Evolution of overlapping reading frames in virus genomes

Laura Muñoz Baena,

Supervisor: Poon, Art FY, The University of Western Ontario
A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Microbiology and Immunology
© Laura Muñoz Baena 2023

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Bioinformatics Commons, Evolution Commons, and the Virology Commons

Recommended Citation
https://ir.lib.uwo.ca/etd/9531

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlsadmin@uwo.ca.
Abstract

Viruses are formidable pathogens that represent the majority of biological entities on our planet, and their genomes are a source of interesting enigmas. One feature in which virus genomes are usually rich, is the presence of overlapping reading frames (OvRFs) — portions of the genome where the same nucleotide sequence encodes more than one protein. OvRFs are hypothesized to be used by viruses to encode proteins more compactly and to regulate transcription. In addition, OvRFs might be a source of gene novelty, facilitating the creation of new open reading frames (ORF) within the transcriptional context of existing ones.

To characterize the distribution OvRFs in viruses, I analyzed 12,609 reference genomes from the NCBI virus database and discovered that, while the number of OvRFs increases the genome length, the overlapping regions tend to be shorter in longer genomes. I also demonstrated that different frameshifts have distinct patterns in OvRFs. For example, +2 frameshifts are predominantly found in dsDNA viruses, whereas +0 frameshifts in RNA viruses tend to involve longer overlaps, which may increase the selective burden of the same nucleotide positions within codons. Further, I retrieved $n = 8,586$ protein-coding sequences from $n = 1,224$ reference genomes, and used an alignment-free method to cluster these sequences within virus families. I used these clusters to develop a new network-based representation of the distribution of OvRFs, which provides a means of visualizing and analyzing these genome features for each virus family. I also used these networks to generate a high-level visualization of how overlapping genes are distributed among virus genomes in the same family.

Evolution in overlapping genes is complicated because the effect of a nucleotide substitution has multiple contexts. To unravel the effects of OvRFs on virus evolution, I developed HexSE, a simulation model of nucleotide sequence evolution along a phylogeny that tracks the substitution rates at every nucleotide site. In HexSE, I implemented a customized data structure to efficiently track the substitution rates at every nucleotide site. These rates are determined by the stationary nucleotide frequencies, transition bias, and the distribution of selection biases ($dN$ and $dS$) in the respective reading frames. Next, I compared HexSE simulations under varying settings to an align-
ment of actual hepatitis B virus (HBV) genomes, which revealed consistent drops in synonymous substitution rates (dS) in association with overlapping regions of an ORF.

This thesis explores the cryptic information contained in viral genomes to help explain the evolutionary processes that shape them. In particular, understanding the impact of OvRFs on the evolution of virus genomes will provide us with crucial pieces of a significant puzzle — understanding the origin of new genes in virus genomes, and thereby virus diversity.

**Keywords:** Virus evolution, molecular dynamics, overlapping reading frames, bioinformatics, genome networks.
Summary for lay audience

This research delves into the intriguing world of viruses, which are highly diverse pathogens with genomes that hold clues to the origin of life. Many viral genomes have a type of gene arrangement known as overlapping reading frames (OvRFs), where the same sequence encodes multiple proteins. OvRFs are thought to be used by viruses to increase the amount of information contained in smaller genomes, regulate transcription, and contribute to the creation of new genes.

By analyzing thousands of viral genomes, I found that OvRFs in longer genomes tend to be shorter in length, and that different types of viruses exhibit distinct patterns in OvRFs, with specific frameshift preferences. Additionally, I developed a unique network-based approach to visualize and analyze the OvRF distribution within virus families. Notably, the presence of OvRFs is correlated with network-based statistics for some virus families such as Coronaviridae, Rhabdoviridae, and Papillomaviridae. To explore the evolutionary impact of OvRFs, I also developed a simulation model called HexSE. By simulating nucleotide sequence evolution in hepatitis B virus, I discovered consistent drops in synonymous substitution rates within overlapping gene regions.

This project aims to decipher the cryptic information contained within viral genomes to shed light on the evolutionary processes shaping them. Understanding the role of OvRFs in virus genomes provides valuable insights into the origin of new genes and the diversity of viruses. Overall, this research contributes to our understanding of virus genomes and their significance in the larger context of life’s origin.
Co-Authorship Statement

I would like to acknowledge the invaluable work of the members of the PoonLab that contributed to my thesis.

- Chapter 1 was written by me and proofread by Art Poon.

- Chapter 2 was conceptualized by Art Poon. He and I equally contributed to designing the methodology, writing the software, and preparing the manuscript. I was in charge of the data curation and the formal analysis of the sequences.

- Chapter 3 was initially conceived by Art Poon. I was in charge of writing the pipeline with the invaluable help of Kaitlyn Wade. Art Poon and I prepared and edited the manuscript. Recent improvements to HexSE such as unit testing and implementing new features for rate variation have been made by Sandeep Singh Dhaliwa, Gopi Gugan and myself.

- Chapter 4 was conceived by Art Poon and myself. I was in charge of collecting the data and performing the formal analysis. I wrote the majority of the software with help from William Wang. I wrote the manuscript, that was edited by Art Poon.
Resumen

Esta investigación pretende evaluar algunas de las particularidades de los virus, una clase de patógenos altamente diversos con genomas que pueden contener información valiosa sobre el origen de la vida en la Tierra. Algunos genomas de virus se caracterizan por un tipo de organización conocida como marcos de lectura sobrelapantes, o OvRFs por sus siglas en inglés (overlapping reading frames), que consisten en secuencias de nucleótidos que codifican múltiples proteínas. Se cree que los marcos de lectura sobrelapantes son usados por los virus para aumentar la cantidad de información contenida en genomas pequeños, para regular procesos de transcripción, y que contribuyen a la creación de nuevos genes.

En esta tesis analizamos miles de genomas de virus y encontramos que los OvRFs presentes en los genomas más largos (usualmente virus de ADN) suelen ser cortos (alrededor de 4 nucleótidos en longitud), y que los patrones de solapamiento y corrimientos en el marco de lectura suelen ser específicos dependiendo del tipo de genoma codificado por el virus (ADN de cadena doble, AND de cadena simple, ARN de cadena doble, o ARN de cadena simple). Adicionalmente desarrollamos un método para graficar los genomas de virus dentro de la misma familia que permite visualizar la distribución de los OvRFs y predecir su formación de acuerdo a algunas a características de las gráficas en familias como Coronaviridae, Rhabdoviridae, y Papillomaviridae.

Para explorar el impacto evolutivo de los OvRFs, también desarrollamos un modelo de simulación llamado HexSE. HexSE fue usado para simular el proceso de evolución en el genoma del virus de la hepatitis B (VHB), donde descubrimos que el número de mutaciones sinónimas disminuía significativamente en regiones con marcos sobrelapantes de lectura.

Este proyecto es un intento por decifrar la información contenida en los genomas de los virus y nos permite encontrar algunas respuestas sobre los procesos evolutivos que los afectan. Entender el rol de los OvRFs en los genomas de virus genera un conocimiento valioso sobre el origen de nuevos genes y en general, de la diversidad de los virus en la tierra.
Acknowledgements

I believe that science is a collaborative effort of humanity to explain the world. Recognizing the limitations inherent to our physical and biological capacities, we have come to understand that it is through collective work that we can address some of the most intricate questions that confront us. This thesis is only a modest contribution within the vast ocean of knowledge that humanity has accumulated throughout history. Nevertheless, this research was possible because of the invaluable help of the people that fueled my passion and sustained my commitment to scientific exploration over the years.

First, I would like to thank Art Poon, who welcomed an international student into his lab, transcending language and cultural barriers. Since bioinformatics is the convergence of countless concepts, formulas, and models from different disciplines, the learning curve of this constantly evolving field is quite steep. Art’s patience and guidance allowed me to understand the theory behind my experiments while nurturing my scientific curiosity over the years. His support extended beyond the research, showing a genuine interest in both my academic success and personal well-being, and encouraging me to explore my interests beyond academia while ensuring I stayed on track to complete my degree. I’m truly thankful for Art’s mentorship and the positive impact he’s had on my scientific journey.

My gratitude extends to the members of the PoonLab — Connor, John, Abayomi, Kaitlyn, Emmanuel, Erin, Gopi, Sareh, Roux-Cil, Amjad, Paula, and William — who contributed not only to the technical aspects of my project, aiding with coding and biological queries, but also helped me refining my presentation and writing skills. Their camaraderie transformed my time at Western into a period of both academic growth and personal enrichment.

To my Canadian friends — Tammy, Alicia, and Faisal — your companionship became a vital anchor during my transition to a foreign land, making it feel like home across continents. Charan and Brittany, your blend of humor and wisdom not only brought joy to my time at Western but also emphasized the importance of a vibrant and enjoyable learning process.

To my friends back home, each of you has played an indispensable role in my life. Helena,
your steadfast guidance through this doctoral journey, coupled with your emotional sustenance and practical aid in presentations, posters, and emails, have been unwavering. I am profoundly fortunate to know that our sisterhood will endure, no matter where life takes us. Cami, your expertise in bioinformatics and thoughtful insights into humanity’s deeper quests kept me focus and steered my aspirations of the scientist that I want to be. Dani, you not only read my thesis and corrected my grammar but also suggested some of the most amazing books that have accompanied me throughout these years. Munoz and David, thank you for being an invaluable source of laughs, I know I can count on you to lift my spirit anytime. My high school friends — Wendy, your infectious passion for evolution constantly triggers my curiosity and fascination for the natural world; Eli, your ability to find beauty and your sensibility to detect hidden treasures fills my heart with joy; Juli, your love for science and your deep understanding of aquatic ecosystems never cease to amaze me; Sabi, you are the insightful geologist who enhance my understanding of planet Earth; and Cami, our real doctor, whose care and patience keep us healthy. You constant support has shaped my worldview, reminding me of the invisible connections that bind us despite the distance and keep us strong during the challenging times.

Thanks to my partner, Juancho — you’ve been the secret force behind my coding skills. You’re my personal stack overflow, patient and thorough in explaining coding concepts. Moreover, your caring companionship, morning teas, and soothing hugs eased my stress and brought me peace whenever I needed. Your support knows no bounds, and I am profoundly grateful for your presence in my life.

To my family — Juan Luis, your sense of responsibility has always been an inspiration for me; Guillermo, dad, your wisdom is my compass, and your words, always perfectly timed, lift burdens from my shoulders, remind me to be proud of my efforts, and inspire me to be a kind and respectful human above all else; Marta, mom, your abilities transcend any scientific discovery and somehow, even though I can’t find a way to explain it, your love ‘covers me with a mantle of protection and illuminates me with a light ray’. Your existence is the magic touch that a scientist needs to assign a probability to the impossible.
Lastly, to my grandmothers, Esther Duque and Zoraida Ruiz — though denied access to education themselves, their love and dedication forged a path for future generations of women to venture into the world guided by their innate curiosity. Their enduring legacy emboldens me to play a role in shaping a kinder, more equitable, and truth-driven society.

With immense gratitude and from the bottom of my heart, I thank you all.
Agradecimientos

Personalmente creo que la ciencia es un esfuerzo colaborativo de la humanidad para explicar el mundo. Las limitaciones inherentes a nuestra naturaleza física y biológica nos han hecho reconocer que sólo a través del trabajo colectivo es posible resolver algunas de las preguntas más complejas sobre el universo y la vida en la tierra. Esta tesis es tan solo una modesta contribución en el vasto océano de conocimientos que ha acumulado la humanidad durante su historia. Sin embargo, esta investigación fue posible gracias a la ayuda invaluables de personas que alimentaron mi pasión y apoyaron mi compromiso con la exploración científica durante estos cinco años.

Me gustaría empezar por agradecer a Art Poon, quien acogió a una estudiante internacional a pesar de las barreras idiomáticas y culturales. El proceso de aprendizaje de técnicas bioinformáticas es particularmente complicado dado que en ellas convergen conceptos, fórmulas y modelos de diferentes disciplinas. Art me guió pacientemente a través de las teorías y ecuaciones detrás de mis experimentos, y me permitió cultivar mi curiosidad científica. Su apoyo se expandió más allá del aspecto investigativo, mostrando un genuino interés tanto por mi éxito académico como por mi bienestar personal, y motivándome a explorar mis intereses por fuera de la academia mientras se aseguraba que cumpliera con todos los requerimientos necesarios para terminar mis estudios. Me siento realmente agradecida por la labor de Art como mi mentor y por su inmenso impacto en mi vida como científica.

Mi gratitud se expende a todos los miembros del laboratorio — Connor, John, Abayomi, Kaitlyn, Emmanuel, Erin, Gopi, Sareh, Roux-Cil, Amjad, Paula, and William — quienes contribuyeron no solamente a responder mis preguntas técnicas, sino que me ayudaron a refinar mis presentaciones y mis habilidades de escritura. Su camaradería tranformó mi tiempo en Western en un periodo de crecimiento académico y enriquecimiento personal.

A mis amigos canadienses — Tammy, Alicia, y Faisal — su compañía se convirtió en una parte fundamental de mi transición a tierras extrangleras, haciéndome sentir como en casa en otro continente. Charan y Brittany, ustes son una mezcla de humor y sabiduría que no solo me trajo muchísima felicidad sino que también mantuvo presente en mi la importancia de mantener una
relación emocionante y divertida con el mundo académico.

