challenging. In a $p \times b \times r$ model with $p, b, r > 1$, setting a parameter (e.g., β to 35) can be approached in several ways: 1) assign 35 to every element in the

$p \times b$ matrix, 2) randomize values such that their mean is 35, or 3) scale existing values to achieve a mean of 35.

Typically, for bacterial communities, a common choice is to use a random parameter matrix, possibly with a predetermined structure.

There are numerous interactions and parameters in large and complex systems, making it challenging to analyze them effectively. The model network and parameter values are relatively random, and a figure of the network interactions, along with a copy of the parameter inputs, are needed to analyze why a phage or bacterium is behaving in a certain way. Furthermore, it is challenging to simplify or eliminate a phage or bacterium from the model due to the interconnected nature of the system.

6.2 Future Work

The following steps involve collaborating with the researchers running the lab experiments to verify the results, as seen in the output, by comparing the lab results with the model output. With the lab results, the model can be adjusted to better align with the lab findings. The ODE results can replicate the lab results by modifying parameter values or adjusting the model equation. The user can decide to add the Monod microbial growth model to the growth of the bacteria or adapt the Monod equation to being dependent on multiple sources. Using the model, the technicians can improve and validate their methods. If the empirical results significantly deviate from the model results, the technician can theorize what might be happening and alter the model to account for the discrepancy. Dey et al. [40] was able to adapt their model to account for the discrepancy between the model results and the results seen in the lab. They theorized that phages were somehow being deactivated. By adding the debris term, they were better able to account for phage deactivation and achieved a better, more accurate curve fit.

6.2.1 Model Replication

Being able to replicate other models like that of Nilsson [3] would allow me to compare model outputs. A benefit of implementing Cocktail's model is that it would be possible to model multiple bacteria and phages at the same time, as noted as a limitation in Section 2.6.4. Cocktail limits itself to two phages, one bacteria, and one resource. Cocktail supports adding more phages at set times, but only at most three times. This arbitrary limitation can be removed with Cocktail's model implementation.

6.2.2 Debris

Further investigation into the debris and its effects could form the next step in the project. I demonstrated that adding a debris term increased the survivability of bacterial populations on average, resulting in a higher uninfected bacterial population and lower counts of both uninfected and phage populations.

6.2.3 Lab Work

The next logical step is to conduct lab work to generate curves for comparison with the simulation results. If these empirical curves differ significantly from those produced by the current ODE model, a new ODE model can be developed. Curve fitting algorithms can then be employed to estimate the interaction parameters for the revised model numerically. By leveraging the simulation software, researchers can reduce the number of required experiments, thereby saving time, money, and resources.

The lab work would act as an essential model validation step. Dey et al. [40] showed how their ODE model would eventually diverge from the lab-produced ODE curve. They were able to achieve a better curve fit by adapting the model to include the debris term.

Future lab work can also involve identifying the bacteria, phages, and resources present in marine water by analyzing samples collected from the environment. The next step would be to build an interaction network, along with experimentally determining the interaction parameter values through laboratory work. Researchers can predict how the system would behave under new, untested conditions, saving money and time. It may also indicate to researchers if they made an error during testing. Suppose the model indicates that the system should behave in one way, but the system acts differently. In that case, the researcher can review their methods and consider making adjustments to how they conduct the experiments. All in all, having a model that runs in seconds will help researchers gain a better understanding of the system.

6.2.3.1 Environmental Modelling

Many results in research papers come from controlled lab settings. As a next step, researchers can actively collect daily water samples and measure phage, bacteria, and resource concentrations. Collecting samples for over a year would create an ODE-like population curve of the entities. This approach would provide deeper insights into the dynamics of bacterial and phage populations in natural environments, albeit at the expense of losing control over conditions. By continuously monitoring environmental

factors such as hourly temperature, rainfall, and the concentrations of each entity, researchers can gain a deeper understanding of the causal relationships within the ecosystem. A year of experimentation averages out short-term fluctuations in daily measurements, resulting in a smoother overall curve.

The next step would be to use the model to fit and explain observed population dynamics. We want to model how environmental variables (such as temperature, nutrient availability, and rainfall) influence the interactions between phages, bacteria, and resources. Key phenomena could include year-long seasonal cycles, such as dry and wet seasons, rapid growth and decline in population counts, resilience, and the impact of random events, including storms or pollution spikes. By fitting the model to real-world data, we could identify which parameters or interactions are most sensitive to environmental changes and predict how the ecosystem might respond to future events and scenarios. Trying to isolate these communities and run different experiments could be the next step. Analyzing which phages interact with which bacteria or conducting a knockout experiment to investigate how the loss of a bacterium or resource node has a cascading effect on population growth.

Clegg and Gross [79] created a $b \times r$ bacteria-resource model without phages and identified which bacteria consumed which resources. By adjusting the number of resources required for survival, the researchers were able to alter the community's diversity. By adding and removing bacteria-resource interactions to the generalized Golding model, these results could be replicated. Adding more bacteria-resource edges would introduce more competition, resulting in a higher rate of bacterial extinction and a decrease in community diversity. Removing edges would remove competition for resources, and community diversity would increase.

6.3 Other Users

Although I created the simulation framework to facilitate the running and analysis of simulation results, the idea behind this program is that anyone can download the framework's open-source code, edit it, and run their simulations. This tool could be used in an "Introduction to Bacteriology" or "Biological Modelling" course, where the professor would instruct the students to design and implement an ODE model. They would instruct the students to interact with the model as an introduction to modeling phage and bacteria populations. A researcher with weaker programming skills can utilize this low-code tool to gain a deeper understanding of how the system they are analyzing in their lab would behave under different conditions. The program is structured such that users with basic programming skills can create their own analyses by copying and

pasting boilerplate-like code. Appendix D contains a sample of the boilerplate code that implements the ODE model of the generalized Golding model.

Other users and researchers can program their own subset of tools to work with the primary tool. They can use NetworkX to create the graph network programmatically. Since the network is graph-based, they can run network analysis programs on it to identify the node with the most edges (most likely to be important and drive other population growths) and remove that node. The user can create a tool that programmatically removes nodes and edges. Since the simulation framework is object-orientated, the user can interface with the framework from their code and skip the dashboard to automate every step along their simulation pipeline. This would enable users to run a broader range of more unique simulations programmatically and gain a greater understanding of phage-bacteria dynamics.

Chapter 7

Ethics and Data Management

7.1 Ethical Considerations

There are some ethical considerations needed. Phages can be used to treat bacterial infections and may need to be administered under the supervision of a doctor. Attempting to control phages in food or the environment by releasing a phage cocktail into waterways could cause issues further down the line if the released solution contains unwanted chemicals. An imbalance can be introduced into the ecosystem, further exacerbating the problem. An additional step in the food production process will increase food costs and make food production more difficult to control. The cost of creating, maintaining, and using phages at an industrial scale can become substantial and require a significant amount of energy. Dumping phages into the ecosystem could cause issues if the phage concoction includes resources that the bacteria can use, and this can become costly for taxpayers.

7.2 Data Management

All data and results can be found on GitHub. Some simulation data will have to be recreated as the *.parquet* data files are too big for GitHub to store. Measures have been taken to label and document the code, datasets, and parameter configurations, for example, as shown in Appendix E. Any qualified researcher should be able to replicate, audit, use, and edit the computational experiments and code if needed. This systematic approach to version control and storage aligns with best practices, ensuring that results are both traceable and verifiable.

7.3 Adherence to Codes and Principles

I acknowledge that the thesis adheres to the ethical code and research data management policies of UvA and IvI.

The following table lists the data used in this thesis, with the source code. I confirm that the list is complete and that the listed data are sufficient to reproduce the results of the thesis.

Short description (max. 10 words)	**Availability (e.g., URL, DOI)**	**License**
Dataset	Simulation Results	MIT
Source Code	Source Code	MIT
Simulation Conditions	Appendix E and text under figures and in the text	

Appendix A

Appendix A: Equation Parameters

Parameters used in the various equations.

A.1 Simple/Advanced Golding Model Parameters

Variable	Name	Description
R/R_r	Resource entity	Resource r concentration
U/U_b	Uninfected Bacteria population	Uninfected population for bacteria b
I_i/I_{b_i}	Infected Bacteria population	Infected population for bacteria b at stage $1, \ldots, i, \ldots, M$
B/B_b	Bacteria population	Total bacteria population for bacteria b, assuming $B_b = U_b + \sum_{i=1}^{M} I_{b_i}$
P/P_p	Phages population	Phage population for phage p
e/e_{br}	Consumption rate	Rate at which resource r is consumed by bacteria b
β/β_{pb}	Burst size (B matrix)	Lytic burst size for phage p and bacteria b
r/r_{pb}	Successful phage/cell encounter	Probability of a phage p successfully infecting bacteria b
τ/τ_b	Latent period (tau vector)	Time it takes bacteria b to go through the infection stage
v/v_{br}	Maximal growth rate	Growth rate of bacteria b from resource r
K/K_{br}	Monod Constant	Monod constant representing at what resource r concentration at which bacteria b grows at half its maximal rate v
ω^i/ω_r^i	wash-in rate	Rate of resource r being added
ω^o	wash-out rate	Rate of phages, bacteria, and resources being removed, acts on everything proportionally
M	Number of infection stages	Number of infection stages that a bacteria goes through, all bacteria entities have the same value for M
d	Debris	A debris term used to deactivate phages. See Section 4.7.1 for more information.
t	time	time value through the simulation

Table A.1: Golding model parameters (Equation (2.1) and Equation (2.2)) with variables, names, and descriptions. Subscripts on parameters indicate relationships; for example, e_{br} is nonzero if there is an edge connecting bacteria b to resource r in the network, zero otherwise.

A.2 Sobol Parameters

Variable	Name	Description
Y	Univariate parameter output	univariate model output, such as mean μ or variance σ
X	Input vector	Vector of size d, input vector to f
i	Parameter input	Parameter i of input
X_i	Parameter value	Value of vector X at position $i = 1, \ldots, d$, the value of parameter i
d	Input size	Size of input vector X
$X_{\sim i}$	Parameter input	All values of X that are not X_i
f	Function f	Arbitrary black-box function describing model
N	Samples	Number of samples, power of 2, 2^x
D	Parameter input size	Number of parameters inputted into Sobol, $d = \|X\|$
ST_i	Global sensitivity	Contribution of parameter X_i to output variance of Y due to interactions with other variables
$S1_i$	First order sensitivity	Contribution of X_i to output variance of Y

Table A.2: Sobol parameter symbols, name, and description. See Section 3.1.3.2 for the equations.

A.3 Linear Regression Parameters

Variable	Name	Description
a	Slope	Slope of the linear regression line
c	Intercept	y-intercept of linear regression line
R^2	Regression Coefficient	Coefficient of determination of linear regression fit, quality of regression
x_i	Data point	Data point on the x-axis
y_i	Actual Value	Actual value of data for a given x_i
$\hat{y}_i$	Predicted Value	Value predicted of equation for a given x_i
$\bar{y}$	Average Value	Average y value
n	Number of samples	Number of samples being tested

Table A.3: Variable symbol, name, and description used for the linear regression.

Appendix B

Appendix B: Industrial and Real Life Applications of Phages

Due to the nature of killing bacteria, there are numerous applications where a researcher or an organization might be interested in controlling bacterial populations.

A Food Safety Specialist might be interested in introducing a solution containing a high concentration of phages during food production to prevent the spread and growth of *Salmonella* or *E. coli* in the pet food. Alternatively, the Food Safety Specialist might want to promote beneficial bacteria like *Streptococcus thermophilus*, used in the production of Emmental cheese, which heat would kill during the pasteurization process.

A doctor might be interested in providing swallowable pills, more commonly known as phage cocktails, to a patient with a bacterial infection. There is evidence that phage-resistant bacteria are more susceptible to antibiotics; therefore, the doctor might prescribe both medicines to treat the infection effectively.

An Environmental Protection Officer might be interested in seeing how they can use phages to stop the spread of *Cyanobacteria* blooms in waterways, more commonly known as blue-green algae, a photosynthetic microscopic organism that is technically classified as a type of bacteria. This would keep waterways safe for boating and swimming activity, aquatic life, and water consumption in farms, factories, and homes.

When there are a few known bacterial strains, a targeted cocktail of phages can be used to control bacterial population growth in any setting, whether it be food, healthcare, or the environment. Phages offer properties of microbial control that other methods do not, making them an ideal candidate for some applications.

B.1 Controlling Foodborne Bacteria

Foodborne diseases are one of the primary ways for bacteria to spread to humans and animals. Some bacteria use the food as a vector to infect hosts, while others deposit toxins on the food, which is then ingested. If consumed in large enough quantities or further produced in the host, the toxins can be fatal to the host.

Methods exist to control bacterial growth, for example, by storing food at temperatures below 5°C or above 60°C. Bacteria need moisture to grow, so starches like rice will have minimal bacterial growth. Bacteria prefer to live in slightly acidic to neutral pH environments; therefore, having an extremely acidic environment, such as vinegar, will prevent bacterial growth. The use of chemical antibacterial agents, such as bleach, is undesirable due to the potential for leaving residues on food, which can be fatal if ingested. Physical entities like heat or radiation can kill bacteria, but at the cost of altering the food quality [80].

For example, *Streptococcus thermophilus* is one of three different bacteria strains used to create Emmental cheese. However, Emmental cheese does not use pasteurized milk, which increases the risk of *E. coli*. Emmental cheese producers can add phages that target *E. coli* to the milk during the production stage while not affecting the bacteria used to produce the cheese.