A mis amigos de Colombia. Helena, tu has sido para mi un apoyo invaluable en mi paso por el doctorado, ofreciéndome no solo apoyo emocional, sino también ayuda en las cosas más complicadas como escritos y presentaciones, hasta las cosas más simples como responder correos. Me siento extremadamente afortunada de saber que nuestra amistad perdurará sin importar dónde estemos. Cami, tus habilidades en bioinformática y tus conocimiento crítico sobre los aspectos más profundos de las sociedades humanas guían mis deseos sobre clase de científica que quiero ser. Dani, tu no solo leíste y corregiste mi tesis (lo cual considero algo invaluable), sino que además me has recomendado los mejores libros que han hecho más liviano mi paso por el doctorado. Muñoz y David, ustedes son mi fuente principal de risas. Se que puedo contar con ustedes para siempre que lo necesite. Mis amigas del colegio — Wendy, tu pasión por la evolución es realmente contagiosa y siempre inspira mi curiosidad y mi fascinación por el mundo natural; Eli, tu habilidad para encontrar belleza y detectar tesoros ocultos me llenan el corazón de alegría; Juli, tu amor por la ciencia y tu profundo conocimiento de los ecosistemas acuáticos nunca deja de emocionarme y sorprenderme; Sabi, tus enseñanzas de geología me impresionan y han engrandecido mi entendimiento del planeta que habito; y Cami, nuestra verdadera doctora, cuya paciencia y cuidados nos mantiene cuerdas y saludables. El apoyo de todas ustedes ha moldeado mi visión del mundo y me recuerda las conexiones invisibles que nos unen en la distancia y nos mantienen fuertes en la adversidad.

Gracias a Juancho, mi compañero de vida, a quien le debo enormemente por mis habilidades de programación. No se qué hubiera sido de mi si no hubieras respondido pacientemente a todas mis preguntas y buscado junto a mi los errores en mi código. Tu compañía tiene sobre mi un efecto relajante que se acentúa con los tés por la mañana, los abrazos largos y las palabras de aliento en los días difíciles. Tu apoyo no tiene límites, como no tiene límite mi gratitud por tu presencia en mi vida.

Finalmente quiero agradecer a mi familia — Juan Luis, tu inmenso sentido de la responsabili-
dad siempre ha sido para mi una fuente de inspiración; Guillermo, padre, tu sabiduría es mi brújula
y tus palabras, que llegan siempre en el momento perfecto, me liberan de las cargas autoimpuestas y me recuerdan que debo estar orgullosa de mis esfuerzo por hacer las cosas bien. Sobre todo, gracias pa’ por inspirarme a ser una persona respetuosa de la diferencia y cuidadosa con los demás. Marta, madre, tus habilidades trascienden cualquier conocimiento científico y, de alguna manera que nunca podré explicar, tu amor siempre me encuentra para ‘protegerme con el manto de la protección e iluminarme con el rayo de luz’. Tu existencia es el toque de magia que una científica necesita para poder asignarle una probabilidad a lo imposible.

Por último, quiero agradecer a mis abuelas, Esther Duque y Zoraida Ruiz. A pesar de que a ellas les fue negado el acceso a la educación, el amor y la dedicación de mis abuelas pavimentó el camino para que futuras generaciones de mujeres pudiéramos explorar el mundo guiadas por nuestra propia curiosidad. Su legado me motiva a contribuir a la construcción de sociedades más justas, equitativas y honestas.

Con inmensa gratitud, y desde el fondo de mi corazón, muchas gracias!
General Description

The ways in which life on Earth changes and adapts has always fascinated me, and when I discovered that viruses were an essential part of that process, I knew that studying them would be a captivating and fulfilling experience. This thesis represents the culmination of my years as a student at Western University, where I delved into the evolution of a specific arrangement of genes known as Overlapping Reading Frames (OvRFs) in virus genomes. My investigation into OvRF evolution can be divided into three main sections. Firstly, I sought to comprehend the significance of OvRFs and their prevalence in virus genomes at a global scale. Secondly, I developed a program to simulate the evolution of nucleotide sequences with OvRFs, and finally, I utilized this program to conduct a study of selection patterns in OvRFs under different coding contexts. In this manuscript, I articulated my research insights across four chapters:

1. Introductory Chapter[1] provides a comprehensive overview of OvRFs, describing their general characteristics and presenting the fundamental concepts that underpin my research on their evolution in virus genomes.

2. Subsequently, Chapter[2] focuses on my research concerning the global distribution of OvRFs. Additionally, it includes the description of a network-based representation of virus genomes that I developed to visualize and analyze various genome features. This work was independently published as a standalone research article, while the methods employed were documented in a separate Protocol paper.

3. Chapter[3] introduces HexSE, a simulation model for nucleotide sequence evolution along a phylogeny that tracks substitution rates at every nucleotide site. The resulting pipeline was documented and published as a standalone research article.

4. Finally, Chapter[4] describes how I utilize HexSE to validate comparative methods for detecting patterns of selection in OvRFs in virus genomes. This manuscript of this Chapter will be submitted for review.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>i</td>
</tr>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Summary for lay audience</td>
<td>iv</td>
</tr>
<tr>
<td>Co-Authorship Statement</td>
<td>v</td>
</tr>
<tr>
<td>Resumen</td>
<td>vi</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>vii</td>
</tr>
<tr>
<td>Agradecimientos</td>
<td>x</td>
</tr>
<tr>
<td>General Description</td>
<td>xiii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xviii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xix</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Overlapping reading frames (OvRFs) in virus genomes</td>
<td>3</td>
</tr>
<tr>
<td>1.1.1 Diversity of OvRFs across Baltimore groups</td>
<td>4</td>
</tr>
<tr>
<td>1.2 Understanding molecular evolution: methods, models, and simulations</td>
<td>7</td>
</tr>
<tr>
<td>1.2.1 Continuous-time Markov process and substitution models</td>
<td>9</td>
</tr>
<tr>
<td>1.2.2 Evolution in protein-coding sequences</td>
<td>12</td>
</tr>
</tbody>
</table>

xiv
5 Concluding remarks

5.1 Conclusions ................................................................. 147
5.2 Limitations ............................................................... 149
5.3 Future Directions ....................................................... 151
List of Tables

1.1 Methods to estimate selection in protein-coding sequences with Overlapping Reading Frames ........................................... 21

2.1 Sources and solutions to potential errors ........................................... 102

4.1 Simulation parameters for HexSE run replicates ................................. 133
List of Figures

1.1 Illustration of overlapping reading frames ............................................. 2
1.2 Baltimore classification of viruses based on their genetic material ............... 5
1.3 Basic terminology for the description of OvRF topology ............................ 7
1.4 Markov-chain models of nucleotide substitution ................................. 11
1.5 Counting synonymous and nononymous sites to estimate \( dN \) and \( dS \) rates in sequence pairs .................................................. 13
1.6 Simulating molecular evolution along a phylogeny .................................. 16
1.7 Effect of mutations in overlapping reading frames ................................. 19
1.8 Using \( k \)-mer counts to assess sequence similarity .............................. 30
1.9 Graphical and mathematical representation of a network .......................... 33
1.10 Network features of clusters of homologous proteins with overlapping reading frames .............................................................. 35
1.11 Triangle formation in adjacency graphs .............................................. 37
2.1 Distribution of overlapping genes across virus genomes ......................... 60
2.2 Associations between overlap lengths and frame shifts ......................... 62
2.3 Adenoviridae family analysis ............................................................... 64
2.4 Adjacency graphs for different virus families ....................................... 66
2.5 Forest plot of zero-inflated binomial regressions on adjacency graphs .......... 68
2.6 Notation of the 6 possible frameshifts used in this study and two other studies .... 75
2.7 Scatterplots of genome length on mean overlap length by Baltimore class .... 76
2.8 Creation of adjacency plots ................................................................. 77
2.9 Word clouds of protein names mapped to clusters for Adenoviridae .......... 78
2.10 Random introduction of 10% missanotation on the ORFs database .......................... 79
2.11 Random introduction of 50% missanotation on the ORFs database .......................... 80
2.12 Distribution of protein names across clusters in Papillomaviridae ............................ 81
2.13 Adjacency graph based on proteins clustered according to a pairwise alignment for Papillomaviridae .......................................................... 81
2.14 Distribution of protein names across clusters for Coronaviridae proteins ................ 82
2.15 Adjacency graph of Adenoviridae family ................................................................. 83
2.16 Downloading list with accession numbers .............................................................. 87
2.17 Downloading protein sequences .............................................................................. 89
2.18 Calculating distances between pairs of proteins ..................................................... 92
2.19 Clustering homologous proteins ............................................................................. 94
2.20 Visualizing the results of the clustering assignment .............................................. 98

3.1 Pipeline overview .................................................................................................... 114
3.2 Traversal of an event probability tree to select mutations ........................................ 117
3.3 Simulating Evolution on HBV ................................................................................ 119
3.4 Comparison of selection pressures on between gene C of HBV genotype F and two simulated alignments ................................................................. 122
3.5 Measuring HexSE computing times ........................................................................ 123

4.1 Obtaining parameters for HexSE runs from non-overlapping regions of gene P ...... 131
4.2 Change point detection in \(dS\) estimates on gene P of HBV under varying coding contexts ................................................................. 138
4.3 Changepoints detected under different penalty values evaluated with CROPS ......... 139
4.4 Effect of sequence divergence in the classification of overlapping regions. ............ 140
4.5 True positive rate when varying \(dN\) parameters ....................................................... 141
Chapter 1

Introduction

While working on the foundations of genetic sequencing techniques, Barrel and collaborators [1] encountered a puzzling scenario: a discrepancy between the number and molecular weights of proteins produced by the bacteriophage ϕX174, and the length of its genome. An analysis of this first sequenced DNA genome, revealed the nucleotides used to translate the scaffolding gene were shared by the replication gene, and that the lysis gene was embedded entirely within the external scaffolding gene [2]. The term ‘overlapping genes’ [1] was thereby introduced to describe the occurrence of protein-coding sequences (CDSs) that share the same nucleotides.

As we gained a better understanding of the molecular mechanisms that shape the genomes of living organisms, the annotation of CDSs and the discovery of novel open reading frames (ORFs) became a common feature in genetic research. Thus, cases of overlapping reading frames (OvRFs) have now been documented in all types of genetic material including chromosomes, mitochondrial, and bacterial genomes [3–6]. Because OvRFs were first documented in the short genome (5,386 nts) of ϕX174, their presence became associated with the need to maximize the amount of genetic information that could be encoded in small genomes [7]. However, the current reports of OvRFs across all living organisms have challenged these preconceptions. For example, recent work on the human genome estimated that around a quarter of all annotated protein-coding genes are involved in OvRFs [8]. Moreover, nested genes that are completely encoded within the bounds of larger ones, and seem to be a result of a neutral evolutionary process in vertebrates, fruit flies, and nematodes genomes [9]. OvRFs are also ubiquitous features across virus and bacterial genomes, where
Figure 1.1: Illustration of Overlapping reading frames (OvRFs). Discovered in the circular single-stranded DNA genome of the bacteriophage φX174 (represented as the blue circle, where letters A-K denote open reading frames), OvRFs are nucleotide sequences with multiple signals that trigger the synthesis of different proteins. In the diagram, a ribosome recognizes an AUG as a signal to translate a protein in blue. A shift of one nucleotide leads to the recognition of a second start signal for the synthesis of the protein in orange.

Some estimates suggest that approximately one third of the genes are involved in overlaps [6]. In particular, OvRFs can persist for long evolutionary time spans in prokaryotes, where homologous overlapping gene pairs can be identified in different species when the overlap confers a selective advantage [10].

In viruses, OvRFs are hypothesized to be used to encode proteins more compactly and regulate transcription [11]. In addition, OvRFs might be a source of gene novelty, facilitating the creation of new open reading frames (ORF) on top of parental CDSs [12]. This chapter provides an overview of our current understanding of OvRFs in viruses, and the methods employed to investigate the underlying selection processes that shape evolution in virus genomes.
1.1 Overlapping reading frames (OvRFs) in virus genomes

Defining what a virus actually is remains a non-trivial task due to the unique position they occupy in the evolutionary history of all living organisms. In fact, the emergence and nature of viruses are subjects of great debate. One hypothesis attributes the origin of viruses to genetic elements that have escaped from cellular forms to become autonomous entities that selfishly make copies of themselves [13]. On the other hand, the lack of close homologous genes between cellular life forms and viruses seems to support an opposing model of viral origins in a pool of selfish replicators that existed before modern cells with large double-stranded DNA [13, 14]. Regardless which model is correct, we now know that viruses have played a major role in the evolution of life and share three common features: 1) viruses have genetic information contained in DNA or RNA molecules, 2) the genetic information is protected by a capsid or envelope, and 3) viruses exploit the resources and physiological processes of host cells to make copies of their genetic material and synthesize the proteins required to produce new viral particles that can be released to infect new cellular hosts [15].

Different mechanisms have been proposed to explain the abundance of OvRFs in virus genomes. First, reports of a significant correlation between capsid size and genome length suggest that OvRFs are used by viruses to overcome the physical restriction on genome size [16–18]. For instance, according to Chirico and colleagues [11], argued that small viruses with icosahedral capsids face limitations in adjusting their capsid size. Icosahedral capsids are formed by assembling 60 structural units into a geometrical shape with 20 equilateral sides. Increasing the capsid size would require to also increase the number of structural units required for making each side — T number. Consequently, the acquisition of new virus functions is more likely to occur through overlapping genes than through increasing the genome length, which has an upper limit imposed by the capsid size.