B.1.1 Current Applications

Phage cocktails like SalmoFresh^{TM} have been proven to safely reduce *Salmonella* contamination in pet food and raw pet food ingredients [7], as well as in romaine lettuce and bean sprouts [8]. Pet food contains meat and vegetables, where vegetables grown in or on the ground are at risk of *Salmonella* due to contact with soil, manure, compost, and other agricultural runoff from neighboring farms [19]. Figure B.1 and Figure B.2 show how the application of phages has reduced the count of *Salmonella* in ingredients used in pet food as well as romaine lettuce and bean sprouts. In Figure B.1, each food group noticed at least a 68% reduction in CFU/g compared to the control when the 9×10^6 phage treatment was applied. There was at least an 80% reduction in CFU/g across all food groups when treated with a concentration of 9×10^6 or stronger. In Figure B.2, the lettuce and bean sprouts noticed a reduction of at least 0.6 log CFU/mL in *Salmonella* count across all temperature ranges. The most significant reduction in bacterial count in lettuce was observed at 1 hour at 2°C with an absolute reduction of 62.0% between the control and treatment. The most significant reduction in bacteria of 90.0% was observed at 72 hours at 2°C. For the bean sprouts, the lowest reduction in phages was found at 1

hour at 2°C with a reduction of 78.1%, and the most significant reduction was 90.0% at 25°C after 48 hours. Although these values are still above the food-safe threshold, the ability to reduce the *Salmonella* population by at least 62% and up to 90% at various temperatures and incubation periods is impressive. It can prolong shelf life, especially for foods that have short shelf lives before spoiling due to bacterial growth. As such, phages can be shown to control the spread of *Salmonella* in food sources and extend the potential shelf life of certain foods.

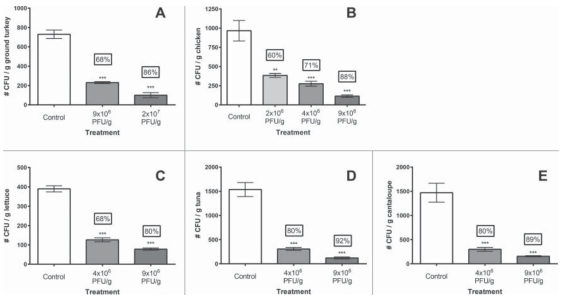


Figure B.1: SalmoLyse® reduces Salmonella contamination on various food surfaces: Mean and standard error bars shown. Statistical analyses were carried out for each food group independently. Asterisks denote significant reduction from corresponding controls based on one-way ANOVA with Tukey's post-hoc tests for multiple corrections: ** denotes $p < 0.01$, while *** denotes $p < 0.001$ compared to the corresponding controls. There was a significant reduction in Salmonella on all food surfaces with the addition of SalmoLyse® compared to the controls; the mean percent reductions from the control are noted in the boxes above treatment bars. CFU/g D colony forming units per gram. Each letter denotes a food group that was tested with SalmoLyse® and compared to a control: A= chicken; B= lettuce; C= tuna; D= cantaloupe; E= ground turkey. Plot sourced from Soffer et al. [7].

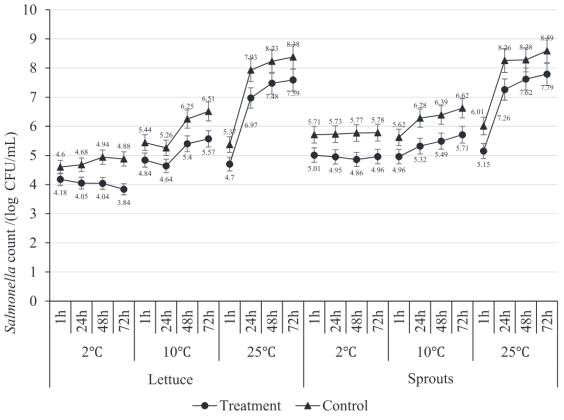


Figure B.2: *Salmonella* count in a mixture of 5 *Salmonella* strains spot-inoculated (CFU/g) onto a) lettuce and b) sprouts after spraying with a mixture of bacteriophage (SalmoFresh™) relative to positive controls at 2, 10 and 25C and stored for 1, 24, 48 and 72 h. Plot sourced from Zhang et al. [8]

B.2 Phage Therapy and Antibiotics

Antibiotics are a common treatment for bacterial infections. However, antibiotics are not selective in the bacteria they kill, killing both harmful and beneficial bacteria. This can lead to the development of antibiotic-resistant bacteria, which makes it harder to combat those bacteria in the future. It has also been demonstrated that antibiotics can harm the gut microbiome and brain development in mice. Phages are an alternative to antibiotics, as they are selective in the bacteria they kill and do not interact with cells or other important biological functions. The rise in antibiotic-resistant bacteria can be attributed to the overuse and overprescription of antibiotics, as well as incorrect usage (for example, prematurely stopping) [15]. These actions exert evolutionary pressure on bacteria to mutate and develop resistance to antibiotics. Phage therapy can contain any number of different phages that target specific bacterial infections, such as *Streptococcus pneumoniae*, with minimal risk of side effects.

B.2.1 Current Applications: Bacterial Infection Control

One active area of research is the use of phages to control bacterial infections. Due to the specificity of phages, they can be used to target specific bacteria strains without affecting other beneficial bacteria. When sick with a bacterial infection, patients swallow antibiotic pills to help the body fight the infection. Antibiotics work by either interrupting intercellular processes like the synthesis of RNA [81], by disrupting the structural integrity of the cell wall [82], or by inhibiting protein synthesis [83].

However, antibiotics are not strain-specific and indiscriminately kill other bacteria as well. Common side effects of antibiotics, although usually not serious, include diarrhea, nausea, and headaches. It has also been shown that the effects of early-stage penicillin exposure in mice have been found to have a long-lasting effect on the gut microbiome, frontal cortex gene expression, and amygdala gene expression [16]. Penicillin increases cytokine expression (small proteins involved in cell signaling) in the frontal cortex of the brain, modifies the integrity of the blood-brain barrier, and alters behavior. The mice exhibited an increase in aggression and anxiety-like behavior [84]. Phages can be used as an alternative to antibiotics, offering benefits without side effects and without affecting the gut microbiome.

With an increase in antibiotic use, there has been a corresponding rise in antibiotic-resistant bacteria. The World Health Organization has stated that antibiotic resistance poses a significant threat to modern medicine and the sustainability of an effective, global public health response to the enduring challenge of infectious diseases. Common

infections that previously would have been easy to treat are more complicated to treat and can increase the risk of disease spread, severe illness, and death [85].

One area of research is exploring how bacteria can exchange traits such as phage resistance and antibiotic resistance. Some bacteria are multi-drug resistant and no longer respond to the medicine.

Laure and Ahn [17] showed evidence that *Salmonella Typhimurium* is more susceptible to ampicillin in the presence of phages, and phage-resistance can lead to reduced virulence and decreased antibiotic resistance.

Zhao et al. [18] showed that there exists an antagonist coevolution between the bacteria and phages, where the dynamics changed from an arms race dynamic (ARD) to a fluctuating selection dynamics (FSD). Due to phage selection and bacterial competition pressure, when bacteria gained phage resistance, they often lost antibiotic resistance. Genome analysis revealed mutations in the textit btuB gene of *Salmonella anatum*, with a higher mutation frequency during the ARD stage. A knockout experiment confirmed that the btuB gene is a receptor for the *JNwz02* phage, resulting in reduced bacterial competitiveness. Further analysis detected multiple single nucleotide polymorphism (SNP) mutations in the phage-resistant strains. The SNPs potentially affected the membrane components, partially weakening the cell defense against antibiotics. These findings help advance our understanding of phage-host-antibiotics interactions and the impact of adaptations to antibiotic resistance. The research demonstrates how phages can be utilized to reintroduce antibiotic susceptibility to previously insusceptible bacteria, thereby preventing costly and lengthy research on new antibiotics [18].

Phage research is facing challenges due to bacterial strains evolving resistance to phages. Understanding the interplay between antibiotics and phages is essential for shaping future research [18].

B.3 Environmental Protection

Algae blooms, also called red tides, is the rapid spread of bacterial or algae organisms. Blooms are a growing environmental concern impacting water quality, aquatic ecosystems, and human health. These rapid increases in algae populations, often fueled by excess nutrients such as nitrogen and phosphorus, can occur in freshwater, coastal, and marine environments.

Cyanobacteria blooms have significant impacts on both the aquatic environment and human health. Cyanobacteria release nitrogen and phosphorous, which the bacteria use

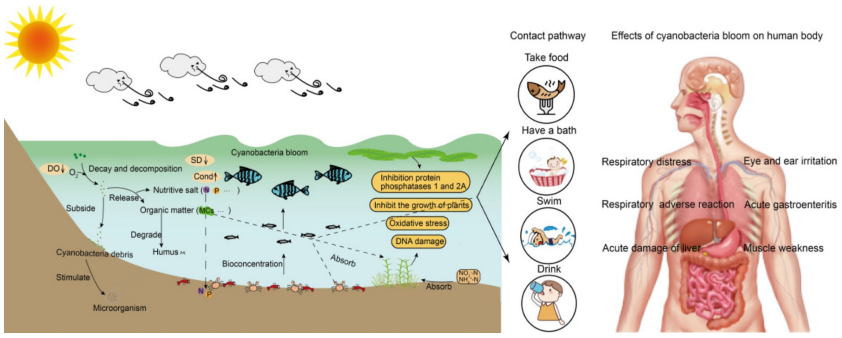


Figure B.3: Cyanobacteria degradation cycle, main hazards of cyanobacteria bloom to water bodies, aquatic organisms, and the human body. (DO: dissolved oxygen; SD: water transparency; Cond: conductivity; N: nitrogen; P: phosphorus; MCs: microcystins). [9]

to grow with oxygen, outpacing other aquatic growth and killing aquatic marine life. Bacterial toxins can make their way into the food and water consumed by humans, causing muscle fatigue, respiratory issues, liver damage, and gastrointestinal issues [9]. Figure B.3 shows the process of how cyanobacteria degrade and are absorbed into the environment, eventually making their way into the human body via various contact points.

B.3.1 Current Applications

There is interest in using phages to control cyanobacteria blooms. Phages can offer better and safer alternatives to chemical options when attempting to control bacterial blooms. Chemical options are indiscriminate, killing cyanobacteria while also killing other beneficial bacteria and aquatic life, and can eventually seep into groundwater. Although not used to control bacteria blooms, some chemicals like PFAS, also called "Forever Chemicals", can last a long time in the environment and do not degrade and keep on negatively affecting the environment. Due to the specificity of phages, only cyanobacteria will be targeted, and this will not affect the surrounding environment.

Tucker and Pollard found that an isolated phage cocktail collected from Lake Baroon in Australia could decrease the abundance of *M. aeruginosa* by 95% within 6 days in a lab setting, before recovering within 3 weeks [14].

There is evidence that phage-resistant bacteria can influence the population dynamics of other bacteria. It has been shown that the plankton level has been experimentally affected by the frequency of the phage-resistant *Nodularia* marine bacteria. Populations with high phage resistance ($> 50\%$) dominate the plankton communities despite a high phage count and eventually outcompete other bacteria due to their slower decline in

population size. Contrastingly, populations of bacteria with low phage resistance (between 0% and 5%) were lysed to extinction, releasing resources such as nitrogen. This allows for other bacterial strains to absorb the resources and dominate the bacterial community. Phages and the lysis of bacterial strains can have a dramatic effect on the community dynamics and composition of other entities, such as phages, bacteria, and resources [13]. Phages have the potential to be used as a specific strategy for controlling cyanobacterial blooms with minimal environmental impact, offering control of bacterial blooms with limited environmental impact. Usage should be safe, novel, efficient, and sensitive.

However, there are issues with using phages to control bacterial blooms. Bacterial blooms can cover vast areas or occur in areas that are difficult to reach, such as marshlands; applying phages to combat the bloom may be infeasible. If the method of choice were to spray a solution of water containing phages, the solution would need to be shipped to the site and loaded onto special boats to spray the solution into the water, or the trucks would need to drive along the shore and spray the solution into the water. This solution may not address the root cause, for example, the constant discharge of wastewater into a river.

The phage density in the solution must be relatively high to combat the bloom quickly. These problems pose significant logistical challenges in creating the phages in a lab or factory, transporting them, and administering the phages to the waterways. Phages can only diffuse through the water and cannot actively swim, so they are dependent on the rate of diffusion and water currents. This will be difficult in marshlands, where the bacteria can "hide" in the grass and crevices created by aquatic life. If the bloom occurs in a high-current area, such as a river or a bay, the water can wash the phages away.

Scientists have not yet fully understood the phage infection mechanism, and research into the artificial engineering of phages is limited, making it challenging to conduct studies in this area [86, 87].

Algae can produce toxins that threaten wildlife, contaminate drinking water, and disrupt local economies dependent on fishing and tourism. In the state of Florida, between the years 1995 and 2000, the restaurant and hotel industry lost an estimated $6.5 million to algae blooms. This accounts for approximately 25% of the average total monthly sales revenue in the region from June through October, the months most commonly affected by red tide[88]. During a red bloom event, hospital diagnoses in the county of Sarasota for pneumonia, gastrointestinal, and respiratory illness increased by 19%, 40% and 54% respectively [89, 90], with a respiratory illness visit costing between $0.5 and $4 million [91].

Appendix C

Appendix C: Flowchart of User and System Interactions

Figure C.1 shows how the user can interact with the system, the input and outputs for subsystems, and the systems working with one another. To read the flow chart, start from the top to the bottom. First, the user creates a network using the GUI Network Creation Tool. After the graph is complete, the user provides an implementation of the network as an ODE model in Python. Once finished, the user provides the network file and ODE model to the ODE solver. The solver uses information from the network file to determine the number of entities to create, parameter details (including names, values, and dimensions), and setting values. Then, the user interacts with the Visualization Dashboard Tool, for example, by clicking on buttons to run simulations, changing parameter values, (un)selecting checkboxes, zooming in and out of plots, and hovering over plots to show data. Once a user has selected the parameter values, they are sent to the solver. The solver calculates the time and population values using the provided graph and ODE model and then sends the data back to the Visualization Dashboard Tool, which outputs the visualizations. If the user has run an Ultimate Analysis, they can query the saved data to create their custom visualizations.