Another theory states that OvRFs could be used by viruses as a mechanism to reduce the fitness cost incurred by the accumulation of deleterious mutations, known as the mutation load [11]. High
mutation rates are advantageous for viruses because maintaining high replication fidelity leads to slower replication rates. In other words, viruses can be either quick or accurate, but not both \[19, 20\]. Furthermore, a rapid accumulation of mutations can be advantageous for organisms that need to adapt to constantly changing environments \[21\]. In either scenario, as mutations occurring in OvRFs have the potential to simultaneously affect functions encoded in different reading frames, gene overlap could be employed by viruses to reduce mutation load. By amplifying the deleterious effects of mutations, gene overlap enables selection to more effectively eliminate these mutations from the viral population \[22\].

Finally, overlapping genes have been proposed as a mechanism for generating gene novelty. The concept of *de novo* genes was initially introduced by Pierre Grasse in 1977 \[23\]. According to his theory, genes could arise within existing reading frames through a process known as **overprinting**, that occurs when mutations in an open reading frame enable the expression of a second reading frame while preserving the function of the first one, resulting in an overlapping gene arrangement \[24\].

To test the overprinting hypothesis, Keese and Gibbs \[5\] conducted a phylogenetic analysis of overlapping gene pairs in tymovirus, luteovirus, and lentivirus. They observed that one gene tends to be restricted to a particular lineage (overprinted), while the other is more widely distributed across the phylogeny (parental), providing evidence of new proteins being expressed from nucleotides within previously established coding sequences. Subsequently, Sabath and colleagues \[25\] examined 12 viral genes generated by overprinting and estimated their relative ages. Their findings suggest that young overlapping genes evolve rapidly and experience positive or weak purifying selection, in contrast to their ancestral overlapping counterparts that are subject to strong purifying selection.

### 1.1.1 Diversity of OvRFs across Baltimore groups

Classifying Earth’s virome is a daunting task due to the vast diversity of viruses capable of infecting all forms of life. While RNA viruses and reverse-transcribing viruses are thought to
1.1. OVERLAPPING READING FRAMES (OvRFs) IN VIRUS GENOMES

Example

![Virus Icons]

Table: Baltimore classification of viruses based on their genetic material

<table>
<thead>
<tr>
<th>Group I</th>
<th>Group II</th>
<th>Group III</th>
<th>Group IV</th>
<th>Group V</th>
<th>Group VI</th>
<th>Group VII</th>
</tr>
</thead>
<tbody>
<tr>
<td>dsDNA</td>
<td>ssDNA</td>
<td>dsRNA</td>
<td>+ssRNA</td>
<td>-ssRNA</td>
<td>+ssRNA-RT</td>
<td>dsDNA-RT</td>
</tr>
</tbody>
</table>


Although OvRFs are ubiquitous in viruses, there are substantial differences in the frequency and type of overlaps across the different Baltimore classes. For instance, 61% of dsDNA virus genomes and 65% of ssDNA virus genomes contain some type of gene overlap, whereas only 43%
of ssRNA, 24% of dsRNA, and 19% of dsRNA genomes are annotated with overlapping regions \[29\]. Interestingly, I found that longer genomes of DNA viruses tend to carry shorter overlaps than those of RNA viruses, resulting in a negative correlation between genome length and the number of nucleotides involved in an overlap (Chapter \[2\]). This trend might be attributed to differences in mutation rates, which range between \(10^{-8}\) to \(10^{-6}\) substitutions per nucleotide site per replication cycle in DNA viruses, while, in RNA viruses, the range is between \(10^{-6}\) and \(10^{-4}\). Thus, OvRFs may provide genomic stability to RNA viruses \[30\], counteracting the fitness cost associated with the high mutation rates induced by the lack of proofreading activity, as well as their need for rapid replication to complete their life cycles \[31\].

Overlapping genes can be encoded in either the same (+) or opposite (−) strand relative to the ‘upstream’ or parental ORF, and they can be shifted by one or two nucleotides, resulting in frameshifts +2, +1, −0, +0, −1, −2 with respect to the parental reading frame — which is usually the longer gene in the pair (Figure \[1.3\]). Previous research \[32\] \[33\] has demonstrated a preference for the +1 frameshift during overprinting, as it introduces fewer stop codons and more start codons, benefiting from the specific arrangement of nucleotides in the genetic code. This phenomenon also explains why the long overlaps of RNA viruses are primarily involved in +1 frameshifts. Furthermore, the prevalence of +1 frameshifts in long overlaps can be attributed to the preservation of similar physicochemical properties of the amino acids between the parental protein and the overprinted product \[34\].

Viruses have harnessed every conceivable means to generate proteins by producing mRNA from the two types of nucleic acids (DNA and RNA) \[26\]. Their use of diverse mechanisms to complete their life cycles, coupled with their high mutation rates, has resulted in a wide range of genomes with variations in size, shape, and packaging. The widespread occurrence of OvRFs in virus genomes raises questions about their role in virus evolution. Our most informed hypotheses suggest that OvRFs in virus genomes facilitate exploration of novel functions through overprinted proteins, expand storage capacity by increasing the number of genes despite constraints imposed by the capsid size, and regulate gene expression. Overall, the presence of OvRFs highlights the
1.2. Understanding molecular evolution: methods, models, and simulations

Life on Earth began over 3.7 billion years ago [35], when organisms thrived in diverse environments, developed various survival mechanisms, and progressively increased in complexity. To comprehend the complexity of biological systems, evolutionary models seek to identify patterns in observed variation that can be used to inform mathematical models. However, these models often encounter limitations when faced with the vast number of possibilities that could explain our observations of the natural world, especially when we do not fully comprehend the specific dynamics of the system, making it challenging to handle under mathematically exact conditions [36].

Computer simulations provide a powerful framework to characterize the patterns of dynamic systems under varying conditions, such as populations of evolving organisms. With simulations, we can evaluate computational methods using data with known ‘ground truth’, which can sub-
sequently be used to fit a model to empirical data. For instance, Posada and Crandall [37] used simulations to re-create recombination events — exchange of genetic material as a result of temporary interactions of homologous genetic material from different individuals. They used these simulations to evaluate the ability of distance-based, phylogenetic, and substitution distribution methods to detect recombination. Similarly, Anisimova and collaborators [38] employed computer simulations to assess the accuracy of selection estimates (as described in section 1.2.2) in the presence of recombination events. By simulating sequence evolution under different levels of recombination, the authors demonstrated that methods used to detect positive selection are biased when analyzing sequences that are the result of recombination events, such that the sudden introduction of multiple genetic changes could resemble the effects of molecular adaptation. In other words, high recombination rates produce variation among sequences that could erroneously be interpreted as diversifying selection.

Assessing evolution in overlapping reading frames (OvRFs) has presented a longstanding challenge in virology and evolutionary biology (as shown in section 1.3). To address this problem, I developed a simulation pipeline (described in Chapter 3) aimed at exploring multiple selection scenarios and models to gain a deeper understanding of the selection processes that shape OvRFs in viruses. The pipeline is based on a widely adapted model in evolutionary simulation called the continuous-time Markov process. However, unlike methods that assume independent evolution of sites, my implementation allows the recreation of a system where instantaneous substitution rates (the rate at which a substitution occur in a given nucleotide) are not constant over time; instead, they are updated in response to changes at adjacent sites [39]. The following section provides an overview on the use of continuous-time Markov to understand molecular evolution, and describes the basic components of these models that are required to simulate evolution, including the different pipelines created for this general purpose.
1.2.1 Continuous-time Markov process and substitution models

Continuous-time Markov processes or ‘chains’ are a category of probabilistic models that describe changes between states of a system that appear to vary randomly over time. Evolution is the heritable change in the genetic composition of a population over time. Thus, Markov chains are a natural fit to describe evolution. For instance, Markov chains are frequently used to describe substitutions at different sites of a nucleotide sequence over time, where the four nucleotides correspond to different discrete states of the chain. A defining feature of a Markov chain is that it has no memory. In other words, the probability that the system occupies a particular state at some point in the future depends only on the current state, and not any previous events, i.e., how the current state was reached [40]. A Markov chain is primarily defined by a rate matrix \( Q = \{q_{ij}\} \), where \( q_{ij} \) represents the instantaneous rate of change from state \( i \) to \( j \). For instance, this change may represent a substitution from nucleotide \( i \) to nucleotide \( j \) with \( (i, j) \in \{A, C, G, \text{or} \ T\} \). Indeed, several models of nucleotide substitution are described as Markov chains. For instance, the Jukes-Cantor model (JC69) [41] assumes that all nucleotides change to any other nucleotide at the same rate \( \lambda \). In this model, the substitution rate matrix \( Q \) describing the rate at which A, C, G, and T mutate is:

\[
Q = \{q_{ij}\} = \begin{bmatrix}
-3\lambda & \lambda & \lambda & \lambda \\
\lambda & -3\lambda & \lambda & \lambda \\
\lambda & \lambda & -3\lambda & \lambda \\
\lambda & \lambda & \lambda & -3\lambda
\end{bmatrix}
\] (1.1)

Note that each row of the matrix sums to 0 as the total rate of substitutions of any nucleotide \( i \) to any other nucleotide \( j \neq i \) is \( 3\lambda \). Moreover, \( q_{ij}\Delta t \) gives the rate at which any given nucleotide \( i \) will change to a different nucleotide \( j \) in an infinitesimally small interval \( \Delta t \).

To fit the Markov chain to observed state changes over time, we need to calculate transition probability \( p_{ij}(t) \) that a nucleotide \( i \) will change to nucleotide \( j \) after some time \( t \). These transition probabilities comprise the transition probability matrix \( P(t) \), which can be computed from \( Q \) by
matrix exponentiation. For the JC69 model where all nucleotides have the same substitution rate \( \lambda \), the transition probability matrix is [40]:

\[
P(t) = \{p_{ij}(t)\} = e^{Qt} = \begin{pmatrix}
    p_0(t) & p_1(t) & p_1(t) & p_1(t) \\
    p_1(t) & p_0(t) & p_1(t) & p_1(t) \\
    p_1(t) & p_1(t) & p_0(t) & p_1(t) \\
    p_1(t) & p_1(t) & p_1(t) & p_0(t)
\end{pmatrix},
\]

with \( p_0(t) = \frac{1}{4} + \frac{3}{4} e^{-4\lambda t}, \)

\( p_1(t) = \frac{1}{4} - \frac{1}{4} e^{-4\lambda t} \) \hspace{1cm} (1.2)

An important feature of this transition probability matrix is that every row sums to 1 because the chain has to be in one of the four nucleotide states at time \( t \). As the amount of elapsed time \( t \) approaches 0, \( e^{-4\lambda t} \rightarrow 1 \). Hence the diagonal entries \( p_0 \) converge to 1 and the off-diagonal entries \( p_1 \) converge to 0, reflecting the case of no evolution. Conversely as \( t \rightarrow \infty \), \( e^{-4\lambda t} \rightarrow 0 \), and the off-diagonal entries \( p_{ij}(t) \) converge to 1/4 for all \( i \neq j \). In other words, over long periods of time, so many substitutions have occurred at every site that the target nucleotide is random, with probability 1/4 for every nucleotide, irrespective of the starting nucleotide state. This situation is referred to as saturation. In general, if the probability of the nucleotides follow a stationary distribution, then probability that the chain is in state \( j \) when \( t \rightarrow \infty \) is represented by \( \pi_j \). This quantity can also be interpreted as the expected proportion of sites in a nucleotide sequence in state \( j \) after an infinite amount of evolution [40].

Since the total rate at which any nucleotide change is 3\( \lambda \), the JC69 model estimates the probability \( p \) of a nucleotide in a descendant sequence being different from a nucleotide in an ancestral sequence as:

\[
p = 3p_1(t) = 3/4 - 3/4 e^{-4\lambda t} = 3/4 - 3/4 e^{-4\lambda t/3}
\] \hspace{1cm} (1.3)

From here, we can estimate the distance \( \hat{d} \) between two sequences (in terms of the expected number of nucleotide substitutions per site) by equating this Equation 1.3 to the observed proportion of differences \( \hat{p} \):
1.2. Understanding molecular evolution: methods, models, and simulations

Figure 1.4: Markov-chain models of nucleotide substitution. In the JC96 model [41], all substitutions occur at the same rate $\lambda$. The K2P [42] model introduces the parameters $\alpha$ and $\beta$ to differentiate rates at which transitions or tranversions occur. In the HKY model [43] each nucleotide has a different stationary frequency $\pi$, to represent the expected proportion of the nucleotide as the sequence evolves over long periods of time.

$$\hat{d} = -\frac{3}{4} \ln(1 - 4/3\hat{p})$$ (1.4)

Note that $\hat{d}$ is always greater than or equal to $\hat{p}$, a quality that reflects the effect of multiple substitutions occurring at the same site (multiple hits).