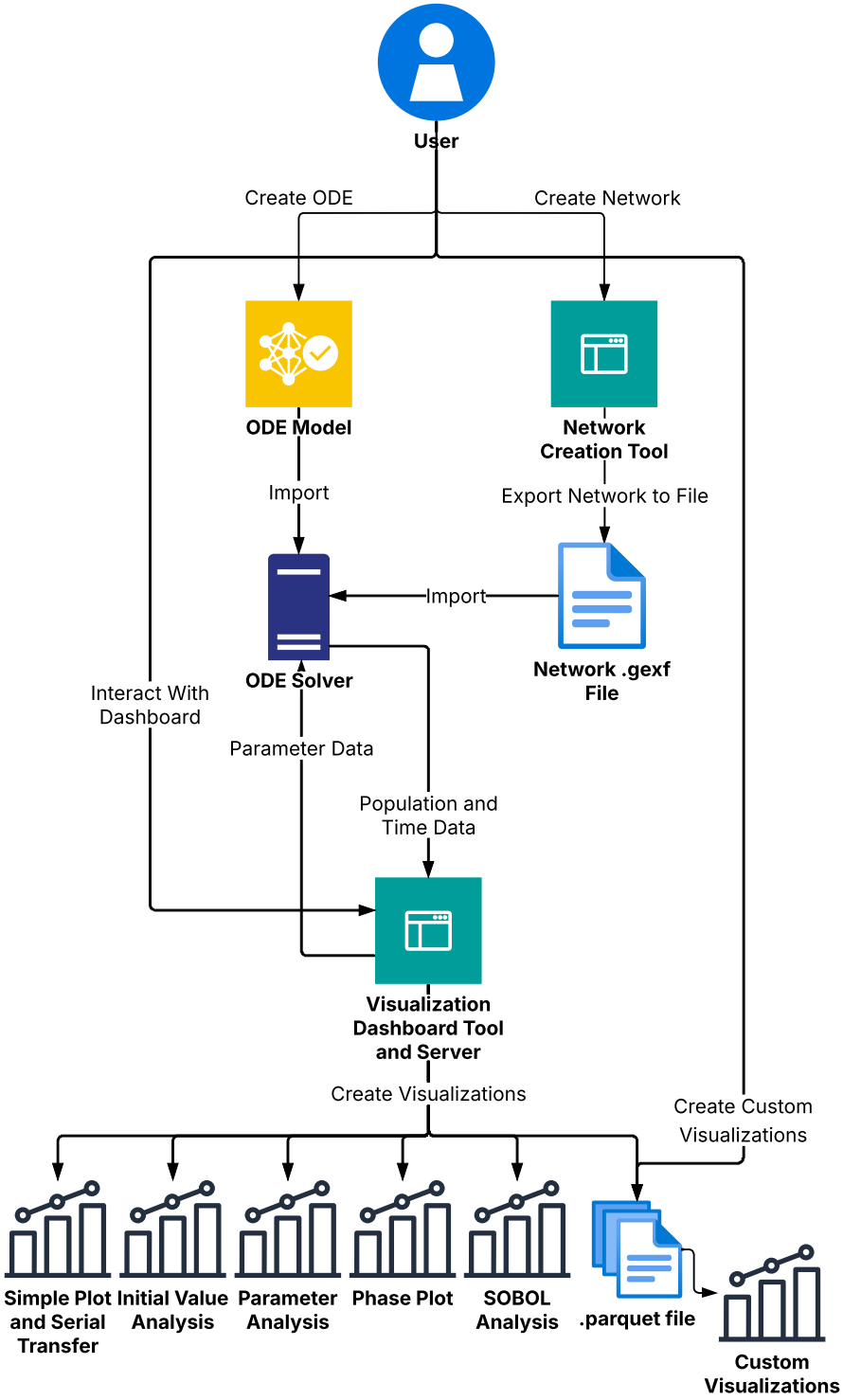


Figure C.1: The flowchart of user and system interactions. Read from top to bottom.

Appendix D

Appendix D: ODE Model Implementation

The code listed here is the implementation of the adapted Golding model, Equation (2.2) along with the setup and running of the visualization tool.

```
import numpy as np
from Classes.Analysis import Analysis
from Classes.GraphMakerGUI import GraphMakerGUI
from Classes.Visualizer import Visualizer
np.random.seed(0)

# use base class Analysis to create a new class System
class System(Analysis):
    def __init__(self, graph_location):
        """The ODE representation of the Golding Model from 'Using population dynamics to count bacteriophages and their lysogens' from Yuncong Geng, Thu Vu Phuc Nguyen, Ehsan Homaee, and Ido Golding.
        Uses Classes.Analysis as a base class.
        Args:
            graph_location (str): location of the graph file
        """
        super().__init__(graph_location)

    def odesystem(self, t, Y, *params):
        """The system of ODEs that represent the Golding Model.
        Args:
            t (float): The time value that the solver is currently at.
            Y (np.array): The initial (for t=0)/current (for t>0) population values of the system. A 1D array of length equal to the number of variables in the system.
        """
        def g(R, v, K):
```

```
24          """Calculate the growth rate of the bacteria. The growth rate
     is a function of the concentration of the resource, the growth rate
    of the bacteria, and the carrying capacity of the bacteria.
25          Args:
26              R (float): Resource concentration
27              v (float): max rate of growth
28              K (float): carrying capacity
29          Returns:
30              float: The growth rate of the bacteria. v*R/(R+K)
31          """
32          return (R * v) / (R + K)
33      # Unpack the parameters. Need to get the graph network, the list
    of phage/bacteria/resource node names, value of M, the vectors (tau
    and washin), and the matrices (e, v, K, r, B), and the environment
    settings
34      # graph_object, phage_nodes, bacteria_nodes, resource_nodes, M,
    tau_vector, washin_vector, e_matrix, v_matrix, K_matrix, r_matrix,
    B_matrix, environment = params
35      graph_object, phage_nodes, bacteria_nodes, resource_nodes, M,
    tau_vector, washin_vector, e_matrix, v_matrix, K_matrix, r_matrix,
    B_matrix, debris, environment = params
36      graph = graph_object.graph
37
38      Y = self.check_cutoff(Y) # check to see if any values are really
    small. If yes, set to 0
39      R, U, I, P = self.unflatten_initial_matrix(Y, [len(resource_nodes
    ), len(bacteria_nodes), (len(bacteria_nodes), M), len(phage_nodes)]) #
     Turn Y into the shape of the system.
40      new_R = np.zeros_like(R) # create fresh copies of the arrays to
    be updated.
41      new_U = np.zeros_like(U)
42      new_I = np.zeros_like(I)
43      new_P = np.zeros_like(P)
44
45      #update N vector
46      for resource in resource_nodes: # loop over the resource names
47          r_index = resource_nodes.index(resource) # get the index of
    the phage
48          washin = washin_vector[r_index]
49          sum = 0
50          for bacteria in bacteria_nodes: # loop over the bacteria
    names
51              b_index = bacteria_nodes.index(bacteria)
52              if graph.has_edge(bacteria, resource): # important to
    check if the edge between bacteria_b and resource_r exists.
53                  e = e_matrix[b_index, r_index] # get the associated
    values for this interaction
54                  v = v_matrix[b_index, r_index]
```

```
55                  K = K_matrix[b_index, r_index]
56                  U_b = U[b_index] # uninfected
57                  I_b = np.sum(I[b_index]) # sum of all infected
    bacteria b agents
58                  sum += e * g(R[r_index], v, K) * (U_b + I_b)
59          new_sum = -sum - R[r_index] * environment['washout'] + washin
     # calculate the new value of the resource.
60          if new_sum <= 0 and R[r_index] <= self.cutoff_value: # if the
     new sum is negative and the resource population is greater than 0,
    set it to 0
61              new_R[r_index] = 0
62          else:
63              new_R[r_index] = new_sum
64
65      # update U vector
66      for uninfected_bacteria in bacteria_nodes:
67          u_index = bacteria_nodes.index(uninfected_bacteria)
68          p_sum = 0
69          g_sum = 0
70          for resource in resource_nodes: # loop over the resource
    names
71              r_index = resource_nodes.index(resource) # get index of
    the resource
72              if graph.has_edge(uninfected_bacteria, resource):
73                  g_sum += g(R[r_index], v_matrix[u_index, r_index],
    K_matrix[u_index, r_index])
74          for phage in phage_nodes: # loop over the phage names
75              p_index = phage_nodes.index(phage) # get index of the
    phage
76              if graph.has_edge(phage, uninfected_bacteria): # check if
     the edge between phage_p and bacteria_b exists.
77                  p_sum += r_matrix[p_index, u_index] * P[p_index]
78          # update the uninfected bacteria
79          new_sum = U[u_index] * g_sum - U[u_index] * p_sum - U[u_index
    ] * environment['washout']
80          if new_sum <= 0 and U[u_index] <= 0: # if the new sum is
    negative and the uninfected population is greater than 0, set it to 0
81              new_U[u_index] = 0
82          else:
83              new_U[u_index] = new_sum
84
85      # update I vector
86      for infected_bacteria in bacteria_nodes:
87          if (P[p_index] <= 0):
88              continue
89          i_index = bacteria_nodes.index(infected_bacteria)
90          for k_index in range(0, M):
```

```
91              if k_index == 0: # if we are at the first stage of
    infection
92                  p_sum = 0
93                  for phage in phage_nodes: # loop over the phage names
94                      p_index = phage_nodes.index(phage) # get index of
     the phage
95                      if graph.has_edge(phage, infected_bacteria): #
    check if the edge between phage_p and bacteria_b exists.
96                          p_sum += r_matrix[p_index, i_index] * P[
    p_index]
97                  M_tau = 0 if tau_vector[i_index] == 0 else M /
    tau_vector[i_index]
98                  new_sum = U[i_index] * p_sum - M_tau * I[i_index, 0]
    - environment['washout'] * U[i_index]
99                  if new_sum <= 0 and I[i_index, 0] <= 0: # if the new
    sum is negative and the infected population is greater than 0, set it
    to 0
100                     new_I[i_index, 0] = 0
101                 else:
102                     new_I[i_index, 0] = new_sum
103             else: # if we are at the other stages of infection
104                 if(tau_vector[i_index] == 0): # prevent divide by 0
    error
105                     M_tau = 0
106                 else:
107                     m_tau = M / tau_vector[i_index] # get the value
    of M_tau
108                 right = I[i_index, k_index - 1] - I[i_index, k_index]
109                 new_sum = m_tau * right - environment['washout'] * I[
    i_index, k_index]
110                 if new_sum <= 0 and I[i_index, k_index] <= 0: # if
    the new sum is negative and the infected population is less than 0,
    set it to 0
111                     new_I[i_index, k_index] = 0
112                 else:
113                     new_I[i_index, k_index] = new_sum
114
115         # update P vector
116         for phage in phage_nodes: # loop over the phage names
117             p_index = phage_nodes.index(phage) # get index of the phage
118             left_sum = 0 # initialize sums
119             right_sum = 0
120             debris_sum = 0
121             for infected_bacteria in bacteria_nodes: # loop over the
    bacteria names
122                 i_index = bacteria_nodes.index(infected_bacteria) # get
    index of the bacteria
```

```
123                 if graph.has_edge(phage, infected_bacteria): # check if
    the edge between phage_p and bacteria_b exists.
124                     if (tau_vector[i_index] == 0): # prevent divide by 0
    error
125                         M_tau = 0
126                     else:
127                         M_tau = M / tau_vector[i_index] # get the value
    of M_tau
128                     left_sum += B_matrix[p_index, i_index] * M_tau * I[
    i_index, -1]
129                     right_sum += r_matrix[p_index, i_index] * (U[i_index]
     + np.sum(I[i_index]))
130                     # debris_sum += debris[p_index, i_index] * P[p_index]
     # get the debris value for this phage and bacteria interaction
131             # update the phage value
132             new_sum = left_sum - right_sum * P[p_index] - environment['
    washout'] * P[p_index] - debris_sum
133             if new_sum <= 0 and P[p_index] <= 0: # if the new sum is
    negative and the phage population is greater than 0, set it to 0
134                 new_P[p_index] = 0
135             else:
136                 new_P[p_index] = new_sum
137
138         # flatten the new updated initial conditions, undoes the
    flattening done by unflatten_initial_matrix().
139         flattened_y1 = self.flatten_lists_and_matrices(new_R, new_U,
    new_I, new_P)
140         return flattened_y1
141
142
143 # graph = GraphMakerGUI(seed=0) # create a new object using the GUI tool.
144 # system = System('a_good_curve.gexf') # load the graph from the file.
145 # system = System('a_good_curve_2.gexf') # load the graph from the file.
146 system = System('complex_graph.gexf') # load the graph from the file.
147 # system = System('large_graph.gexf') # load the graph from the file.
148 # system = System('large_graph_knocked_out_P2_B14.gexf') # load the graph
     from the file.
149
150 phage_nodes = system.get_nodes_of_type('P') # get the phage nodes
151 bacteria_nodes = system.get_nodes_of_type('B') # get the bacteria nodes
152 resource_nodes = system.get_nodes_of_type('R') # get the resource nodes
153 environment_nodes = system.get_nodes_of_type('E') # get the environment
    nodes
154
155 # get the 'Initial_Condition' attribute values from the nodes. Saves as
    vector.
156 R0 = system.initialize_new_parameter_from_node("Initial_Concentration",
    resource_nodes)
```

```
U0 = system.initialize_new_parameter_from_node("Initial_Population",
    bacteria_nodes)
I0 = system.initialize_new_matrix(len(U0), system.M)
P0 = system.initialize_new_parameter_from_node("Initial_Population",
    phage_nodes)
# get the 'tau' and 'washin' values from the bacteria and resource nodes.
     Saves as vector
tau_vector = system.initialize_new_parameter_from_node('tau',
    bacteria_nodes)
washin = system.initialize_new_parameter_from_node('washin',
    resource_nodes)

# get the 'e', 'v', 'K', 'r', and 'B' values from the edges between the
    listed nodes. Saves as matrix.
e_matrix = system.initialize_new_parameter_from_edges('e', bacteria_nodes
    , resource_nodes)
v_matrix = system.initialize_new_parameter_from_edges('v', bacteria_nodes
    , resource_nodes)
K_matrix = system.initialize_new_parameter_from_edges('K', bacteria_nodes
    , resource_nodes)
r_matrix = system.initialize_new_parameter_from_edges('r', phage_nodes,
    bacteria_nodes)
B_matrix = system.initialize_new_parameter_from_edges('Burst_Size',
    phage_nodes, bacteria_nodes)

visualizer = Visualizer(system) # start the visualizer system.

# add the initial conditions to the visualizer, with a name, the initial
    conditions, and the node names.
visualizer.add_graph_data("Resources", R0, resource_nodes)
# create an uninfected 'hidden' agent
visualizer.add_graph_data("Uninfected Bacteria", U0, bacteria_nodes)
# create infected 'hidden' agent. provide row names, as well as column
    names.
visualizer.add_graph_data("Infected Bacteria", I0, column_names=[f"
    Infected Stage {i}" for i in range(int(system.M))], row_names=[f"
    Infected B{i}" for i in range(len(bacteria_nodes))], add_rows=int(
    system.M))
visualizer.add_graph_data("Phages", P0, phage_nodes)

# add the vector parameters to the visualizer, with a name, the parameter
     values, and the node names.
visualizer.add_non_graph_data_vector("tau_vector", tau_vector,
    bacteria_nodes)
visualizer.add_non_graph_data_vector("washin", washin, resource_nodes)

# add matrix parameters to the visualizer, with a name, the parameter
    values, and the node names.
```

```
186 visualizer.add_non_graph_data_matrix("e_matrix", e_matrix, bacteria_nodes
        , resource_nodes)
187 visualizer.add_non_graph_data_matrix("v_matrix", v_matrix, bacteria_nodes
        , resource_nodes)
188 visualizer.add_non_graph_data_matrix("K_matrix", K_matrix, bacteria_nodes
        , resource_nodes)
189 visualizer.add_non_graph_data_matrix("r_matrix", r_matrix, phage_nodes,
        bacteria_nodes)
190 visualizer.add_non_graph_data_matrix("B_matrix", B_matrix, phage_nodes,
        bacteria_nodes)
191 matrix1 = np.random.uniform(low=0.01, high=0.2, size=(len(phage_nodes),
        len(bacteria_nodes)))
192 visualizer.add_non_graph_data_matrix("debris_matrix", matrix1,
        phage_nodes, bacteria_nodes)
193
194 # optionally add other parameters to the visualizer, that will be passed
        to the ODE system.
195 visualizer.add_other_parameters(phage_nodes, bacteria_nodes,
        resource_nodes, int(system.M))
196
197 visualizer.run() # run the visualizer/dashboard
```