Subsequent models have extended this model, incorporating additional parameters to account for various mutation patterns observed in sequences (Figure 1.4). For example, the Kimura 2-parameter model (K2P) [42] distinguishes between a transition rate $\alpha$ for substitutions between two purines (molecules with two rings: $A \leftrightarrow G$) or two pyrimidines (molecules with a single ring: $T \leftrightarrow C$), and a transversion rate $\beta$ for substitutions between purines and pyrimidines, or vice versa ($A, G \leftrightarrow C, T$). However, similar to JC69, K2P maintained an equal nucleotide proportions across the sequence $\pi_i = 1/4$, meaning that, when the substitution process reaches equilibrium, the proportions of each nucleotide will be equal across the sequence. In 1985, the Hasegawa-Kishino-Yano model (HKY) [43] proposed to account for base composition biases by specifying the stationary nucleotide frequencies as additional model parameters denoted as $\pi_A, \pi_C, \pi_G,$ and $\pi_T$, that represent the proportion of the nucleotides, as $t \to \infty$ as mentioned above. Later, the Tamura and Nei model (TN93) [44] extended this idea and introduced two additional parameters.
to model different transition rates between purines ($A \leftrightarrow G$) and pyrimidines ($T \leftrightarrow C$).

### 1.2.2 Evolution in protein-coding sequences

Sometimes, nucleotide substitutions lead to adaptation in populations of organisms in response to their environments by increasing their fitness or their ability to survive, grow, and reproduce. Studying how evolution influences the adaptative processes of viruses and other organisms often involves evaluating the effects of specific nucleotide substitutions on the amino acid residues encoded by protein-coding sequences. Current techniques focus on comparing the impact of two types of mutations on fitness: **nonsynonymous substitutions**, which alter the protein product and potentially incur evolutionary costs, and **synonymous mutations**, which are assumed to be neutral, but may have effects in protein expression [45].

Codon substitution models are used to evaluate evolution in protein-coding sequences. A codon is a triplet of nucleotides that are translated into a particular amino acid. A codon substitution model replaces the nucleotide with the codon as the basic unit of evolution. Thus, codon models confer the ability to account for the redundancy of the genetic code in cases where different triplets encode the same amino acid residue. For this purpose, codon models use the parameter $\omega$, which is calculated as the ratio between the number of synonymous substitutions per synonymous site (denoted as $dS$ or $K_S$) and the number of nonsynonymous substitutions per nonsynonymous site (denoted as $dN$ or $K_A$) [46–48], and is used as a means to characterize selection in nonsynonymous mutations.

The counting method developed by Nei and Gojobori (NG86) [49] estimates $dN$ and $dS$ in three steps (exemplified in Figure 1.5):

1. Count the expected number of synonymous ($ES$) and nonsynonymous substitutions ($EN$) per site. Since each codon has three nucleotide sites, they can be affected by 9 possible changes. Such changes can be classified into synonymous and nonsynonymous categories according to the residue the codon produces if a substitution occurs.
1.2. Understanding molecular evolution: methods, models, and simulations

Figure 1.5: Counting synonymous and non-synonymous sites to estimate $dN$ and $dS$ rates in sequence pairs. (Upper panel) A codon consists of 3 nucleotide sites, with each site capable of undergoing any of the 9 possible nucleotide substitutions. (Down left) To estimate the expected number of substitutions between a pair of sequences under neutral evolution, we calculate the count of synonymous and non-synonymous sites within each codon of the sequences. Then, we determine the average number of expected mutations between the two sequences. (Down right) The ratio of observed to expected substitutions gives the value of $dN/dS$. A $dN/dS < 1$ indicates that non-synonymous substitutions are being removed from the population as an effect of natural selection as fewer cases are observed than expected. (Figure inspired from slides by Sergei L Kosakovsky Pond https://www.hyphy.org/resources/slides-selection-2016.pdf, and Yang, 2006 [40])
2. Count synonymous and nonsynonymous differences between two sequences by evaluating, codon by codon, the number of synonymous ($S$) and nonsynonymous ($N$) substitutions observed.

3. Calculate the proportions of differences at the synonymous and nonsynonymous sites at the codon:

$$dS = S/ES,$$

$$dN = N/EN$$

(1.5)

Note that the NG86 model assumes a simplified model of evolution that does not account for the possibility of multiple substitutions at the same site. This would lead to misleading estimates on the number of synonymous and nonsynonymous substitutions required to change, for example, the codon $ACT$ to $AGG$. This particular issue was addressed by the codon models proposed by Goldman and Yang (GY94) [50] and Muse and Gaut (MG94)[51], where multiple substitutions might be explained by several single nucleotide steps, so $AGG$ could be reached from $ACT$ via $ACT \Rightarrow AGT \Rightarrow AGG$ [50]. This is possible because, instead of counting sites, the GY94/MG94 models use a maximum likelihood approach to fit a continuous time Markov model to data in order to estimate the rate of accumulation of synonymous and nonsynonymous substitutions. Under the codon models, the probability of a codon transitioning from one state to another in a Markov chain can be estimated by evaluating the instantaneous rate of a codon $i$ changing to codon $j$:

$$q_{ij} = \begin{cases} 
0, & \text{if } i \text{ and } j \text{ differ at two or three codon positions,} \\
\mu \pi_j \alpha, & \text{if } i \text{ and } j \text{ differ by a synonymous substitution,} \\
\mu \pi_j \beta, & \text{if } i \text{ and } j \text{ differ by a nonsynonymous substitution} 
\end{cases}$$

(1.6)

Here, $\pi_j$ is the equilibrium frequency of codon $j$, $\mu$ is a scaling factor (affected by the transition-transversion rate ratio $\kappa$), $\alpha$ is the rate of accumulation of synonymous substitutions, and $\beta$ is the rate of accumulation of nonsynonymous substitutions. To estimate the values for parameters $\alpha$...
1.2. UNDERSTANDING MOLECULAR EVOLUTION: METHODS, MODELS, AND SIMULATIONS

and $\beta$, the GY94/MG94 method propose a maximum likelihood (ML) approach that evaluates the likelihood of the observed sequence data under a set of parameter values for the model. The parameter values that maximize the likelihood of the data are chosen as the best estimates.

Subsequent improvements to the GY94/MG94 model include incorporating more realistic models of codon substitution patterns [48] and accounting for factors like biases in codon usage and variation in selective pressures across different lineages [52]. Advancements in statistical techniques and computational power have enabled the development of site-specific models that consider variation of selection pressures across individual sites within a coding sequence [53]. Furthermore, codon models have been expanded to include other evolutionary scenarios such as multiple-hits, where random substitutions occur in tandem at adjacent nucleotides, affecting the same codon [54].

1.2.3 Simulating evolution

Simulation methods of molecular evolution require a ‘known’ phylogenetic tree to randomly sample nucleotide substitutions from a sequence along the tree branches. First, the simulation is configured to use a particular instantaneous rate matrix $Q$ which represents a model of evolution described by a Markov chain. The values on each cell of the matrix are calculated based on mutation biases such as the transition-transversion rate ratio $\kappa$, the synonymous-nonsynonymous rate $\omega$, and the stationary nucleotide frequencies $\pi$.

The process generally begins with an initialization step (Figure 1.6) that assigns the nucleotide sequence to the root of the phylogeny — where branch lengths represent an expected number of substitutions. To simulate sequence evolution along a branch of length $t$ (defined by the average number of nucleotide or protein substitutions per site), the rate matrix of the Markov chain $Q$ is calculated using the specified mutation biases derived from the chosen substitution model. Additionally, the total instantaneous mutation rate is computed as the sum of substitution rates across sites ($\Lambda = \sum_{ij} q_{ij}$). The waiting time until the next substitution event follows an exponential distribution with a mean of $1/\Lambda$. Consequently, a random waiting time $s$ is drawn from the exponential
Figure 1.6: **Simulating molecular evolution along a phylogeny.** Substitution events are stochastically drawn based on a probability distribution proportional to the transition probabilities of nucleotides, which follow a Markov chain. The simulation process involves sampling an exponentially distributed waiting time, denoted as $s$, which is determined by the cumulative instantaneous substitution rates of the nucleotides. The mutation biases depend on the model of molecular evolution implemented in the simulator. In this example, substitution rates are calculated considering the baseline mutation rate ($\mu$), the transition/transversion rate ratio ($\kappa$), the stationary frequency of the target nucleotide ($\pi_j$), and the synonymous/nonsynonymous rate ratio ($\omega$) (as implemented in HexSE — Chapter 3).

distribution ($s \sim \exp(-\Lambda)$). If $s$ is smaller than number of expected substitutions at the branch ($s < t$), a substitution event from $i$ to $j$ occurs with a probability proportional to $q_{ij}/\Lambda$ and the sequence is updated accordingly. Otherwise, no event occurs and the sequence is stored at the subsequent node of the tree. The remaining time for the branch becomes $t - s$, and the process is iterated until the branch time is exhausted as shown in Figure 1.6.

**Simulation Pipelines**

One of the earliest examples of a computer program to simulate evolution of a nucleotide sequence was Seq-Gen, written in the C programming language by Rambaut and Grassly in 1997 [55]. Seq-Gen was designed to simulate evolution along a phylogeny, using the Hasegawa-Kishino-Yano (HKY), Kimura 2-parameter (K2P) and Jukes-Cantor (JC69) substitution models. This
program takes one or more rooted phylogenetic trees as an input, and simulates sequences evolving along the branches on each tree starting from the root. That same year, Stoye and his team \cite{56} also used C to develop Rose, a similar program to evolve sequence over branches, but with the ability to also simulate evolution on amino acid sequences, as well as insertions and deletions. Nevertheless, neither of these programs models the proportion of sites that remain constant over time (i.e., invariant sites) or simulate exchange of genetic material by recombination. To fill this gap, Cartwright developed Dawg in 2005 using Perl as the programming language \cite{57}.

Programs such as MySSP \cite{58}, SIMPROT \cite{59}, indel-Seq-Gen \cite{60} and EvolveAGene 3 \cite{61} were later developed to address different needs of the research communities. For example, MySSP considers non-stationary processes. In other words, the stationary nucleotide frequency $\pi$ changes over time, allowing users to independently specify evolutionary models for each branch. SIMPROT models indels in distantly related protein sequences using the Qian-Goldstein distribution \cite{62} — obtained for the length of indels from a database of protein alignments with long sequence divergence (sharing no more than 25% sequence identity). Indel-Seq-Gen \cite{60} provides a flexible program to evolve amino acid sequences starting from one or multiple sequences at the root of the tree. Additionally, indel-Seq-Gen simulates evolution at different rates for multidomain protein families and incorporate indels with different lengths based on a distribution customised by the user. EvolveAGene 3 \cite{61} allows the user to specify selection conditions over the sequence, so it is possible to distinguish between selection intensities over the branches.

More recently, Spielman and Wilke \cite{63} developed Pyvolve, a simulation program written in Python. The main advantage of Pyvolve is the flexibility to select different models of evolution for nucleotide, amino acid, or codon sequences. The fact that Pyvolve is flexible and open source is a major improvement over other simulation programs. To address the challenges posed by extinct lineages and the horizontal transmission of genes (such as lateral gene transfers) involving species absent from the phylogeny, Davin and colleagues developed a Python-based pipeline called Zombi \cite{64}. In Zombi, users can specify extinction and speciation rates, allowing for the simulation of species phylogenies, genomes, and sequences at the codon, nucleotide, or amino acid level along
the branches of the evolutionary tree.

In response to the growing need for more sophisticated evolutionary models that capture the complexity of the evolutionary landscape, numerous novel models have been proposed. For example, functional constraints on sites in a gene often cause shifts in site-specific evolutionary rates [65]. To account for different mutation rates (heterogeneity) between genes, partition models classify sites based on the gene or codon composition category to which it belongs, and assign them to independent blocks with their own model of nucleotide substitution [66]. Similarly, mixture models combine information about different models and compute a single likelihood for each site [67] to address the possibility that more than one model can apply to the same site.

Different parts of the genome can have distinct evolutionary histories due to introgression (transfer of genetic information between species that are not completely reproductively isolated), horizontal gene transfer (movement of genetic information between distantly related organism), or recombination [68]. To model substitutions in sequences with different evolutionary paths, multi-tree models use a mixture of phylogenetic trees to represent multiple histories for a single sequence alignment. Many of these models have been integrated into widely used programs for phylogeny inference, such as IQ-TREE [69]. Furthermore, Ly-Trong and colleagues developed AliSim [70], an extension of IQ-TREE that offers the capability of studying the evolutionary processes underlying empirical alignments by accommodating the diversity of multiple evolutionary models into a single framework.

### 1.3 Estimating evolution in overlapping reading frames

As described in Section [1.2.2] the rate ratio of accumulation of nonsynonymous ($dN$) to synonymous ($dS$) substitutions ($dN/dS = \omega$) is used to measure the strength of selection acting on a protein-coding sequence. For instance, a surplus of synonymous over non-synonymous substitutions ($\omega < 1$) is indicative of purifying selection, meaning that substitutions that induce amino acid changes are being removed from the population by natural selection. Most genes tend to be
1.3. Estimating evolution in overlapping reading frames

Figure 1.7: Effect of mutations in overlapping reading frames. When a substitution occurs, it can differentially affect the amino acid residues of the overlapping proteins. In this example a mutation $C \rightarrow A$ in position 15 of an hypothetical coding sequence with and overlap is synonymous in the first open reading frame but nonsynonymous in the second one, where the codon that used to encode a Proline (P) now encodes a histidine (H).

under predominantly purifying selection since existing genes are the product of many generations of adaptation. Conversely, diversifying or positive selection ($\omega > 1$) indicates that amino acid changes are contributing to adaptation to changing environmental conditions, and are therefore beneficial for the population.

To uncover the primary targets of natural selection, various models have been developed to estimate the selection parameter, $\omega$. Traditionally, evolutionary models assume independent evolution of codon sites, allowing efficient calculation of model likelihoods through multiplication across codon sites. However, this assumption does not hold true for OvRFs as each substitution has the potential to alter the context for subsequent mutations at neighboring sites. In other words, a single nucleotide change may induce substitution events at two different and overlapping codon sites.

In addition, a substitution that is synonymous in one reading frame may be non-synonymous in another (Figure 1.7). Because we expect a majority of sites to be under purifying selection, this should cause a depletion of synonymous substitutions in the other reading frame. For instance,
early studies on the Hepatitis B virus (HBV) revealed a significantly lower number of synonymous substitutions in gene S, attributed to its complete nesting within gene P \cite{71}. Consequently, estimates of $dS$ in overlapping genes can be artefactually decreased due to purifying selection in other reading frames, leading to inflated $\omega$ values and potentially misleading conclusions regarding positive selection \cite{30,72}. Moreover, by counting the number of substitutions in sequence alignments of pairs of overlapping genes, Angelo Pavesi observed that in 32 out of 65 cases, there are significant differences in selection pressures on the two proteins — this is, genes are under asymmetric evolution \cite{73}.

Some of the techniques that have been developed to address the problem of measuring selection in OvRFs are documented in Table \ref{table:selection_techniques} \cite{74}. Based on the foundational nucleotide-based model developed by Nei and Gojobori to calculate $dN$ and $dS$ \cite{49}, Hei and Støvblæ \cite{75} developed a maximum likelihood method that accounts for overlapping genes by partitioning sites into degeneracy classes associated to the selection strengths on each protein. The classes were defined based on the amino acid changes produced by varying any nucleotide in a codon among the four possible nucleotides. Thus, substitutions would results either in the same amino acid (four-fold degenerate), two amino acids, each pair different by a transition (two-fold degenerate), or four different amino acids (non degenerate). To estimate selection on a site with an OvRF, the method combines the effect of the two classes (one for each CDS) when the substitution is nonsynonymous in both CDSs.

In 1996, Felsenstein and Churchill \cite{76} proposed to use a hidden Markov model (HMM) to represent the correlated changes in substitution rates across a sequence alignment. This model assumes that each site evolves according to one of a finite number of rates. The assignment of rates is assumed to follow a Markov chain as we move across the alignment, with random transitions between rates. We cannot directly observe which sites evolve under which rates. Instead, these hidden states must be inferred from the observed numbers of substitutions at each site. The likelihood of the data is thereby averaged over all possible assignments of rates, weighted by their respective probabilities. McCauley and collaborators \cite{77} adapted this HMM framework to extend
### 1.3. Estimating Evolution in Overlapping Reading Frames

<table>
<thead>
<tr>
<th>Reference</th>
<th>Method Description</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hei and Støvlbæ, 1995</td>
<td>Maximum Likelihood method to estimate selection strength on a nucleotide as a combination of degeneracy classes (one for each reading frame to which a site belongs)</td>
<td>For each site, the class is assumed to be fixed across the entire phylogeny, and it doesn’t allow different selection pressures within the same ORF</td>
</tr>
<tr>
<td>Pedersen and Jensen, 2001</td>
<td>Codon-based model, where exact parameters values are estimated for the different selective constrains in OvRFs</td>
<td>Very computationally-expensive, requires a pairwise alignment</td>
</tr>
<tr>
<td>McCauley et al., 2007</td>
<td>Hidden Markov Model extending the degeneracy class idea proposed by Hei and Støvlbæ</td>
<td>Constant degeneracy class for each site, whereas degeneracy changes over time as substitutions occur</td>
</tr>
<tr>
<td>Sabath et al., 2008</td>
<td>Codon-based model where selection strength for each nucleotide is estimated as the product of the $\omega$ value for each ORF</td>
<td>Only implemented for pairs of sequences. Amino acid diversity between the two proteins has to be smaller than 50% or greater than 5%</td>
</tr>
<tr>
<td>Wei and Zhang, 2015</td>
<td>Estimates $\omega$ for each ORF by counting the proportion of sites under four categories (NN, NS, SN, SS)</td>
<td>Single selection estimate for the entire gene, only implemented for pairs of sequences</td>
</tr>
<tr>
<td>Nelson et al., 2020</td>
<td>Extension of the Wei-Zhang method, but using a minimal overlapping unit of 6 nucleotides and counting single nucleotide differences based on an alignment</td>
<td>Single selection estimate for the entire gene, loss of power for pairwise distance $&gt; 0.1$ and neighboring variants</td>
</tr>
</tbody>
</table>

Table 1.1: Methods to estimate selection in protein-coding sequences with Overlapping Reading Frames
Hei and Støvlbæ’s method to measure selection in OvRFs. This enabled them to infer multiple levels of selection pressures by allowing sites to choose from a given set of selection strengths. A major limitation of the Hei-Støvlbæ and McCauley et al. methods is that they both assume a constant degeneracy class for each site; however, degeneracy changes over time as substitutions occur.

Rather than relying on codon degeneracy classes, Pedersen and Jensen [78] extended the codon-based approach by Goldman and Yang [50], creating an exact evolutionary model with parameters that represent the degrees of selection constraints operating in the different reading frames, and estimated those parameters using Markov chain Monte Carlo (MCMC). However, the precision of this method was overshadowed by the extensive computational power required to calculate the likelihood of a model with independent substitution rate parameters for each site. Similarly, Sabath and collaborators [79] estimated selection by calculating a substitution rate matrix for each gene depending on the instantaneous rate of a sextet (or hexamer), namely, the minimum unit of evolution composed by a codon (the reference codon), and the adjacent nucleotides from the overlapping codons assumed to be at +2, +1, −1, or −2 frameshifts (see Figure 1.3). Substitution rates are then calculated by multiplying the $\omega$ ratios of each codon that the nucleotide is part of. With this method, $\omega$ values above 1 could indicate either neutral or positive selection, since it does not test statistically for positive selection and, importantly, it was only implemented for pairs of sequences.

To overcome these issues, Wei and Zhang [80] proposed to estimate selection independently on ORF1 and ORF2 by calculating $\omega_1 = d_{NN}/d_{SN}$ and $\omega_2 = d_{NN}/d_{NS}$. Here, $d_{NN}$ is the rate of accumulation of mutations with nonsynonymous effects in both ORFS, and $d_{NS}$ is the rate of accumulation of nonsynonymous mutations in the first ORF but synonymous mutations in the second ORF. To calculate $d_{XX}$, the method first classifies nucleotide substitutions in four categories depending on their coding context ($NN$, $NS$, $SN$, $SS$). For example, a nucleotide might be classified as $1/3\ NN$ and $2/3\ NS$ if one-third of potential mutations at the site cause nonsynonymous changes in both ORFs and two-thirds of potential mutations at the site cause nonsynonymous changes in ORF1 but synonymous changes in ORF2. Then, the Wei-Zhang method estimates $L$ by counting the expected
number of nucleotides under each one of the categories, and averages those counts between both sequences so that each site in the sequence is assigned to either $L_{NN}$, $L_{NS}$, $L_{SN}$, or $L_{SS}$. Next, they count the frequency of each category on the sites where the sequences differ, obtaining $M_{NN}$, $M_{NS}$, $M_{SN}$, and $M_{SS}$, and estimates the proportion of observed ($M$) versus the potential ($L$) substitutions at each one of the four categories (i.e., $P_{NN} = M_{NN}/L_{NN}$ is the proportion of $NN$ sites). Finally, they use the Jukes-Cantor [41] formula (Equation 1.4), to estimate distance between the two sequences as $d_{NN} = \frac{3}{4} \ln 1 - \frac{4P_{NN}}{3}$.

Later, Nelson and collaborators created OLGenie, a program to estimate selection in OvRFs using a modified version of the Wei-Zhang method, where, instead of considering all mutational pathways, they only count single nucleotide differences [81]. To assess the performance of OLGenie, the authors tested for purifying selection in different regions of HIV-1, and corroborated findings of purifying selection demonstrated under laboratory experiments. Ultimately, OLGenie proved to be a useful method to estimate $dN/dS$ on sequences with a relatively low level of divergence, but lose power for pairwise distances greater than 0.1.

Overall, estimating selection in OvRFs has fascinated scientists for decades, leading to innovative approaches that address the complexity inherent in calculating the effects of mutations on multiple ORFs from different perspectives.

### 1.3.1 Methods for the detection of OvRFs

To detect overlapping genes, previous approaches have exploited the following biological characteristics: (1) the function of proteins encoded by an OvRF are not necessarily well defined; (2) OvRFs might be subject to different substitution patterns, and in particular, they tend to accumulate synonymous mutations at a lower rate than their non-overlapping counterparts; and (3) atypical codon biases can be observed in *de novo* genes created by overprinting — where a new protein is translated from a different reading frame on top of an existing coding sequence [5].

Initially, Firth and Brown [82] developed a statistical method called MLOGD (Maximum Likelihood Overlapping Gene Detector) to detect overlapping coding sequences. MLOGD estimates
the probability of obtaining the observed pattern of mutations across a sequence alignment under two models: 1) a null model, where known locations for the coding sequence (CDS) are specified and 2) an alternative model, where putative proteins are encoded on top of the known CDS. Given two aligned sequences, $S_1$ and $S_2$ of length $N$, MLOGD estimates the probability that $S_1$ mutates to $S_2$ after time $t$ under model $M$ (which can be null or alternative) by:

$$\log P(S_1 \rightarrow S_2; t, M) = \sum_{k=1}^{N_{\text{nt}}} \log P(N^k_1 \rightarrow N^k_2; t, m^k) \quad (1.7)$$

where $N^k_1$ and $N^k_2$ are the nucleotides in $S_1$ and $S_2$ at the $k$th alignment position; $P(N^k_1 \rightarrow N^k_2; t, m^k)$ is calculated using nucleotide, codon and amino acid substitution matrices (as described in [82]); and $m^k$ indicates whether the nucleotide at position $k$ is involved in 0, 1, or 2 CDSs according to $M$. Evidence of overlap is established when the likelihood of the alternative model is higher than the likelihood of the null model.

Later, Sabath and Graur [83] tested the hypothesis that overlapping genes can be detected by measuring purifying selection across the nucleotide sequences. In their method the authors used a maximum-likelihood framework to fit a Markov model of codon substitution to data from two aligned overlapping sequences. This method was used to detect functional overlapping genes and demonstrated to have higher sensitivity than the one presented by Firth and Brown.

Since virus coding-sequences tend to be only a thousand codons long or less, there is often insufficient information to characterize codon transition patterns accurately. With this in mind, Firth [84, 85] developed an algorithm to detect OvRFs based on statistically significant reductions in synonymous substitution rates called Synplot2. First, the algorithm establishes a null model of neutral evolution at synonymous sites using a Kimura nucleotide substitution matrix, accounting for differences between transition and transversion rates [42] (see section 1.2.1). Then, Synplot2 sets the expected diversity at synonymous sites as the total number of synonymous substitutions across the entire alignment, and the observed diversity is calculated as integers (1, 2, or 3) between sequential codon positions from a pair of sequences. The sequence pairs are selected following a
1.3. Estimating evolution in overlapping reading frames

phylogeny constructed from the full alignment. Finally, Synplot2 estimates the probability that any reduction in synonymous-site variability in the window would be as great as observed if the null model (sites evolving neutrally) was true. Overlapping regions are predicted to be located on sites where the expected versus observed number of synonymous mutations are significantly different (statistical significance assessed Bonferroni-like correction for multiple testing).

Following a similar approach, Sealfon and colleagues [86] devised FRESCo, a framework aimed at identifying regions subject to synonymous constraints. Initially, their approach involves fitting a nucleotide substitution model using a maximum-likelihood method to an alignment. The resulting model parameters are then employed to estimate branch lengths of the phylogeny relating all sequences, as well as the substitution parameters of the codon model. Subsequently, FRESCo employs a sliding window technique across the nucleotide alignment to estimate a null model (that accounts only for nonsynonymous substitution rates $dN$ by constraining the synonymous substitutions rates $dS$ to 1), and an alternative model (estimates both $dN$ and $dS$ for each site). Ultimately, a likelihood ratio test is conducted to compare the two models, identifying the regions with excess synonymous constraints.

A more straightforward method was proposed by Schulb and colleagues [87], positing that ORFs encoding functional proteins tend to be larger than those arising from random chance alone. As a consequence, significantly larger ORFs are deemed indicative of predicted overlapping genes. Applying this methodology to all available linear RNA genomes in GenBank yielded evidence of 29 previously undocumented OvRFs, with 17 of them being antisense ORFs.

Interestingly, Pavesi and colleagues [24] proposed to identify genes created de novo by overprinting by detecting statistically significant deviations in the codon usage of a pair of overlapping genes. The idea behind this method is that ancestral genes should have patterns of codon usage (i.e. which synonymous codon(s) is preferred to encode each amino acid) closer to that of the rest of the genome. Their methodology involves quantifying the occurrences of synonymous codons, which encode the same amino acid residue, in both genes. Then, the observed differences in codon patterns are assessed in relation to the remaining genome in order to identify which of the two
Chapter 1

genes is the ancestral one.