Appendix E

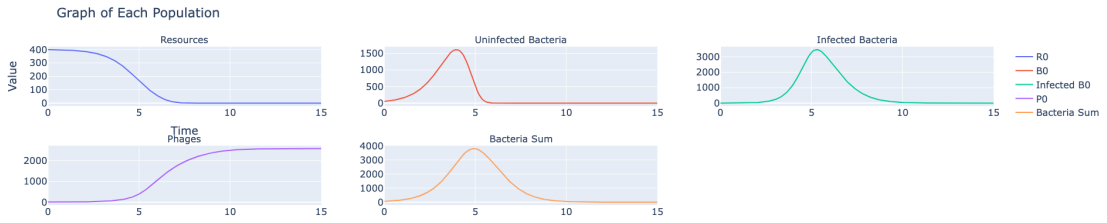
Appendix E: Parameter Values Used

E.1 Realistic Growth Curves

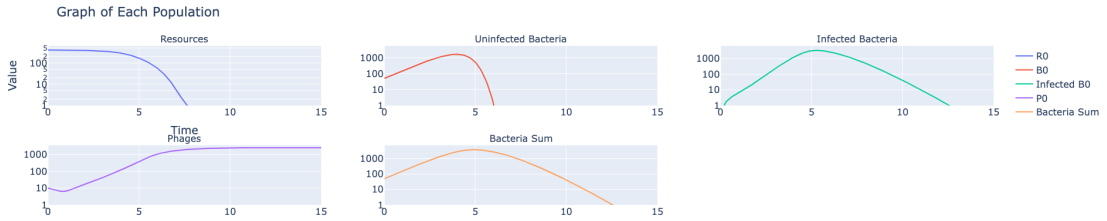
IC					
Resources	**Uninfected Bacteria**			**Infected Bacteria**	**Phages**
400	50			$\begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix}$	10
Vector Data					
$\boldsymbol{\tau}$	$\boldsymbol{\omega^i}$				
2.14	0				
Matrix Data					
$\boldsymbol{e}$	$\boldsymbol{v}$	$\boldsymbol{K}$	$\boldsymbol{r}$	$\boldsymbol{\beta}$	
0.03	1.2	10	0.01	20	
Environment Data					
$\boldsymbol{M}$		$\boldsymbol{\omega^o}$			
4		0			

Table E.1: The parameter values used for Figure 2.3.

E.2 A Second Realistic Growth Curve



(a) A second realistic growth curve, linear y-axis



(b) A second realistic growth curve, logarithmic y-axis.

Figure E.1: The parameters used for this plot can be found in Table E.1.

IC				
Resources	**Uninfected Bacteria**		**Infected Bacteria**	**Phages**
200	50		$\begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix}$	10
Vector Data				
$\boldsymbol{\tau}$	$\boldsymbol{\omega^i}$			
0.7	0			
Matrix Data				
$\boldsymbol{e}$	$\boldsymbol{v}$	$\boldsymbol{K}$	$\boldsymbol{r}$	$\boldsymbol{\beta}$
0.12	1	10	0.001	10
Environment Data				
$\boldsymbol{M}$		$\boldsymbol{\omega^o}$		
4		0		

Table E.2: Another set of realistic growth curves. The linear and logarithmic plot of this data can be seen in Figure E.1.

E.3 Sobol Analysis

IC				
Resources	**Uninfected Bacteria**		**Phages**	
1-500	1-100		1-50	
Vector Data				
$\boldsymbol{\tau}$	$\boldsymbol{\omega^i}$			
0.5-3.5	0-100			
Matrix Data				
$\boldsymbol{e}$	$\boldsymbol{v}$	$\boldsymbol{K}$	$\boldsymbol{r}$	$\boldsymbol{\beta}$
0.05-0.25	0.8-1.9	10-250	0.001-0.2	1-100
Environment Data				
$\boldsymbol{\omega^o}$				
0-0.1				

Other Data				
Seed Value	**2nd Order**	**Number Samples**	**Simulations Run**	**Simulation Length**
0	False	15	$2^{15}(9+2) = 360448$	15

Table E.3: The parameter values used for the Sobol sensitivity analysis in Figure 4.1 (Sobol analysis without washin and washout) and Figure F.1 (Sobol analysis with washin and washout). For Sobol analysis with washin and washout, there are $2^{15}(9+2) = 425984$ unique simulations run.

E.4 Parameter Values for a $3 \times 2 \times 3$ Model

IC

Resources	**Uninfected Bacteria**	**Infected Bacteria**	**Phages**
$\begin{bmatrix} 236 & 287 & 270 \end{bmatrix}$	$\begin{bmatrix} 53 & 69 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 10 & 5 & 8 \end{bmatrix}$

Vector Data

$\boldsymbol{\tau_b}$	$\boldsymbol{\omega_r^i}$
$\begin{bmatrix} 2.73340 & 2.25015 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$

Matrix Data

$\boldsymbol{e_{br}}$	$\boldsymbol{v_{br}}$
$\begin{bmatrix} 0.15680 & 0.10871 & 0 \\ 0 & 0 & 0.18009 \end{bmatrix}$	$\begin{bmatrix} 1.27601 & 0.86393 & 0 \\ 0 & 0 & 1.22625 \end{bmatrix}$

$\boldsymbol{K_{br}}$	$\boldsymbol{r_{pb}}$	$\boldsymbol{\beta_{pb}}$
$\begin{bmatrix} 139.58353 & 12.83058 & 0 \\ 0 & 0 & 82.86684 \end{bmatrix}$	$\begin{bmatrix} 0 & 0.11695 \\ 0.144459 & 0 \\ 0.11895 & 0.13065 \end{bmatrix}$	$\begin{bmatrix} 0 & 15 \\ 34 & 0 \\ 11 & 57 \end{bmatrix}$

Environment Data

$\boldsymbol{M}$	$\boldsymbol{\omega^o}$
4	0

Table E.4: The parameter values used for the $3 \times 2 \times 3$ network model rounded to 5 decimal points. If there is no edge between a phage, bacteria, or resource, then in the matrix representation of the parameter, 0 is stored as the default value. The default graph output can be seen in Figure 5.1a.

Appendix F

Appendix F: Extra Plots and Figures

F.1 Sobol Analysis With Washin and Washout

F.1.1 Final Value Analysis

F.1.1.1 Resources

Despite the resource consumption rate directly depending on e, v, and K, the parameters had very little influence on the final value, as evidenced by the ST and $S1$ bars being near 0. The washin and washout values mainly drove the resource population. The peak resources are driven entirely by the washin rate. There were few interactions between two or more parameters.

F.1.1.2 Phages

The most important factor for the final phage value is r, followed by β and ω^o. The other parameters had little to no effect on the final phage value.

F.1.1.3 Total Bacteria

The sum of uninfected and infected bacteria depended heavily on higher-order interactions as $ST_i \gg S1_i$. Although not shown, the bar plots for the total bacteria resembled those of the bar plots for the uninfected bacteria, and were less than those of the infected bacteria.

F.1.2 Peak Value and Time of peak

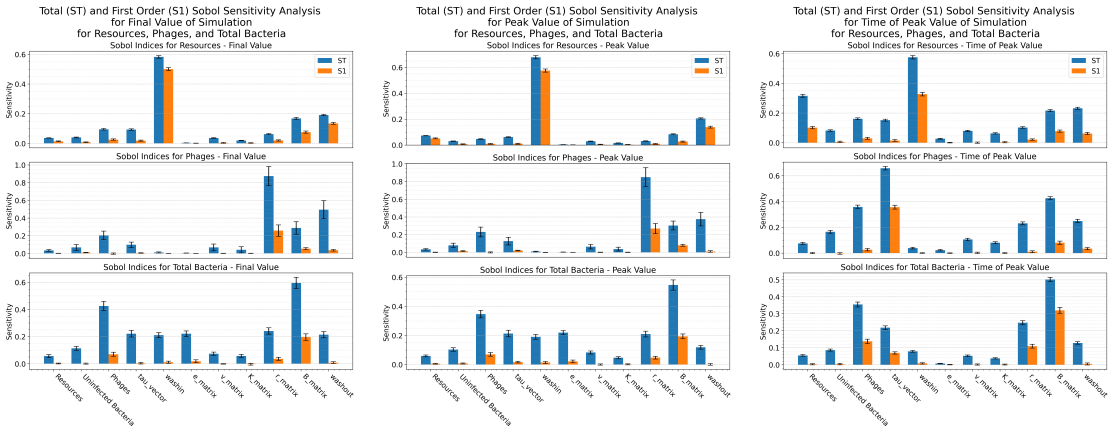
To create the custom Sobol analyses, the peak value and the time at the peak of the population is measured and analyzed. The peak is defined as the point at which the population reaches 95% of its absolute maximum value. The time at peak is measured at the point in time that the population reaches 95% of the maximum value. This removes unintended side effects of the simulation. For populations that are only increasing in value, this prevents the measured peak from clustering at the end of the simulation, thereby skewing the data. As the peak is defined at 95% of the absolute maximum value, populations that experience a faster increase in population count at the end will have a time value closer to the end of the simulation. For populations that reach a plateau, the 95% rule will push the time of peak towards the beginning of the simulation while still "respecting" the absolute final value since 95% $\approx$ 100%. The 95% rule can fail under certain situations, such as when there is cyclic behavior. See Appendix F.2 for a more detailed explanation on why the 95% rule is used.

The results of the Sobol peak and time at peak analyses can be seen in Figure F.1b and Figure F.1c. Although some bars between the final, peak, and time at peak values are the same, others differ. However, overall, similar values can be seen across the final, peak, and time at peak analyses. The peak infected values are more certain compared to the final infected values, which could be due to the 95% rule removing some of the noise of the simulation. The time at peak values have less error compared to the final and peak values. This is due to the restricted range of values. The time at peak value can only fall somewhere between 0 and 15, which corresponds to the start and end values of the simulation, respectively. The final and peak values can fall anywhere between 0 and any value, depending on the IC and how high the population can rise, as well as how fast the population can fall, *if* the population count falls.

F.2 Why 95%?

The 95% rule helps in the IVA analysis. Due to the solver, when taking the absolute peak value, the same time value can occur multiple times. Alternatively, in an ever-increasing value, such as phages, the peak values occur at the last time step of the simulation, or they plateau and no longer grow. However, as the parameter value changes, each graph for every input change will change the growth rate of the entity, changing how fast the entity population grows.

Figure F.2 shows how using the 95% rule vs the 100% rule for finding the max value reached helps smooth out computational errors from the ODE solver and smooths out



(a) Final population value. **(b)** Peak population value. **(c)** Time at peak population.

Figure F.1: Sobol analyses for the final, peak, and time of peak value with a washin and washout rate. The data was saved from the dashboard and plotted using Matplotlib. The values used for this Sobol test can be found in Table E.3.

the shape. For the phages, using the 100% rule (Figure F.2b) shows that the population peaked at the end of the simulation, $t = 15$, for all e values. However, at $t = 15$, the population plateaued, as evident by the line graph. Plotting the same plot but calculating the peak at 95% of the actual peak (Figure F.2a) shows that the green line ($e = 0.25$) "reached" its peak at $t = 8.4$ before the red line ($e = 0.05$) at $t = 9.4$, a whole unit of time after $e = 0.25$. The user can thus conclude that, for this instance, larger e values will cause the phage population to reach its "peak" faster than smaller e values.

Figure F.2c and Figure F.2d likewise show how the 95% rule can improve analysis of the change in time of peak. Figure F.2d shows how the peak is reached at set time values. Due to how *solve_ivp()* from SciPy works, it automatically chooses time values that it thinks would best capture the dynamics of the system without calculating too many steps. The user can control the step size by adjusting the absolute and relative error bounds, as well as by minimizing the time step size. The user can also specify their time range, along with the number of steps to run, thereby increasing control over the chosen time values. It takes approximately 0.02321 seconds to simulate 15-time units, where 200-time steps are selected and solved by the solver. Comparatively, a simulation with 1000 time units and 1000000 (a 5000x increase in samples) equidistant time samples takes about 1.71651 seconds to run, a 73.95562258x increase in time spent computing the simulation. The total time taken to run the entire method call, including a call to the simple graph maker at the top of the dashboard, was 1.76130 seconds, compared to 17.70634 seconds.

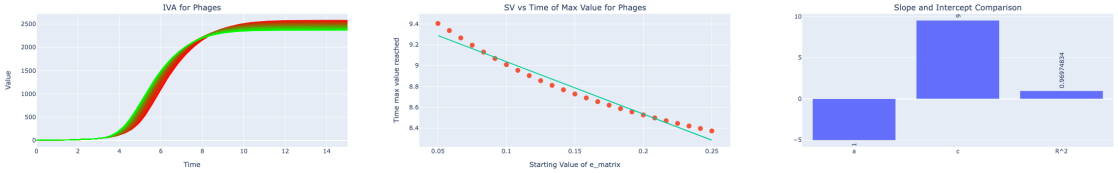
Alternatively, instead of controlling the solver, the user can use the 95% rule. Although some accuracy is lost. Going from the 100% rule to the 95% rule, the solver still captures the peak values and the dynamics, but the accuracy is lost. The 100% rule shows that

for $e = 0.25$, the time the uninfected population reached its peak occurred at $t = 3.2$. However, for the 95% rule, the time at which the peak occurred at $t = 3.05$. The slope (the a value) and the intercept (the c value) are somewhat similar, with very high and similar R^2 values (0.97), suggesting a good linear fit of the data.

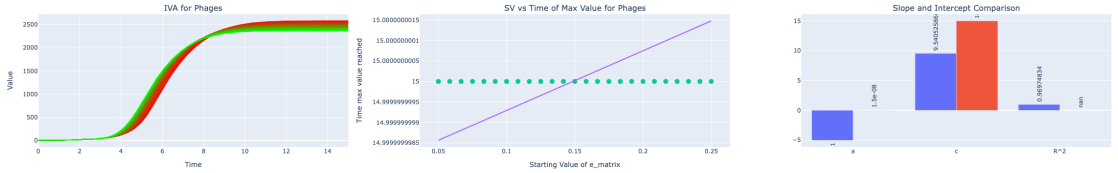
Figure F.2e illustrates how increasing the time sampling to a more fine-grained level yields a more accurate graph. Instead of having the solver choose the time values to test, 1000 equidistant time values were selected between 0 and 15. The solver can more accurately calculate the population values and calculate the proper time of peak. Comparing the 100% rule without the custom time values with the 100% rule with the custom time values shows that the same time values were calculated. In both, $e = 0.25$ resulted in a time of peak at 3.2, and for $e = 0.05$, the time of peak occurred at $t = 3.95$. This is in stark contrast to the 95% rule vs 100% rule without the custom time, showing a difference of 0.15 time units. The custom time values also preserved the shape of the curve e-value vs. time curve, being almost identical to that of the 95% rule as seen in Figure F.2c and Figure F.2e.

Another issue that arises with the custom time is that it does not solve the issue seen with the phages, where the time of peak is at $t = 15$.

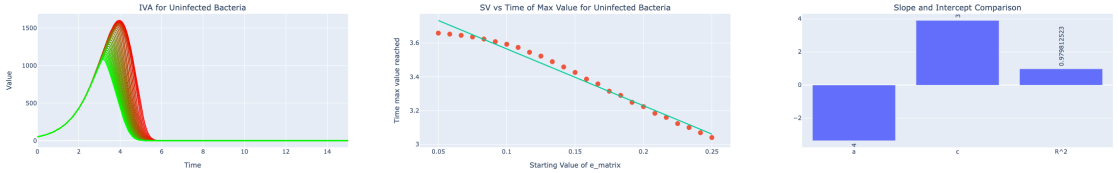
The user can control the % rule with a value input on the dashboard. They can select to use the 95% rule, 100% rule, or even 83% rule if they want by changing the value they use. The user can use their own custom time values to ensure that they get high-quality curves.



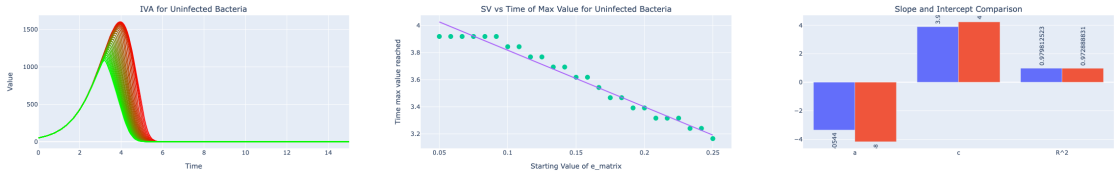
(a) IVA for phage population, 95% rule



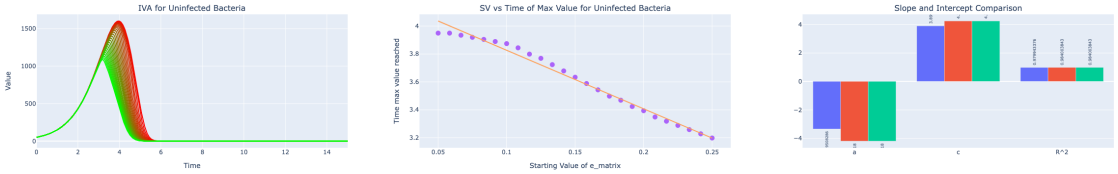
(b) IVA for phage population, 100% rule.



(c) IVA for uninfected population, 95% rule.



(d) IVA for uninfected population, 100% rule.



(e) IVA for uninfected population, 100% rule, 1000 equidistant time steps.

Figure F.2: Testing the 95% rule vs the 100% rule, where the time at the absolute peak is taken and plotted in the second plot. A comparison of phages and uninfected bacteria is shown. Verification of the graph shape between the 95% rule graph and a frequent time step with 100% rule can be seen in c) and e). The e value is changed, ranging from 0.05 to 0.25.

F.3 Graph Behavior with IVA

F.3.1 Resources R

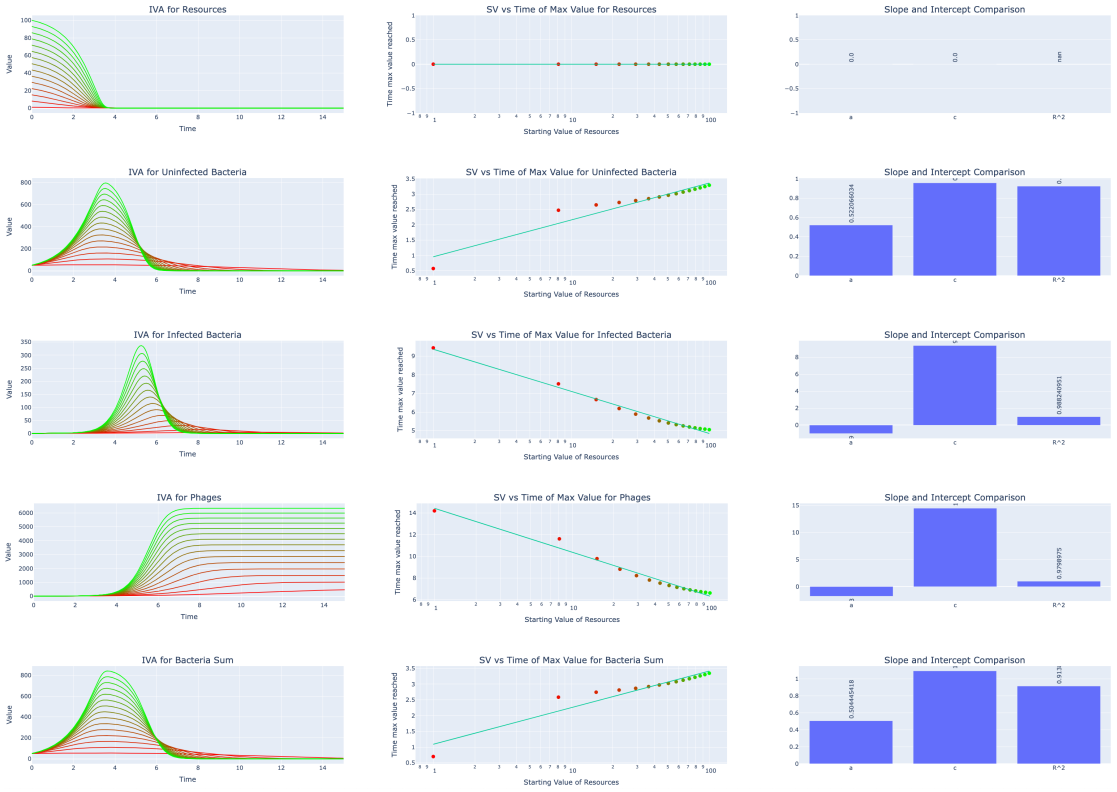


Figure F.3: Changing initial resource values.

F.3.2 Uninfected Bacteria U

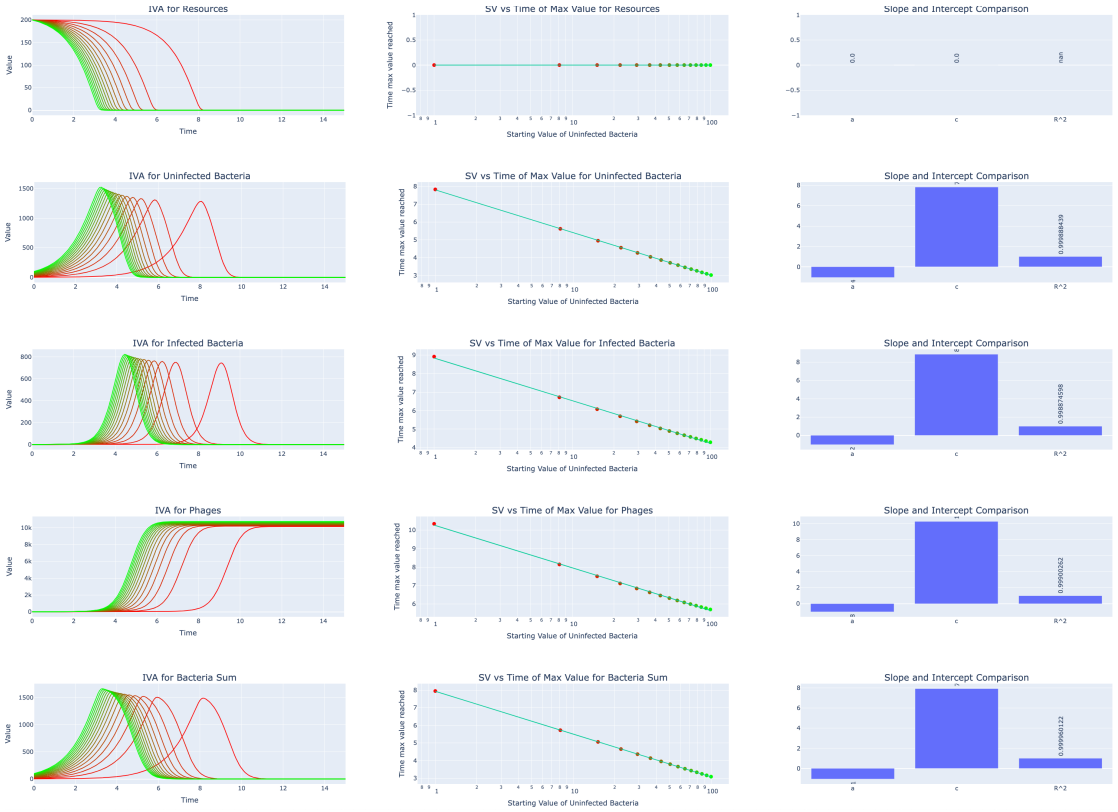


Figure F.4: Changing initial uninfected bacteria values.