1.4 Comparative methods for sequence analysis

Comparison lies at the heart of biology, as we seek to uncover patterns of similarity in individuals, species or sequencing runs, in order to inform our understanding of evolving organisms. In the context of viruses, we often compare sequences to reveal evolutionary relationships, predict genes new genes, characterize protein functions, or find vaccine targets. Various tools have been developed to compare sequences at different levels of similarity. For instance, nucleotide sequences are often used for closely related organisms. As diversity increases, the action of purifying selection, and genomic saturation caused by multiple substitutions at the same site, hinder the comparative task [88]. Under larger evolutionary time spans, amino acid sequences and tertiary protein structures are analyzed to to reduce these effects.

To study overlapping reading frames (OvRFs) in viruses, I used different methods of sequence comparison. For example, to assess the distribution of OvRFs across genomes in various virus families, I used alignment-free methods, that enabled the comparisons between distantly related sequences. To evaluate selection pressures in OvRFs, I used nucleotide alignments to create phylogenetic trees and estimate the rate of accumulation of synonymous and nonsynonymous substitutions in protein-coding sequences with and without overlaps. In the following sections I will outline the fundamental concepts of alignment and alignment-free methods for sequence comparison.

1.4.1 Aligning sequence data

Sequence alignments are hypotheses about how specific nucleotide or amino acid residues descend from the same residue in their common ancestor. The foundational algorithm for sequence comparison was developed in 1970 by Needleman and Wunsch (NW) [89]. The NW algorithm employs dynamic programming: a problem-solving approach that breaks down complex problems
Comparative methods for sequence analysis

into smaller sub-problems and solves each one optimally and independently [90]. The objective is to find the best alignment, considering either the lowest edit distance (the minimum number of operations required to transform one string into the other) or the highest alignment score (that indicates the quality of the alignment) [91].

To compute an alignment score, we assign costs to matches (identical characters), mismatches (different characters), and gaps (insertions or deletions). Positive costs are typically used to reward algorithms for aligning matching residues, while negative scores serve as penalties. The NW algorithm relies on filling out a score matrix (where columns are labeled with sequence 1, and rows are labeled with sequence 2) with comparisons of all sites in both sequences, and traversing it from the lower-right corner to find the optimal path that minimizes the alignment score. Under this method, we create global alignments, which represent optimal alignments spanning from the first to the last nucleotide residue in both sequences. However, there are instances where we need to identify local alignments to find regions of similarity between sequences with significantly different lengths i.e., incomplete sequences. The primary algorithm for locally aligning sequences was developed by Smith and Waterman (SW) [92], who modified the NW algorithm by relaxing the constraint of traversing the score matrix from the lower-right to the upper-left. In the SW algorithm, matrix scores are restricted to values greater than or equal to 0, and alignment paths are recorded only if the starting cell has an alignment score greater than 0 [91].

The fundamental concepts behind the NW and SW algorithms have been extended by multiple alignment-based tools designed to assess sequence homology in more complex scenarios. For instance, evolutionary analyses often require comparisons of more than two sequences, making multiple sequence alignments (MSA) indispensable. In 1994, Thomson and colleges developed ClustalW [93], a progressive alignment algorithm that builds up an MSA by calculating genetic distances between pairs of sequences (pairwise alignment). Using the distance matrix, CLUSTALW employs clustering methods like Neighbour-Joining [94] to create a tree, with branching patterns serving as a guide for aligning sequences based on their degree of relatedness. Over time, numerous novel approaches have been developed to enhance the speed of MSA programs. For example,
the MAFFT aligner [95] utilizes the Fast Fourier Transform (FTT) method to rapidly identify homologous segments between two sequences, forming the basis for the alignment. A vast number of programs for MSA have been developed over time, with Wikipedia listing over 50 tools as of the writing of this thesis.

Alignment methods are also crucial for tools used to search for sequence similarity. One well-known example is the Basic Local Alignment Search Tool (BLAST) [96], developed in 1990. The BLAST algorithm divides the query sequence that we wish to align into subsequences of length $k$, known as $k$-mers. These $k$-mers are stored in a hash table as a key/value pair, which is then compared against other hash tables generated from the target sequences. Exact matches between the $k$-mers in the target and query sequences are called hits. If two hits are detected between the sequences, their positions are used to initiate a candidate local alignment, which is extended in both directions with the SW algorithm. If the alignment score falls below a specific threshold, the alignment is halted, and the candidate is discarded. Similarly, the FASTA method [97], developed in 1987 for quickly generating local alignments, uses shorter $k$-mers than BLAST. Instead of initializing the alignment at a pair of hits, FASTA identifies regions in both sequences that have a high density of hits and retains only the top 10 candidates [91].

So far, we have discussed methods that focus on comparing individuals bases or amino acids that occupy corresponding positions in multiple sequences. This assumption is based on the idea that homologous sequences consist of linearly arranged and relatively conserved stretches. However, certain evolutionary scenarios require alternative approaches that do not rely on comparing residues at specific locations. The study of viruses, in particular, involves complex processes such as high mutation rates, recombination events, horizontal gene transfers, gene duplications, and gene gain, which complicate direct residue comparisons [98]. Consequently, alignment-free methods have been developed to capture the intricate relationships within genome evolution and address broader sequence analysis challenges.
1.4.2 Alignment-free methods

Not all methods for sequence comparison require aligned sequences. When a long stretch of shared residues between sequences is not expected, or when there is a need to analyze large amounts of data or extremely long genomes, alignment-free methods may be preferred. Unlike traditional alignment methods that rely on dynamic programming, alignment-free methods offer several advantages. For instance, they are resistant to events where different regions of the genome have different evolutionary histories, making them applicable to the study of sequences that are product of recombination [99]. Additionally, alignment-free methods provide rough estimates of sequence similarity for large datasets in a less resource-consuming fashion [98]. Furthermore, these methods are particularly useful in the detection of convergent evolution, as they can be applied to sequences without the expectation of a shared common ancestor [100, 101].

Over time, multiple innovative approaches for sequence comparison have emerged, offering a diverse array of options when assessing sequence variation. For example, a straightforward method involves estimating the differences in the proportion of G’s and C’s (GC-content) between sequences. More sophisticated approaches propose compression algorithms and comparing the shortest representation of the sequences [102, 103]. Similarly, one can evaluate estimates of Shannon entropy, which quantifies the uncertainty associated with finding a specific string within a sequence [104]. Among alignment-free methods, one extensively studied approach involves comparing the occurrences of words of length \( k \) (\( k \)-mers) between pairs of sequences. In general, all \( k \)-mer count methods require the following steps [98] (see Figure 1.8):

1. First, the sequences are divided into a collection of words of length \( k \). For example, the sequence \( S_1 = ATGTGTG \) can be divided into a collection of 3-mers \( W_{S_1} = \{ATG, TGT, GTG, TGT, GTG\} \), and the sequence \( S_2 = CATGTG \) can be divided into \( W_{S_2} = \{CAT, ATG, TGT, GTG\} \).

2. Next, the sequences are transformed into arrays of numbers that represent the frequency of each word within the sequence. Since some words may be present only in one sequence,
Figure 1.8: Using k-mer counts to assess sequence similarity. Initially, we divide each sequence into substrings of length $k = 3$. We then calculate each k-mer frequency, and estimate the distance between sequence pairs to obtain a distance matrix relating the entire dataset. Figure adapted from [98].

the set of unique words between $S_1$ and $S_2$ is $W_{S_1,S_2} = \{\text{ATG, TGT, GTG, CAT}\}$, and the vectors of word counts are $c_{S_1} = (1, 2, 2, 0)$ and $c_{S_2} = (1, 1, 1, 1)$.

3. Finally, the dissimilarity between the sequences is quantified using a distance function based on these word count vectors.

A wide variety of distance functions have been employed to compare vectors of k-mer counts.
One of the most commonly used functions is the Euclidean distance, which is based on Euclid’s statement from over two thousand years ago that the shortest distance between two points is a line. According to Equation 1.8, we can calculate the distance for our previous example as 

\[ d(S_1, S_2) = \sqrt{(1-1)^2 + (2-1)^2 + (2-1)^2 + (0-1)^2} = \sqrt{3} = 1.73. \]

\[ Euclidean(f_{S_1}, f_{S_2}) = \sqrt{\sum_{w \in W_{S_1S_2}} (c_{S_1}(w) - c_{S_2}(w))^2} \]  

(1.8)

In contrast, intersection methods require counts of words common to both sequences. These methods use a \( \min \) function to determine the overlap between the distribution of words from \( S_1 \) and \( S_2 \) by recording how many of each \( k \)-mers are present in both sequences (Equation 1.9). Examples of intersection methods include the Czekanowski, Kulczynski, and Tani-moto distances [105], which have demonstrated high performance in measuring distances between amino acid sequences in low and high sequence identity datasets [106].

\[ Intersection(f_{S_1}, f_{S_2}) = \sum_{w \in W_{S_1S_2}} \frac{2\min(c_{S_1}(w), c_{S_2}(w))}{c_{S_1}(w) + c_{S_2}(w)} \]  

(1.9)

Other statistical families, such as \( D2 \), match/mismatch (that includes the Hamming distance), and Markov metrics, are also employed for sequence comparison. These methods have been extensively reviewed in recent years using test datasets of protein sequences, genes, regulatory elements, and full genomes [98, 106-108].

Overall, sequence comparison remains a fundamental tool in biology, being commonly used to unravel evolutionary patterns in biological sequences. By incorporating diverse mathematical and computational approaches such as Markov chains or information theory, humans have created more efficient models that include a wider range of biological properties, allowing us to extract valuable insights from nucleotide and amino acid sequences under more complex scenarios. Using some of these tools, I was able to dig into the hidden signals contained in overlapping reading frames in virus genomes, and as the field of sequence comparison progresses, it will undoubtedly inspire new advancements and approaches in the ongoing quest to understand virus evolution.
1.5 Networks in a nutshell

To reduce the complexity of a system, we have created abstractions of reality that allow us to process and evaluate the relations between its structural components; networks are precisely that. In the most basic form, a network is a collection of points joined together in pairs by lines that we refer to as nodes and edges respectively (Figure 1.9). Based on these abstractions, we can investigate how different parts interact and affect the behaviour of the entire system by identifying patterns of connections. However, as stated by Alfred Korzybski in 1931, ‘the map is not the territory,’ and during the abstraction process a lot of information is lost. To better represent the system, we can incorporate additional attributes to the networks that provide more information about the characteristics of the connections (i.e., labeling nodes or edges, or discerning edges by the strength of the interaction) [109].

The connection patterns in a network are mathematically represented by an adjacency matrix $A$, that in a network with $n$ nodes is defined by a square matrix of $n \times n$, with elements $A_{ij} \in \{0, 1\}$, where 1 signifies an edge between nodes, while a 0 denotes the absence of a connection (Figure 1.9 top panel). In the simpler definition of a network, adjacency matrices have a diagonal with elements equal to zero (no self-edges, meaning that a node cannot be connected to itself), and the matrix is symmetric, meaning that if there is an edge between $i$ and $j$, there is necessarily an edge between $j$ and $i$.

To incorporate further information about a system, weighted networks assign real non-negative numbers to the elements of the adjacency matrix that denote the strength of connections between nodes. Moreover, directed networks include details about the order of connections, with edges pointing from one node to another. The adjacency matrices of directed networks are asymmetric, since the existence of an edge from $i$ to $j$ does not necessarily imply an edge from $j$ to $i$ [109] (Figure 1.9 bottom panel).

Based on the quantitative representation of networks, the mathematical and computational tools for analyzing, modeling, and understanding networks can be applied to any system as long as it
1.5. Networks in a nutshell

![Diagram of a basic network and an adjacency matrix]

**Figure 1.9:** **Graphical and mathematical representation of a network.** *(top)* In the most basic form of a network, nodes are connected by edges, represented in an adjacency matrix as 1s. *(bottom)* Edges can have weights assigned to them to indicate differentiate between the strengths of the interactions. In directed networks, arrows indicate the direction of the edges that go *from* one node to another. In this example, the width of the edges corresponds to the weight assigned in the adjacency matrix. For clarity purposes, the color of the edges corresponds to the color of the cell in the adjacency matrix. Figure adapted from [109].

can be represented by an adjacency matrix. Biological networks, in particular, are used to describe patterns of interaction of multiple systems, for example:

**Metabolic networks:** The pathways by which molecules and their intermediates are produced or broken down by enzymes can be represented by networks. The nodes are the metabolites and the edges are the chemical reactions that use or convert them into useful molecules for subsequent processes.

**Protein interaction networks:** Protein-protein interactions occur via diverse mechanisms, such
as direct physical binding, protein conformational changes, post-translational modifications, and recognition domains. In protein interaction networks, proteins that participate in protein complexes are represented as nodes while their interactions are depicted as edges.

**Genetic regulatory networks:** The mechanisms that govern gene expression can be simplified into networks of nodes that represent genes and edges that represent interactions. In particular, a connection is formed if a genes promotes or inhibits the production of other proteins. In turn, these proteins can also function as transcription factors for additional genes.

**Networks of neurons:** Neurons can be modeled as abstract entities that receive inputs, combine them, and generate outputs that are transmitted to other neurons. Networks capture this flow of information with neurons represented as nodes and signals as edges.

**Ecological networks:** Interactions among species in an ecosystem can be represented by networks where nodes are species and edges symbolise predator-prey interactions, host-parasite infections, or migration patterns, among others.