F.3.3 Phages P

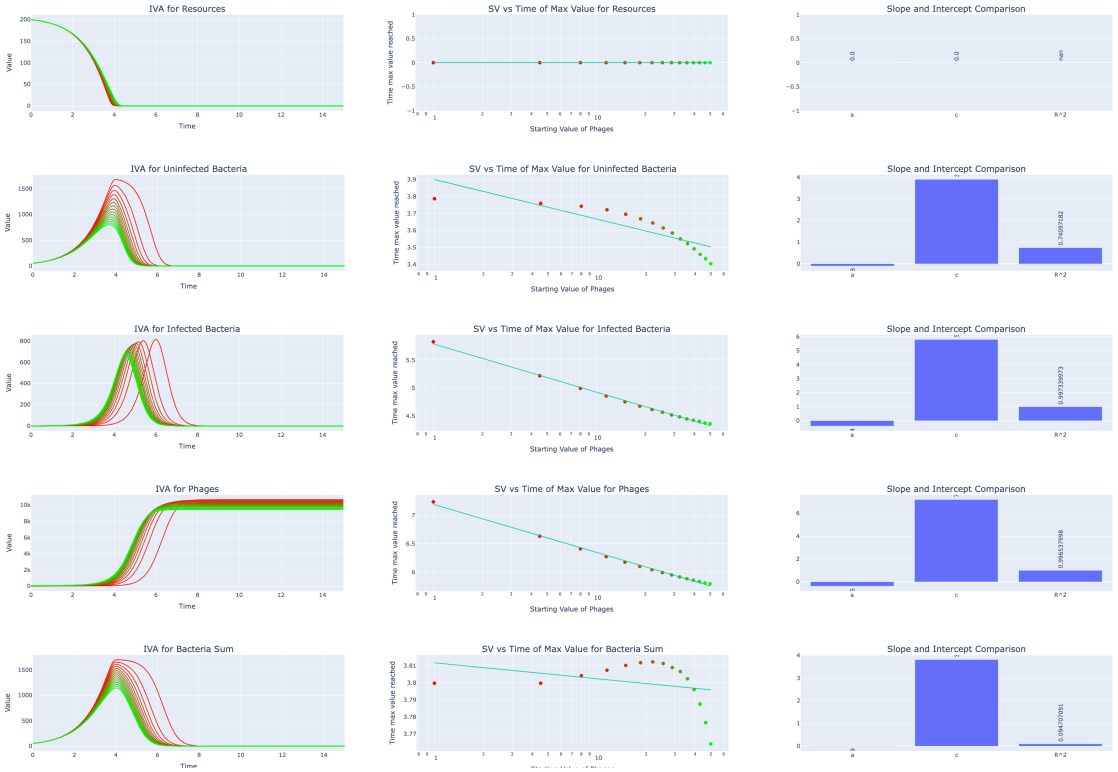


Figure F.5: Changing initial phage values.

F.3.4 Latent Period τ

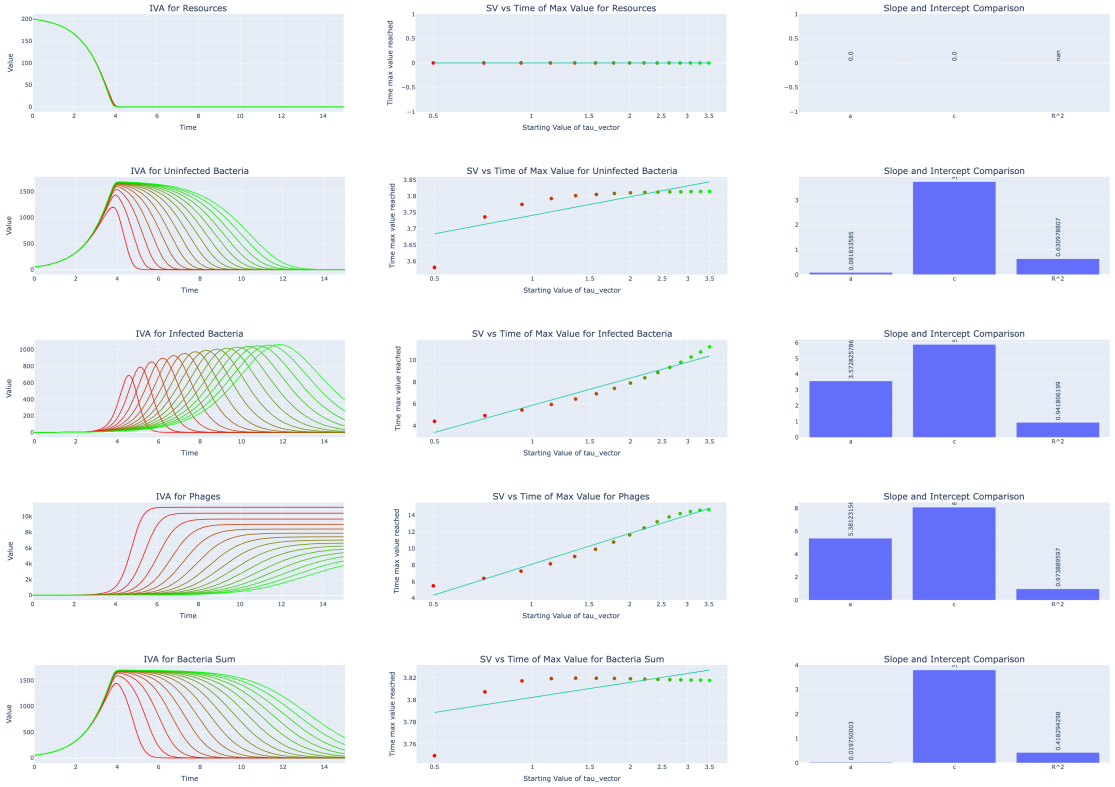


Figure F.6: Changing τ values.

F.3.5 Phage Adsorption Rate r

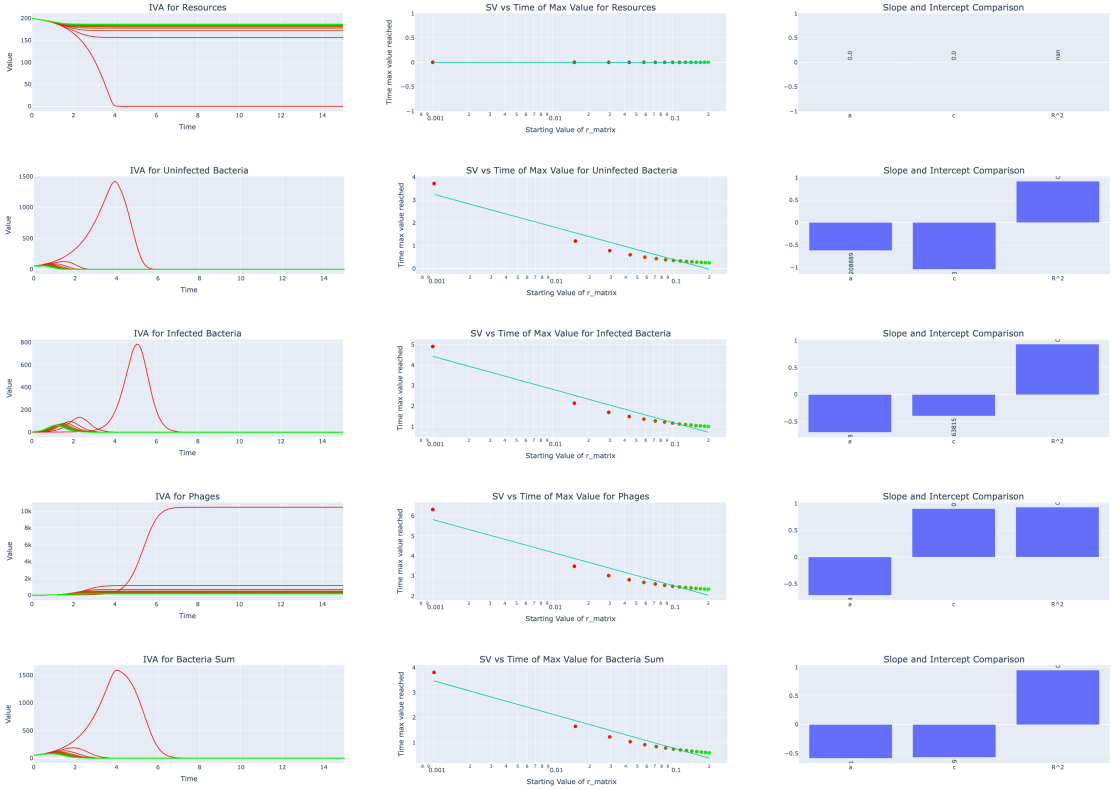


Figure F.7: Changing initial r values.

F.3.6 Burst Size β

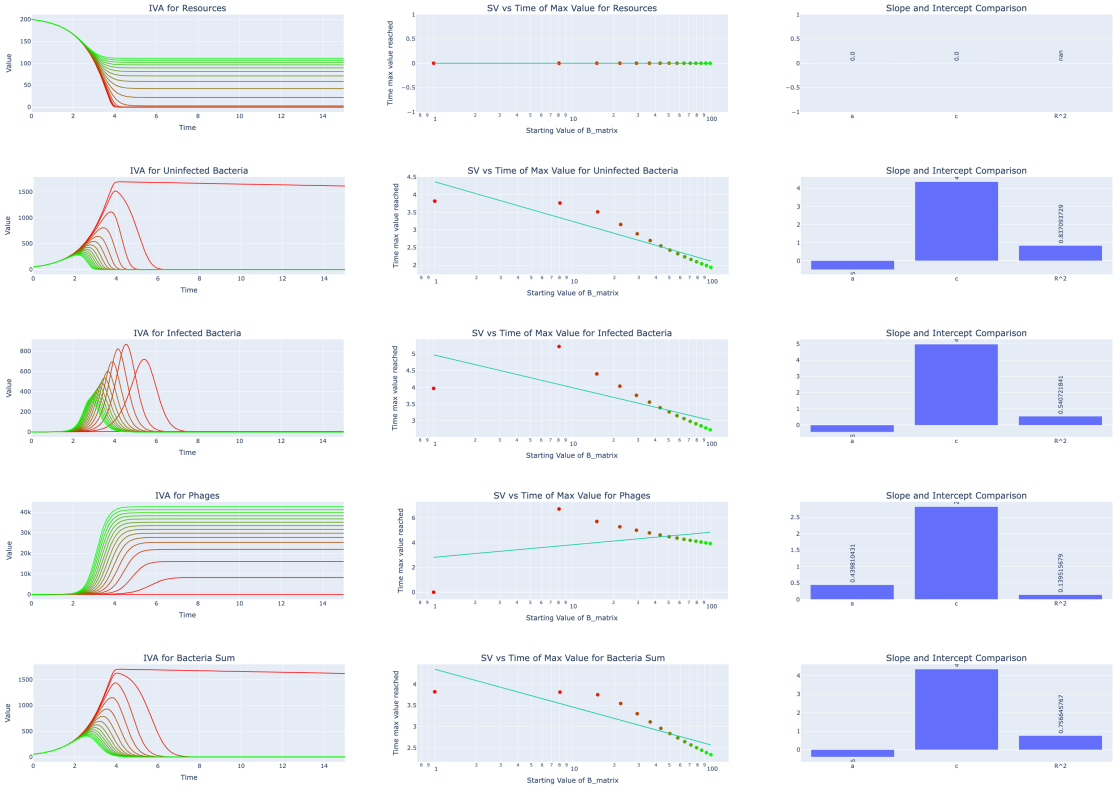


Figure F.8: Changing β values.

Bibliography

[1] Yuncong Geng, Thu Vu Phuc Nguyen, Ehsan Homaee, and Ido Golding. Using bacterial population dynamics to count phages and their lysogens. *Nature Communications*, 15(1):7814, September 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-51913-6. URL https://www.nature.com/articles/s41467-024-51913-6.

[2] Allan Campbell. The future of bacteriophage biology. *Nature Reviews Genetics*, 4 (6):471–477, June 2003. ISSN 1471-0056, 1471-0064. doi: 10.1038/nrg1089. URL https://www.nature.com/articles/nrg1089.

[3] Anders S. Nilsson. Cocktail, a Computer Program for Modelling Bacteriophage Infection Kinetics. *Viruses*, 14(11):2483, November 2022. ISSN 1999-4915. doi: 10.3390/v14112483. URL https://www.mdpi.com/1999-4915/14/11/2483.

[4] Konrad Krysiak-Baltyn, Gregory J. O. Martin, Anthony D. Stickland, Peter J. Scales, and Sally L. Gras. Simulation of phage dynamics in multi-reactor models of complex wastewater treatment systems. *Biochemical Engineering Journal*, 122: 91–102, June 2017. ISSN 1369-703X. doi: 10.1016/j.bej.2016.10.011. URL https://www.sciencedirect.com/science/article/pii/S1369703X16302728.

[5] Gabor Beke, Matej Stano, and Lubos Klucar. Modelling the interaction between bacteriophages and their bacterial hosts. *Mathematical Biosciences*, 279:27–32, September 2016. ISSN 0025-5564. doi: 10.1016/j.mbs.2016.06.009. URL https://www.sciencedirect.com/science/article/pii/S0025556416300700.

[6] Yuval Mulla, Janina Müller, Denny Trimcev, and Tobias Bollenbach. Extreme diversity of phage amplification rates and phage-antibiotic interactions revealed by PHORCE, December 2024. URL https://www.biorxiv.org/content/10.1101/2024.06.07.597930v3.

[7] Nitzan Soffer, Tamar Abuladze, Joelle Woolston, Manrong Li, Leigh Farris Hanna, Serena Heyse, Duane Charbonneau, and Alexander Sulakvelidze. Bacteriophages safely reduce Salmonella contamination in pet food and raw pet food ingredients.

Bacteriophage, 6(3):e1220347, July 2016. ISSN null. doi: 10.1080/21597081.2016.1220347. URL https://doi.org/10.1080/21597081.2016.1220347.

[8] Xuan Zhang, Yan Dong Niu, Yuchen Nan, Kim Stanford, Rick Holley, Tim McAllister, and Claudia Narváez-Bravo. SalmoFresh™ effectiveness in controlling Salmonella on romaine lettuce, mung bean sprouts and seeds. *International Journal of Food Microbiology*, 305:108250, September 2019. ISSN 0168-1605. doi: 10.1016/j.ijfoodmicro.2019.108250. URL https://www.sciencedirect.com/science/article/pii/S0168160519301709.

[9] Weizhen Zhang, Jing Liu, Yunxing Xiao, Yumiao Zhang, Yangjinzhi Yu, Zheng Zheng, Yafeng Liu, and Qi Li. The Impact of Cyanobacteria Blooms on the Aquatic Environment and Human Health. *Toxins*, 14(10):658, September 2022. ISSN 2072-6651. doi: 10.3390/toxins14100658. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9611879/.

[10] Basem Al-Shayeb, Rohan Sachdeva, Lin-Xing Chen, Fred Ward, Patrick Munk, Audra Devoto, Cindy J. Castelle, Matthew R. Olm, Keith Bouma-Gregson, Yuki Amano, Christine He, Raphaël Méheust, Brandon Brooks, Alex Thomas, Adi Lavy, Paula Matheus-Carnevali, Christine Sun, Daniela S. A. Goltsman, Mikayla A. Borton, Allison Sharrar, Alexander L. Jaffe, Tara C. Nelson, Rose Kantor, Ray Keren, Katherine R. Lane, Ibrahim F. Farag, Shufei Lei, Kari Finstad, Ronald Amundson, Karthik Anantharaman, Jinglie Zhou, Alexander J. Probst, Mary E. Power, Susannah G. Tringe, Wen-Jun Li, Kelly Wrighton, Sue Harrison, Michael Morowitz, David A. Relman, Jennifer A. Doudna, Anne-Catherine Lehours, Lesley Warren, Jamie H. D. Cate, Joanne M. Santini, and Jillian F. Banfield. Clades of huge phages from across Earth's ecosystems. *Nature*, 578(7795):425–431, February 2020. ISSN 1476-4687. doi: 10.1038/s41586-020-2007-4. URL https://www.nature.com/articles/s41586-020-2007-4.

[11] Teagan L Brown, Oliver J Charity, and Evelien M Adriaenssens. Ecological and functional roles of bacteriophages in contrasting environments: Marine, terrestrial and human gut. *Current Opinion in Microbiology*, 70:102229, December 2022. ISSN 1369-5274. doi: 10.1016/j.mib.2022.102229. URL https://www.sciencedirect.com/science/article/pii/S1369527422001138.

[12] Sandra Chibani-Chennoufi, Anne Bruttin, Marie-Lise Dillmann, and Harald Brüssow. Phage-Host Interaction: An Ecological Perspective. *Journal of Bacteriology*, 186(12):3677–3686, June 2004. ISSN 0021-9193. doi: 10.1128/JB.186.12.3677-3686.2004. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC419959/.

[13] Sebastián Coloma, Ursula Gaedke, Kaarina Sivonen, and Teppo Hiltunen. Frequency of virus-resistant hosts determines experimental community dynamics. *Ecology*, 100(1):e02554, 2019. ISSN 1939-9170. doi: 10.1002/ecy.2554. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/ecy.2554.

[14] Stephen Tucker and Peter Pollard. Identification of Cyanophage Ma-LBP and Infection of the Cyanobacterium Microcystis aeruginosa from an Australian Subtropical Lake by the Virus. *Applied and Environmental Microbiology*, 71(2):629–635, February 2005. doi: 10.1128/AEM.71.2.629-635.2005. URL https://journals.asm.org/doi/10.1128/aem.71.2.629-635.2005.

[15] Stephen Odonkor and Kennedy Addo. Bacteria Resistance to Antibiotics: Recent Trends and Challenges. *International Journal of Biological & Medical Research*, pages 1204–1210, January 2011.

[16] Angelina Volkova, Kelly Ruggles, Anjelique Schulfer, Zhan Gao, Stephen D. Ginsberg, and Martin J. Blaser. Effects of early-life penicillin exposure on the gut microbiome and frontal cortex and amygdala gene expression. *iScience*, 24(7): 102797, July 2021. ISSN 2589-0042. doi: 10.1016/j.isci.2021.102797. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8324854/.

[17] Nana Nguefang Laure and Juhee Ahn. Phage resistance-mediated trade-offs with antibiotic resistance in Salmonella Typhimurium. *Microbial Pathogenesis*, 171: 105732, October 2022. ISSN 08824010. doi: 10.1016/j.micpath.2022.105732. URL https://linkinghub.elsevier.com/retrieve/pii/S088240102200345X.

[18] Yuanyang Zhao, Mei Shu, Ling Zhang, Chan Zhong, Ningbo Liao, and Guoping Wu. Phage-driven coevolution reveals trade-off between antibiotic and phage resistance in Salmonella anatum. *ISME Communications*, 4(1):ycae039, January 2024. ISSN 2730-6151. doi: 10.1093/ismeco/ycae039. URL https://doi.org/10.1093/ismeco/ycae039.

[19] Beata Kowalska. Fresh vegetables and fruit as a source of Salmonella bacteria. *Annals of agricultural and environmental medicine: AAEM*, 30(1):9–14, March 2023. ISSN 1898-2263. doi: 10.26444/aaem/156765.

[20] Flat illustration of bacteriophage structures and anatomy. | Premium Vector. URL https://www.freepik.com/premium-vector/flat-illustration-bacteriophage-structures-anatomy_18751587.htm.

[21] Nicola Twilley. Inside the World of Viral Dark Matter. *The New Yorker*, February 2015. ISSN 0028-792X. URL https://www.newyorker.com/tech/annals-of-technology/phage-killer-viral-dark-matter.

[22] Viruses may play an unexplored role in the gut-brain axis. URL https://www.science.org/content/article/viruses-may-play-unexplored-role-gut-brain-axis.

[23] A. A. Lindberg. Bacteriophage receptors. *Annual Review of Microbiology*, 27:205–241, 1973. ISSN 0066-4227. doi: 10.1146/annurev.mi.27.100173.001225.

[24] Edel Stone, Katrina Campbell, Irene Grant, and Olivia McAuliffe. Understanding and Exploiting Phage–Host Interactions. *Viruses*, 11(6):567, June 2019. ISSN 1999-4915. doi: 10.3390/v11060567. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6630733/.

[25] Dalton V. Banh, Cameron G. Roberts, Adrian Morales-Amador, Brandon A. Berryhill, Waqas Chaudhry, Bruce R. Levin, Sean F. Brady, and Luciano A. Marraffini. Bacterial cGAS senses a viral RNA to initiate immunity. *Nature*, 623(7989):1001–1008, November 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06743-9. URL https://www.nature.com/articles/s41586-023-06743-9.

[26] Asaf Levy, Moran G. Goren, Ido Yosef, Oren Auster, Miriam Manor, Gil Amitai, Rotem Edgar, Udi Qimron, and Rotem Sorek. CRISPR adaptation biases explain preference for acquisition of foreign DNA. *Nature*, 520(7548):505–510, April 2015. ISSN 1476-4687. doi: 10.1038/nature14302. URL https://www.nature.com/articles/nature14302.

[27] Joanna Warwick-Dugdale, Holger H. Buchholz, Michael J. Allen, and Ben Temperton. Host-hijacking and planktonic piracy: How phages command the microbial high seas. *Virology Journal*, 16(1):1–13, December 2019. ISSN 1743-422X. doi: 10.1186/s12985-019-1120-1. URL https://virologyj.biomedcentral.com/articles/10.1186/s12985-019-1120-1.

[28] P. M. Sharp. Molecular evolution of bacteriophages: Evidence of selection against the recognition sites of host restriction enzymes. *Molecular Biology and Evolution*, 3(1):75–83, January 1986. ISSN 0737-4038. doi: 10.1093/oxfordjournals.molbev.a040377.

[29] Matthew K Waldor and David I Friedman. Phage regulatory circuits and virulence gene expression. *Current Opinion in Microbiology*, 8(4):459–465, August 2005. ISSN 1369-5274. doi: 10.1016/j.mib.2005.06.001. URL https://www.sciencedirect.com/science/article/pii/S1369527405000755.

[30] Louis-Charles Fortier and Ognjen Sekulovic. Importance of prophages to evolution and virulence of bacterial pathogens. *Virulence*, 4(5):354–365, July 2013. ISSN 2150-5594. doi: 10.4161/viru.24498. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3714127/.

[31] Anastasia A. Aksyuk and Michael G. Rossmann. Bacteriophage Assembly. *Viruses*, 3(3):172–203, February 2011. ISSN 1999-4915. doi: 10.3390/v3030172. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3185693/.

[32] I. N. Wang, D. L. Smith, and R. Young. Holins: The protein clocks of bacteriophage infections. *Annual Review of Microbiology*, 54:799–825, 2000. ISSN 0066-4227. doi: 10.1146/annurev.micro.54.1.799.

[33] Richard E Lenski. TWO-STEP RESISTANCE BY ESCHERICHIA COLI B TO BACTERIOPHAGE T2. *Genetics*, 107(1):1–7, May 1984. ISSN 1943-2631. doi: 10.1093/genetics/107.1.1. URL https://doi.org/10.1093/genetics/107.1.1.

[34] Laith Harb, Karthik Chamakura, Pratick Khara, Peter J. Christie, Ry Young, and Lanying Zeng. ssRNA phage penetration triggers detachment of the F-pilus. *Proceedings of the National Academy of Sciences of the United States of America*, 117 (41):25751–25758, October 2020. ISSN 0027-8424. doi: 10.1073/pnas.2011901117. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7568308/.

[35] Sanju Tamang. Horizontal Gene Transfer in Prokaryotes and Eukaryotes, September 2023. URL https://microbenotes.com/horizontal-gene-transfer-prokaryotes-eukaryotes/.

[36] Laura M. Kasman and La Donna Porter. Bacteriophages. In *StatPearls*. StatPearls Publishing, Treasure Island (FL), 2025. URL http://www.ncbi.nlm.nih.gov/books/NBK493185/.

[37] Demeng Tan, Sine Lo Svenningsen, and Mathias Middelboe. Quorum Sensing Determines the Choice of Antiphage Defense Strategy in Vibrio anguillarum. *mBio*, 6(3):10.1128/mbio.00627–15, June 2015. doi: 10.1128/mbio.00627-15. URL https://journals.asm.org/doi/10.1128/mbio.00627-15.

[38] Avinoam Rabinovitch, Ira Aviram, and Arieh Zaritsky. Bacterial debris—an ecological mechanism for coexistence of bacteria and their viruses. *Journal of Theoretical Biology*, 224(3):377–383, October 2003. ISSN 0022-5193. doi: 10.1016/S0022-5193(03)00174-7. URL https://www.sciencedirect.com/science/article/pii/S0022519303001747.

[39] James J. Bull, Kelly A. Christensen, Carly Scott, Benjamin R. Jack, Cameron J. Crandall, and Stephen M. Krone. Phage-Bacterial Dynamics with Spatial Structure: Self Organization around Phage Sinks Can Promote Increased Cell Densities. *Antibiotics*, 7(1):8, March 2018. ISSN 2079-6382. doi: 10.3390/antibiotics7010008. URL https://www.mdpi.com/2079-6382/7/1/8.

[40] Raunak Dey, Ashley R. Coenen, Natalie E. Solonenko, Marie N. Burris, Anna I. Mackey, Julia Galasso, Christine L. Sun, David Demory, Daniel Muratore, Stephen J. Beckett, Matthew B. Sullivan, and Joshua S. Weitz. Emergent higher-order interactions enable coexistence in phage-bacteria community dynamics, May 2025. URL https://www.biorxiv.org/content/10.1101/2025.05.15.651590v1.

[41] Anika Gupta, Norma Morella, Dmitry Sutormin, Naisi Li, Karl Gaisser, Alexander Robertson, Yaroslav Ispolatov, Georg Seelig, Neelendu Dey, and Anna Kuchina. Combinatorial phenotypic landscape enables bacterial resistance to phage infection, January 2025. URL https://www.biorxiv.org/content/10.1101/2025.01.13.632860v1.

[42] Stephen T. Abedon. Phage "delay" towards enhancing bacterial escape from biofilms: A more comprehensive way of viewing resistance to bacteriophages. *AIMS Microbiology*, 3(microbiol-03-00186):186–226, 2017. ISSN 2471-1888. doi: 10.3934/microbiol.2017.2.186. URL http://www.aimspress.com/article/doi/10.3934/microbiol.2017.2.186.

[43] Rasmus Skytte Eriksen, Sine L. Svenningsen, Kim Sneppen, and Namiko Mitarai. A growing microcolony can survive and support persistent propagation of virulent phages. *Proceedings of the National Academy of Sciences*, 115(2):337–342, January 2018. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1708954115. URL https://pnas.org/doi/full/10.1073/pnas.1708954115.

[44] Christoph Lohrmann, Christian Holm, and Sujit S. Datta. Influence of bacterial swimming and hydrodynamics on attachment of phages. *Soft Matter*, 20(24):4795–4805, June 2024. ISSN 1744-6848. doi: 10.1039/D4SM00060A. URL https://pubs.rsc.org/en/content/articlelanding/2024/sm/d4sm00060a.

[45] Pramalkumar H. Patel, Véronique L. Taylor, Chi Zhang, Landon J. Getz, Alexa D. Fitzpatrick, Alan R. Davidson, and Karen L. Maxwell. Anti-phage defence through inhibition of virion assembly. *Nature Communications*, 15(1):1644, February 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-45892-x. URL https://www.nature.com/articles/s41467-024-45892-x.

[46] Michael J. Bucher and Daniel M. Czyż. Phage against the Machine: The SIE-ence of Superinfection Exclusion. *Viruses*, 16(9):1348, August 2024. ISSN 1999-4915. doi: 10.3390/v16091348.

[47] Shuji Kanamaru, Kazuya Uchida, Mai Nemoto, Alec Fraser, Fumio Arisaka, and Petr G. Leiman. Structure and Function of the T4 Spackle Protein Gp61.3. *Viruses*, 12(10):1070, September 2020. ISSN 1999-4915. doi: 10.3390/v12101070. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7650644/.