The graphical representation of networks has proven to be a valuable tool for identifying patterns by visually inspecting the system. While these visual insights are helpful, a wide range of network statistics have been developed to facilitate our understanding of the nature and function of a system, by assessing the shape and structural features of the network that describes it \[109\]. Thus, network statistics are used to quantify the importance of a node (centrality), the type of divisions or communities formed (grouping patterns), the characteristic and prevalence of certain connections (transitivity, reciprocity, structural balance), or the commonalities between two nodes. Similarity statistics include, for example, *regular equivalence* — which describes similarities between node neighbors, or *homophily* — which measures the tendency of a node to associate to other nodes with similar characteristics. Since the toolkit to study networks is quite extensive and complex, I will only briefly review the specific metrics relevant to my research, where I aimed to represent virus genomes as networks in the search for patterns that could predict the presence and distribution of overlapping reading frames (OvRFs).
1.5. Networks in a nutshell

Figure 1.10: **Network features of clusters of homologous proteins with overlapping reading frames.** To analyse the distribution of OvRFs in the context of adjacency graphs, we calculated metrics such as degree size, number of triangles, transitivity and Eigenvector centrality as edge-level attributes, and used a stepwise model selection procedure to determine which combination of attributes was best supported by the data as predictor variable of gene overlaps.

Our approach to create **adjacency graphs** — which are our method for representing genome networks with overlapping genes, is further detailed in Chapter[^2]. In a nutshell, adjacency graphs are directed weighted networks composed by nodes, that represent **clusters of homologous proteins** from genomes in the same virus family — with sizes proportional to the number of proteins assigned to it, and edges, that represent **proximity in the genome** (**Figure 1.10**). Using information on the location of the coding sequences (CDSs) within the genome, the adjacency graph’s edges are used to connect nodes with CDSs that are **adjacent** (one CDS is located right after the other), represented by gray arrows, or **overlapping** (some nucleotides in the genome are used for both CDSs), represented by blue arrows. To extract information from our adjacency graphs, we were interested in detecting network attributes that could be associated with the presence of an overlap. Hence, we evaluated the following network statistics (**Figure 1.10**):
**Degree centrality:** Also known as *edge degree*, this metric is used to quantify the **influence of a node in the network (importance)**, based on the number of edges connected to it (more connection, higher importance). In our adjacency graphs of virus genomes, where a network is created by analyzing the organization of multiple genomes, the number of connections of a given node represents the number of proteins in other clusters that are adjacent to proteins in the current node. A degree centrality of 1 would indicate that all the proteins in the node are always adjacent to the same CDS. A node with a high degree centrality indicates that proteins in that cluster are adjacent to two or more different CDSs that were classified in a set of multiple clusters.

**Eigenvector centrality:** Rather than measuring the number of connections of a node, the eigenvector centrality measures the importance of a node based on its connections with other important nodes. Higher scoring nodes for this metric in our adjacency graphs would be clusters connected to other multiple clusters that also form themselves multiple connections.

**Transitivity:** A relation between A and C is transitive if node A is connected to node B, and B is connected to C. Put simply, ‘the friend of my friend is also my friend’. In particular, *perfect transitivity* occurs if the three nodes form a triangle (A and B are also connected with an edge). Otherwise, it is *partial transitivity*. The level of transitivity can be quantified using the *clustering coefficient*, which is the proportion of paths of length two that are closed in the network. In our adjacency graphs, a triangle is formed when gene C is sometimes inserted between genes A and B, representing clusters of genes that are present in some of the genomes of the family, but not all (Figure 1.11).

Although the information presented in this section only scratches the surface of the available network analysis methods [109], I hope it has offered enough context to understand the use of networks in the study of overlapping reading frames in virus genomes. Overall, networks encompass an extremely rich and diverse set of tools and techniques; nevertheless, this overview serves as a concise foundation to highlight the relevance and potential of networks in studying complex
1.6. Rationale and Significance

Overlapping Reading Frames (OvRFs) are special features of the genome that can be found in a wide range of taxonomic groups, where the same nucleotide sequence encodes more than one protein. In virus genomes, OvRFs have been documented in all Baltimore classes, where they are used to encode proteins more compactly and to regulate transcription. In addition, OvRFs might be a source of gene novelty, facilitating the creation of new open reading frames (ORF) within the transcriptional context of existing ones. To investigate the emergence and evolution of OvRFs in virus genomes, I characterized their distribution in virus genomes, and developed an alignment-free method to cluster coding sequences (CDS) of reference genomes on virus families from different Baltimore classes (Chapter 2). Then, I used these clusters of homology to represent virus genomes as networks, where edges are associated with consecutively encoded proteins, some of which overlap.

Additionally, even though the study of OvRFs has fascinated scientists since the beginning of systems.

Figure 1.11: Triangle formation in adjacency graphs. Genes that are present in some genomes but absent in others within the same virus family are represented as triangles in our genome networks. Genomes of species in the hypothetical family X, comprise genes A, B, and, in some cases, gene C. Since gene C is present in some species (with genome 2), but not in others (with genome 1), the adjacency graph of family X has a triangle formed by $A \leftrightarrow B \leftrightarrow C$. 

Genomes in family X

Triangle formation

Genome organization

Adjacency graph representing genomes in family X
sequencing techniques, the processes over which they evolve remain poorly understood. Estimates of the rates of synonymous (\(dN\)) and non-synonymous substitutions (\(dN\)) quantify the selection intensities acting on the coding sequence. However, these estimates can be skewed by OvRFs, since a mutation that is synonymous in one reading frame may be non-synonymous in another, and vice versa. To decipher the effects of OvRFs in coding sequences, I developed HexSE: a simulation model of nucleotide sequence evolution along a phylogeny that tracks the substitution rates at every nucleotide site (Chapter 3). Overall, my objective was to characterize the amount of variation in selection that can be explained by OvRFs. Thus, I used HexSE to run multiple simulations of evolution in the hepatitis B virus (HBV) by independently modifying the coding contexts and the parameters that shape the selection intensities in the different ORFs. Finally, I used a change detection method to the resulting alignments to identify overlapping regions from variation in \(dS\) in a reference ORF (Chapter 4).

Despite the detection of OvRFs across a wide diversity of genomes, the selective pressures acting on these segments and their overall impact on protein synthesis continue to raise intriguing questions that require further exploration. Gaining insights into how different processes affect gene overlap relies on the development of new methods that allow us to fit evolutionary models to OvRFs. This research is an approach to understanding some of the fundamental characteristics of OvRFs by creating novel computational tools for the analysis of overlapping patterns in virus genomes. I envision that the ideas proposed in this thesis will contribute to the comprehension of OvRFs, and with them, the origin of new genes, the evolution of virus genomes, and, consequently, the diversity of genes themselves.
Bibliography


[7] Staffan Normark, Sven Bergström, Thomas Edlund, Thomas Grundström, Bengtake Jaurin,


Chapter 2

Investigating the global distribution of Overlapping Reading Frames in virus genomes

To generate a global snapshot of the prevalence of overlapping reading frames (OvRFs) in viruses, I analyzed 12,609 reference genomes from the NCBI virus database. I found that as the genome length increases, the number of OvRFs also increases, but OvRFs tend to be shorter in longer genomes. Additionally, I discovered distinct patterns of different frameshifts in OvRFs. Furthermore, I retrieved \( n = 8,586 \) protein-coding sequences from \( n = 1,224 \) reference genomes and employed an alignment-free method to cluster these sequences within virus families. I used these clusters to develop a new network-based representation of the distribution of OvRFs, which provides a means of visualizing and analyzing these genome features for each virus family.

This chapter encompasses the publication that present the results of this study — published in PLoSPathogens [1] (section 2.1), along with a detailed description of our clustering method — published independently in Current Protocols [2] (section 2.2).
2.1 Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

Laura Muñoz-Baena\textsuperscript{1} and Art F. Y. Poon\textsuperscript{1,2*}

\textsuperscript{1} Department of Microbiology and Immunology, Western University, London, ON, Canada.
\textsuperscript{2} Department of Pathology and Laboratory Medicine, Western University, London, ON, Canada.

Gene overlap occurs when two or more genes are encoded by the same nucleotides. This phenomenon is found in all taxonomic domains, but is particularly common in viruses, where it may increase the information content of compact genomes or influence the creation of new genes. Here we report a global comparative study of overlapping open reading frames (OvRFs) of 12,609 virus reference genomes in the NCBI database. We retrieved metadata associated with all annotated open reading frames (ORFs) in each genome record to calculate the number, length, and frameshift of OvRFs. Our results show that while the number of OvRFs increases with genome length, they tend to be shorter in longer genomes. The majority of overlaps involve $+2$ frameshifts, predominantly found in dsDNA viruses. Antisense overlaps in which one of the ORFs was encoded in the same frame on the opposite strand ($-0$) tend to be longer. Next, we develop a new graph-based representation of the distribution of overlaps among the ORFs of genomes in a given virus family. In the absence of an unambiguous partition of ORFs by homology at this taxonomic level, we used an alignment-free k-mer based approach to cluster protein coding sequences by similarity. We connect these clusters with two types of directed edges to indicate (1) that constituent ORFs are adjacent in one or more genomes, and (2) that these ORFs overlap. These adjacency graphs not only provide a natural visualization scheme, but also a novel statistical framework for analyzing the effects of gene- and genome-level attributes on the frequencies of overlaps.
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

2.1.1 Author summary

Gene overlap occurs when the same part of a genome encodes two or more genes. This phenomenon is found in all biological domains of life, but it is particularly common in viruses, where it may play a role in making viral genomes more compact. To understand the prevalence of overlapping genes in viruses, we analyzed over 12,000 genomes of every known type of virus for which this genetic information is available. Although overlaps are more abundant in viruses with larger genomes, for instance, they are also significantly shorter. Overlaps in which one of the genes is read in the opposite direction (−0 overlaps) tend to be longer, which may be an emergent property of the universal genetic code. We developed a new computational method to analyze and visualize the distribution of overlaps among genomes belonging to a group (family) of viruses as a network. This approach enabled us to identify distinct patterns in the organization of genomes within virus families; for example, gene overlap in the coronavirus family tends to involve non-essential genes outside of the "core" of the network of genes.

2.1.2 Introduction

Viruses are an enormous part of the natural world, representing the majority of entities in our planet that undergo organic evolution. For instance, a recent study estimated the existence of over $10^{31}$ bacterial viruses, i.e., bacteriophage [3], which is only a fraction of viral diversity. A particularly noteworthy feature of virus genomes is the ubiquitous presence of overlapping reading frames (OvRFs): portions of the genome where the same nucleotide sequence encodes more than one protein. OvRFs have been documented in all seven Baltimore classes — categories of viruses by genetic material, including double-stranded DNA (dsDNA) and positive single-stranded RNA (ssRNA+) viruses [4]. A number of hypothetical mechanisms have been proposed to explain this abundance of OvRFs in viruses. First, the prevalence of overlapping genes is hypothesized to be related to genome size. Given that genomes of many viruses are physically constrained by capsid size [5], OvRFs provide a mechanism for encoding more information in a given genome length.
Another model proposes that OvRFs could be also used by viruses as a mechanism to accommodate high mutation rates by amplifying the effect size of deleterious mutations (antiredundancy), such that purifying selection removes these mutations more efficiently from the population \[6, 7\]. In addition, OvRFs have been suggested to be a symptom of gene origination, where a new open reading frame (ORF) may arise within the transcriptional context of an existing ORF \[8\]. Recent studies have produced comparative evidence that these \textit{de novo} genes will not initially have a well-established function, but will be able to acquire it over time \[9\].

Previously, Schlub and Holmes \[10\] analyzed overlapping genes in 7,450 reference virus genomes in the NCBI viral genomes database \[11\] to confirm that the number of OvRFs per genome, as well as the number of bases within OvRFs, increases significantly with genome length. In contrast with previous research, however, they also reported that this association was more pronounced in DNA viruses than RNA viruses, and in double-stranded versus single-stranded genomes. Like related work in the literature \[5, 7\], their comparative study employed quantities like the number of OvRFs or total overlap length (\textit{i.e.}, the number of nucleotides in overlapping regions) that do not distinguish one ORF from another. In other words, these are summary statistics where the entire genome is the unit of observation.

Our objective is to incorporate gene homology into characterizing the distribution of OvRFs in virus genomes, with the intent of gaining a more detailed understanding of this phenomenon. This comparative analysis relies on accurate annotation of ORFs in reference genomes. Gene annotation is an increasingly challenging problem, however. For instance, the number of reference virus genomes in the NCBI RefSeq database increased more than five-fold between 2000 and 2015, driven in part by the increasing use of next-generation sequencing platforms \[11\]. Many putative ORFs in newly discovered virus genomes have no recognizable homologs in protein sequence databases \[12\]. Furthermore, ORFs in reference genomes are not always annotated with consistent labels, or are assigned the wrong label altogether. Misannotations are sufficiently prevalent that there are multiple collaborations to create and maintain databases of specific categories of genomes with manually-curated gene annotations \[13, 14\]. To develop a global picture of OvRF
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

diversity across viruses at gene-level resolution, we need an automated method to efficiently label homologous ORFs for related virus genomes.