[48] Justin C. Leavitt, Brianna M. Woodbury, Eddie B. Gilcrease, Charles M. Bridges, Carolyn M. Teschke, and Sherwood R. Casjens. Bacteriophage P22 SieA-mediated superinfection exclusion. *mBio*, 15(2):e02169–23, January 2024. doi: 10.1128/mbio.02169-23. URL https://journals.asm.org/doi/10.1128/mbio.02169-23.

[49] Jacob Beal, Natalie G. Farny, Traci Haddock-Angelli, Vinoo Selvarajah, Geoff S. Baldwin, Russell Buckley-Taylor, Markus Gershater, Daisuke Kiga, John Marken, Vishal Sanchania, Abigail Sison, and Christopher T. Workman. Robust estimation of bacterial cell count from optical density. *Communications Biology*, 3(1):512, September 2020. ISSN 2399-3642. doi: 10.1038/s42003-020-01127-5. URL https://www.nature.com/articles/s42003-020-01127-5.

[50] Portia Mira, Pamela Yeh, and Barry G. Hall. Estimating microbial population data from optical density. *PLOS ONE*, 17(10):e0276040, October 2022. ISSN 1932-6203. doi: 10.1371/journal.pone.0276040. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0276040.

[51] Advanced Wastewater Modelling | GPS-X - Hydromantis. URL https://www.hydromantis.com/GPSX-innovative.html.

[52] Jacqueline Heard, Emma Harvey, Bruce B. Johnson, John D. Wells, and Michael J. Angove. The effect of filamentous bacteria on foam production and stability. *Colloids and Surfaces. B, Biointerfaces*, 63(1):21–26, May 2008. ISSN 0927-7765. doi: 10.1016/j.colsurfb.2007.10.011.

[53] S. J. Schrag and J. E. Mittler. Host-Parasite Coexistence: The Role of Spatial Refuges in Stabilizing Bacteria-Phage Interactions. *The American Naturalist*, 148 (2):348–377, August 1996. ISSN 0003-0147. doi: 10.1086/285929. URL https://www.journals.uchicago.edu/doi/abs/10.1086/285929.

[54] Brendan J. M. Bohannan and Richard E. Lenski. Effect of Resource Enrichment on a Chemostat Community of Bacteria and Bacteriophage. *Ecology*, 78(8):2303–2315, 1997. ISSN 1939-9170. doi: 10.1890/0012-9658(1997)078[2303:EOREOA]2.0.CO;2. URL https://onlinelibrary.wiley.com/doi/abs/10.1890/0012-9658%281997%29078%5B2303%3AEOREOA%5D2.0.CO%3B2.

[55] Richard E. Lenski. Dynamics of Interactions between Bacteria and Virulent Bacteriophage. In K. C. Marshall, editor, *Advances in Microbial Ecology*, pages 1–44. Springer US, Boston, MA, 1988. ISBN 978-1-4684-5409-3. doi: 10.1007/978-1-4684-5409-3_1. URL https://doi.org/10.1007/978-1-4684-5409-3_1.

[56] Allan Campbell. Conditions for the Existence of Bacteriophage. *Evolution*, 15(2): 153–165, 1961. ISSN 0014-3820. doi: 10.2307/2406076. URL https://www.jstor.org/stable/2406076.

[57] J. S. Weitz, H. Hartman, and S. A. Levin. Coevolutionary arms races between bacteria and bacteriophage. *Proceedings of the National Academy of Sciences*, 102 (27):9535–9540, July 2005. doi: 10.1073/pnas.0504062102. URL https://www.pnas.org/doi/abs/10.1073/pnas.0504062102.

[58] Matthew Scott, Carl W. Gunderson, Eduard M. Mateescu, Zhongge Zhang, and Terence Hwa. Interdependence of Cell Growth and Gene Expression: Origins and Consequences. *Science*, 330(6007):1099–1102, November 2010. doi: 10.1126/science.1192588. URL https://www.science.org/doi/10.1126/science.1192588.

[59] Luis S. Mayorga, Ignacio Cebrian, Meghna Verma, Stefan Hoops, and Josep Bassaganya-Riera. Reconstruction of endosomal organization and function by a combination of ODE and agent-based modeling strategies. *Biology Direct*, 13(1): 1–21, December 2018. ISSN 1745-6150. doi: 10.1186/s13062-018-0227-4. URL https://biologydirect.biomedcentral.com/articles/10.1186/s13062-018-0227-4.

[60] S. M. Krone. Spatial models: Stochastic and deterministic. *Mathematical and Computer Modelling*, 40(3):393–409, August 2004. ISSN 0895-7177. doi: 10.1016/j.mcm.2003.09.037. URL https://www.sciencedirect.com/science/article/pii/S089571770480270X.

[61] Michael Klann and Heinz Koeppl. Spatial Simulations in Systems Biology: From Molecules to Cells. *International Journal of Molecular Sciences*, 13(6):7798–7827, June 2012. ISSN 1422-0067. doi: 10.3390/ijms13067798. URL https://www.mdpi.com/1422-0067/13/6/7798.

[62] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, March 2020. ISSN 1548-7105. doi: 10.1038/s41592-019-0686-2. URL https://www.nature.com/articles/s41592-019-0686-2.

[63] J. R. Dormand and P. J. Prince. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, March 1980. ISSN

0377-0427. doi: 10.1016/0771-050X(80)90013-3. URL https://www.sciencedirect.com/science/article/pii/0771050X80900133.

[64] Dash Documentation & User Guide | Plotly. URL https://dash.plotly.com/.

[65] I. M Sobol′. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55(1):271–280, February 2001. ISSN 0378-4754. doi: 10.1016/S0378-4754(00)00270-6. URL https://www.sciencedirect.com/science/article/pii/S0378475400002706.

[66] Joblib Development Team. Joblib: running python functions as pipeline jobs, 2020. URL https://joblib.readthedocs.io/.

[67] Python Software Foundation. Python programming language, version 3.11.6, 2024. URL https://www.python.org. Accessed: 2025-05-19.

[68] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[69] Takuya Iwanaga, William Usher, and Jonathan Herman. Toward SALib 2.0: Advancing the accessibility and interpretability of global sensitivity analyses. *Socio-Environmental Systems Modelling*, 4:18155–18155, May 2022. ISSN 2663-3027. doi: 10.18174/sesmo.18155. URL https://sesmo.org/article/view/18155.

[70] Jon Herman and Will Usher. SALib: An open-source Python library for Sensitivity Analysis. *Journal of Open Source Software*, 2(9):97, January 2017. ISSN 2475-9066. doi: 10.21105/joss.00097. URL https://joss.theoj.org/papers/10.21105/joss.00097.

[71] Aric Hagberg, Pieter J. Swart, and Daniel A. Schult. Exploring network structure, dynamics, and function using NetworkX. Technical Report LA-UR-08-05495; LA-UR-08-5495, Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), January 2008. URL https://www.osti.gov/biblio/960616.

[72] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

[73] Bryan B. Hsu, Travis E. Gibson, Vladimir Yeliseyev, Qing Liu, Lorena Lyon, Lynn Bry, Pamela A. Silver, and Georg K. Gerber. Dynamic Modulation of the Gut

Microbiota and Metabolome by Bacteriophages in a Mouse Model. *Cell Host & Microbe*, 25(6):803–814.e5, June 2019. ISSN 1934-6069. doi: 10.1016/j.chom.2019.05.001.

[74] Garrett Hardin. The Competitive Exclusion Principle. *Science*, 131(3409):1292–1297, April 1960. doi: 10.1126/science.131.3409.1292. URL https://www.science.org/doi/10.1126/science.131.3409.1292.

[75] Shu Ju and Donald L. DeAngelis. The R* rule and energy flux in a plant-nutrient ecosystem. *Journal of Theoretical Biology*, 256(3):326–332, February 2009. ISSN 1095-8541. doi: 10.1016/j.jtbi.2008.10.002.

[76] Naomi Iris van den Berg, Daniel Machado, Sophia Santos, Isabel Rocha, Jeremy Chacón, William Harcombe, Sara Mitri, and Kiran R. Patil. Ecological modelling approaches for predicting emergent properties in microbial communities. *Nature Ecology & Evolution*, 6(7):855–865, July 2022. ISSN 2397-334X. doi: 10.1038/s41559-022-01746-7. URL https://www.nature.com/articles/s41559-022-01746-7.

[77] Val Smith. Effects of resource supplies on the structure and function of microbial communities. *Antonie van Leeuwenhoek*, 81:99–106, September 2002. doi: 10.1023/A:1020533727307.

[78] Lucía Fernández, Ana Rodríguez, and Pilar García. Phage or foe: An insight into the impact of viral predation on microbial communities. *The ISME journal*, 12(5):1171–1179, May 2018. ISSN 1751-7370. doi: 10.1038/s41396-018-0049-5.

[79] Tom Clegg and Thilo Gross. Cross-feeding creates tipping points in microbiome diversity. *Proceedings of the National Academy of Sciences*, 122(19):e2425603122, May 2025. doi: 10.1073/pnas.2425603122. URL https://www.pnas.org/doi/10.1073/pnas.2425603122.

[80] Lars Fieseler and Steven Hagens. Food Safety. In David R. Harper, Stephen T. Abedon, Benjamin H. Burrowes, and Malcolm L. McConville, editors, *Bacteriophages: Biology, Technology, Therapy*, pages 857–890. Springer International Publishing, Cham, 2021. ISBN 978-3-319-41986-2. doi: 10.1007/978-3-319-41986-2_29. URL https://doi.org/10.1007/978-3-319-41986-2_29.

[81] Heinz G. Floss and Tin-Wein Yu. RifamycinMode of Action, Resistance, and Biosynthesis. *Chemical Reviews*, 105(2):621–632, February 2005. ISSN 0009-2665. doi: 10.1021/cr030112j. URL https://doi.org/10.1021/cr030112j.

[82] A. Tomasz. The Mechanism of the Irreversible Antimicrobial Effects of Penicillins: How the Beta-Lactam Antibiotics Kill and Lyse Bacteria. *Annual Review of Microbiology*, 33(Volume 33, 1979):113–137, October 1979. ISSN 0066-4227, 1545-3251.

doi: 10.1146/annurev.mi.33.100179.000553. URL https://www.annualreviews.org/content/journals/10.1146/annurev.mi.33.100179.000553.

[83] Sergei B. Vakulenko and Shahriar Mobashery. Versatility of Aminoglycosides and Prospects for Their Future. *Clinical Microbiology Reviews*, 16(3):430–450, July 2003. doi: 10.1128/cmr.16.3.430-450.2003. URL https://journals.asm.org/doi/10.1128/cmr.16.3.430-450.2003.

[84] Sophie Leclercq, Firoz M. Mian, Andrew M. Stanisz, Laure B. Bindels, Emmanuel Cambier, Hila Ben-Amram, Omry Koren, Paul Forsythe, and John Bienenstock. Low-dose penicillin in early life induces long-term changes in murine gut microbiota, brain cytokines and behavior. *Nature Communications*, 8:15062, April 2017. ISSN 2041-1723. doi: 10.1038/ncomms15062. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5382287/.

[85] Global action plan on antimicrobial resistance. URL https://www.who.int/publications/i/item/9789241509763.

[86] Christopher R. Grasso, Kaytee L. Pokrzywinski, Christopher Waechter, Taylor Rycroft, Yanyan Zhang, Alyssa Aligata, Michael Kramer, and Anisha Lamsal. A Review of Cyanophage–Host Relationships: Highlighting Cyanophages as a Potential Cyanobacteria Control Strategy. *Toxins*, 14(6):385, June 2022. ISSN 2072-6651. doi: 10.3390/toxins14060385. URL https://www.mdpi.com/2072-6651/14/6/385.

[87] Katelyn M. McKindles, Makayla A. Manes, Jonathan R. DeMarco, Andrew McClure, R. Michael McKay, Timothy W. Davis, and George S. Bullerjahn. Dissolved Microcystin Release Coincident with Lysis of a Bloom Dominated by Microcystis spp. in Western Lake Erie Attributed to a Novel Cyanophage. *Applied and Environmental Microbiology*, 86(22):e01397–20, October 2020. doi: 10.1128/AEM.01397-20. URL https://journals.asm.org/doi/10.1128/aem.01397-20.

[88] (PDF) Economic Impacts of Red Tide Events on Restaurant Sales. URL https://www.researchgate.net/publication/23515658_Economic_Impacts_of_Red_Tide_Events_on_Restaurant_Sales.

[89] Yung Sung Cheng, Yue Zhou, Clinton M. Irvin, Richard H. Pierce, Jerome Naar, Lorraine C. Backer, Lora E. Fleming, Barbara Kirkpatrick, and Dan G. Baden. Characterization of Marine Aerosol for Assessment of Human Exposure to Brevetoxins. *Environmental Health Perspectives*, 113(5):638–643, May 2005. ISSN 0091-6765. doi: 10.1289/ehp.7496. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1257561/.

[90] Barbara Kirkpatrick, Judy A Bean, Lora E Fleming, Gary Kirkpatrick, Lynne Grief, Kate Nierenberg, Andrew Reich, Sharon Watkins, and Jerome Naar. Gastrointestinal Emergency Room Admissions and Florida Red Tide Blooms. *Harmful algae*, 9(1):82–86, January 2010. ISSN 1568-9883. doi: 10.1016/j.hal.2009.08.005. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2786186/.

[91] Porter Hoagland, Di Jin, Lara Y. Polansky, Barbara Kirkpatrick, Gary Kirkpatrick, Lora E. Fleming, Andrew Reich, Sharon M. Watkins, Steven G. Ullmann, and Lorraine C. Backer. The costs of respiratory illnesses arising from Florida gulf coast Karenia brevis blooms. *Environmental Health Perspectives*, 117(8):1239–1243, August 2009. ISSN 1552-9924. doi: 10.1289/ehp.0900645.