Here we report a comparative analysis of OvRFs in 12,609 virus reference genomes in the NCBI virus database. First we use conventional genome-level summary statistics to revisit fundamental questions about OvRFs in viruses, e.g., do overlaps tend to occur between certain reading frames in viruses?. Next, we develop and employ an alignment-free method for clustering ORFs by sequence homology within a given virus family. This enables us to generate graphs where nodes represent clusters of homologous ORFs. These nodes are connected by two types of edges that indicate the adjacency of ORFs in genomes and the presence of overlaps, respectively. This graph-based approach not only provides an inherent visualization method for the diversity of OvRFs among different virus families, but also enables us to access the rich library of network statistics [15] to characterize the abundance and distribution of OvRFs in virus families.

2.1.3 Methods

Data collection and processing

First, we downloaded the accession list of all available virus genomes from the NCBI Viral Genomes Resource [11] (https://www.ncbi.nlm.nih.gov/genome/viruses/ accessed on 2020-09-28), a community-based effort to curate references from the growing number of virus genomes in the NCBI Genbank database. This tab-separated file comprised 247,941 rows and six columns labeled as ‘representative’, ‘neighbor’, ‘host’, ‘taxonomy’ and ‘segment name’. Representative genomes are used to denote significant intra-specific variation that cannot be adequately captured by a single reference genome, whereas neighbors are additional validated and complete or nearly-complete genomes for a given species [11]. We used only a single representative genome for each species as sufficient information for our purposes. We used a Python script to retrieve additional metadata (genome length, number of proteins, topology and molecule type) associated with each reference genome using the NCBI Entrez API [16][17].
The same script was used to generate a tabular dataset recording the genome accession number, product, strand, coordinates and start codon position for every coding sequence (CDS). A second Python script was used to identify putative overlapping open reading frames (OvRFs) from the genome coordinates of all CDSs by accession number. Every OvRF was recorded by its location, length in nucleotides, and shift (if applicable) relative to the upstream reading frame. Following convention \[18\], overlaps between reading frames on the same strand were recorded as +0, +1 and +2 when shifted by zero, one and two nucleotides, respectively. Similarly, overlaps on opposing strands were recorded as −0, −1 and −2 (see 2.6 Fig). Next, we extracted Baltimore classifications for virus families from the Swiss-Prot virus annotation resource (https://viralzone.expasy.org [19]).

**Clustering protein data by family**

To analyze the distribution of overlapping open reading frames (OvRFs) in different virus families, we retrieved the protein sequences for all CDSs of all reference genomes of each family from the NCBI virus database. Our objective was to identify homology among protein coding sequences that may be highly divergent and inconsistently annotated at the family level of virus diversity. We also needed to be able to accommodate gene duplication and divergence in DNA viruses, as well as unique ORFs with no homologs in other genomes (**i.e.,** accessory genes, ORFans [20]). As a result, we decided to use an alignment-free method to compute \(k\)-mer-based similarity scores between every pair of ORFs within a virus family (2.8 Fig). We used Python to map each protein sequence to a dictionary of \(k\)-mer counts for \(k = \{1, 2, 3\}\) as a compact representation of the sparse feature vector. Let \(W(s)\) represent the set of all \(k\)-mers (words) in a sequence \(s\), and let \(f(s, w)\) represent the frequency of \(k\)-mer \(w\) in \(s\). Using these quantities, we calculated the Bray-Curtis distance [21] between sequences \(s\) and \(t\):

\[
k(s, t) = 1 - \frac{\sum_{w(s) \cap w(t)} 2 \min(f(s, w), f(t, w))}{\sum_{w(s)} f(s, w) + \sum_{w(t)} f(t, w)}
\]

This \(k\)-mer distance performed relatively well at the task of protein classification in a recent benchmarking study of alignment-free methods [22], where it was implemented as the intersection dis-
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

tance in the AFKS toolkit [23]. Intuitively, this measure reflects the overlap of two frequency distributions, normalized by the total area of each distribution. The resulting distance matrix was used as input for the t-distributed stochastic neighbor embedding (t-SNE) method implemented in the R package Rtsne [24]. This dimensionality reduction method embeds the data points into a lower-dimensional space in such a way that the pairwise distances are preserved as much as possible.

Next, we generated a new distance matrix from the coordinates of the embedded points and then used hierarchical clustering using the R function hclust with Ward’s criterion [25] (‘ward.D2’). Combining dimensionality reduction and clustering methods is frequently used in combination because distance measures have unexpected properties in high dimensional feature spaces [26].

Finally, we used the R function cutree to extract clusters by applying a height cutoff to the dendrogram produced by hclust. Increasing the number of clusters by lowering this cutoff accommodates more ORFans. Conversely, raising the cutoff reduces the number of false positive clusters (ORFs that should not be classified as ORFans). To determine an optimal cutoff for a given virus family, we selected the height that balances two quantities. Let \( f(i, j) \) be the number of ORFs assigned to cluster \( j \in \{1, \ldots, K\} \) in genome \( i \in \{1, \ldots, N\} \). First, we calculated the mean proportion of ORFs with unique cluster assignments per genome:

\[
E_1 = \frac{1}{N} \sum_{i=1}^{N} \left( \frac{\sum_{j=1}^{K} I(f(i, j) = 1)}{\sum_{j=1}^{K} I(f(i, j) > 0)} \right)
\]

where \( I(x) \) is an indicator function that assumes a value of 1 if \( x \) is true, and 0 otherwise.

Second, we calculated the mean frequency of a cluster assignment across genomes:

\[
E_2 = \frac{1}{K} \sum_{j=1}^{K} \left( \frac{1}{N} \sum_{i=1}^{N} f(i, j) \right)
\]

\( E_1 \) increases with an increasing number of clusters, whereas \( E_2 \) declines because ORFs are distributed across more clusters. Thus, we passed the squared difference \( (E_1 - E_2)^2 \) as an objective function for R function optimize to locate the optimal cutoff for each virus family.
Data visualization

Using the OvRF coordinate data from the preceding analysis, we used a Python script to generate adjacency graphs as node and edge lists for each virus family. Every cluster of homologous ORFs is represented by a node. Each node has two ‘connectors’ representing the 5’ and 3’ ends of the corresponding ORFs in the cluster. Such coordinates are drawn from the annotated locations for each CDS. For a given genome sequence, all ORFs are sorted by the nucleotide coordinates of their 5’ and 3’ ends in increasing order. When genes are encoded on the complementary strand their coordinates keep an ascendant order, and the ORF is tagged with a $-1$ to indicate the difference in orientation so they can be processed using the same procedure. Next, we evaluate every adjacent pair of ORFs in this sorted list. If the 3’ end of the first ORF occupies a higher coordinate than the 5’ end of another ORF, then the pair are labelled as overlapping. After screening all adjacent pairs for overlaps, the results were serialized as a weighted graph in the Graphviz DOT language [27], where each node represents a cluster of homologous ORFs. Specifically, we generated two edge lists, one weighted by the frequency that ORFs in either cluster were adjacent in genomes, and a second weighted by the frequency of overlaps. When rendering graphs, we varied edge widths in proportion to the respective weights. In addition, we used the Matplotlib [28] library in Python to visualize the gene order (synteny) of representative genomes in each virus family as concatenated ORFs coloured by cluster assignments, and to visualize the distribution of gene labels by cluster as ‘word clouds’.

Graph analysis

To analyze the distribution of OvRFs in the context of the adjacency graph of a given virus family, we encoded the numbers of overlaps between every pair of clusters (represented by nodes) as a binomial outcome, given the weight of the corresponding adjacency edge. For every node $A$ in the graph, we recorded the number of genomes; number of adjacency edges (degree size); number of triangles ($A \leftrightarrow B \leftrightarrow C \leftrightarrow A$); transitivity (frequency of $B \leftrightarrow C$ given that the graph contains $A \leftrightarrow B$ and $A \leftrightarrow C$); and the Eigenvector centrality [29], a measure of node importance similar to
Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

Google’s PageRank algorithm. Next, we summed these quantities for the two nodes of each edge. We used the resulting values as predictor variables in a zero-inflated binomial regression on the probability of overlap edges, using the \textit{zibinomial} function in the R package VGAM \cite{30}. This mixture model extends the binomial distribution with a third parameter for the probability of zero counts in excess of the binomial. To reduce the chance of overfitting the data, we used stepwise Akaike information criterion (AIC)-based model selection (VGAM function \textit{stepAIVglm}), where the model search space was limited to the intercept-only model as the lower bound, and the full model with all predictors as the upper bound. All source code used for our analyses are available under the MIT license at \url{https://github.com/PoonLab/ovrf-viz}.

2.1.4 Results

To examine the distribution of overlapping open reading frames (OvRFs) across virus genomes, we retrieved 451,228 coding sequences from 12,609 representative virus genomes as identified by the NCBI virus genomes resource \cite{11}. Based on the annotation of coding sequences (CDS) in each record, we identified 154,687 OvRFs in 6,324 viruses (50.2\%). None of the circular ssRNA viral genomes \((n = 39)\) contained any OvRFs based on genome annotations. Using the taxonomic annotations in these records, we were able to assign 9,982 (79.2\%) of the genomes to Baltimore groups (Fig 2.1B). Of the remaining 2,627 genomes, 1,001 (38.1\%) were comprised of DNA and 1,626 were RNA, based on the molecular type annotation of the respective records.

\textbf{Longer genomes tend to carry shorter overlaps}

As expected, the number of OvRFs per genome was positively correlated with the number ORFs (Spearman’s \(\rho = 0.89, P < 10^{-12}\), Fig 2.1A). In addition, the relative number of OvRFs, \textit{i.e.}, normalized by the number of ORFs per genome, varied significantly among Baltimore groups (ANOVA, \(F = 835.2, df = 6, P < 10^{-12}\)). For instance, double-stranded DNA (dsDNA) viruses — the largest group of viruses in our sample — encode on average 202.8 ORFs and 32.7 overlaps per genome. An extreme case from the Phycodnaviridae family is the Paramecium bursaria Chlorella...
Figure 2.1: Distribution of overlapping genes across virus genomes. A. Scatterplot displaying a positive correlation between the log-transformed numbers of overlapping open reading frames (OvRFs) and ORFs per virus genome, stratified by Baltimore class. Genomes with no OvRFs were plotted at 0.5 (labeled ‘0’) with random noise to reduce overplotting. B. Barplot of the number of representative virus genomes per Baltimore class, which also serves as a colour and point-type legend for the scatterplots. ‘DNA’ and ‘RNA’ correspond to the molecular type annotations of virus genomes that have not been assigned to a known virus family. C. A log-log scatterplot displays the distribution of genomes with respect to overall length (in nucleotides, y-axis) and mean length of overlapping regions (x-axis) by Baltimore class. Individual plots are provided in [2.7] Fig. Underneath, ridgeplots summarize the marginal distributions of genomes with respect to mean overlap lengths, to clarify differences between the Baltimore classes.

virus, which infects a eukaryotic algal host and whose genome encodes 1,733 proteins with 541 (31%) overlaps.

In contrast, positive sense single-stranded RNA (ssRNA+) viruses encode on average 9.2 ORFs with 2.5 overlaps per genome, and negative sense single-stranded RNA (ssRNA−) viruses encode about 7.1 ORFs and 1.6 overlaps per genome on average. Some RNA virus genomes have abundant overlapping regions, however; e.g., the simian hemorrhagic fever virus genome (Genbank accession NC_003092) encodes 15 ORFs of which 10 are involved in an overlap.
In contrast, the mean number of nucleotides in overlapping regions was negatively correlated with genome length overall (Spearman’s $\rho = -0.52$, $P < 10^{-12}$; Fig 2.1C). We note that this comparison excludes genomes without any OvRFs, which were significantly shorter (average 6038 nt versus 51424 nt; Wilcoxon rank sum test, $P < 10^{-12}$). After adjusting for multiple comparisons ($\alpha = 6.25 \times 10^{-3}$), correlations remained significantly negative within dsDNA, dsRNA, ssRNA+ and unclassified DNA viruses only (Fig). Correlations within Baltimore classes were largely driven in part by variation among virus families, and we found no consistent trend in correlations within families using a binomial test. While DNA viruses, including single-, double-stranded and unclassified species, tended to carry longer genomes (median 33489 nt, interquartile range (IQR) 2768–59073 nt), their overlapping regions tended to be relatively short (median 15.6 nt, IQR 8.5–61 nt). This trend was largely driven by the dsDNA viruses, and the distributions of overlap numbers and lengths in unclassified DNA genomes (Fig 2.1C) suggest that these predominantly also represent dsDNA viruses. In comparison, RNA viruses carried fewer but relatively long overlapping regions (median 169.12 nt, IQR 31.34–831.75 nt) for their shorter genome lengths (median 4046 nt, IQR 1986–8009 nt).

**Distribution of frameshifts among OvRFs**

5,733 (3.7%) of the 154,687 OvRFs identified in our study involved the alternative splicing of one or both transcripts such that there is no consistent relationship between reading frames. These cases are excluded from this section because they complicate the interpretation of frame shifts. The majority ($n = 92,915$, 62.4%) of OvRFs involved reading frames that were shifted by 2 nt on the same strand (+2; Fig 2.2). These mostly represented dsDNA virus genomes ($n = 78,191$, 84.2%) and comprised almost entirely of overlaps by a single nucleotide (T[AG]ATG) or 4 nt (ATGA). (Note that the density plots in Fig 2.2 summarize the distribution of overlap lengths at the level of individual OvRFs, whereas Fig 2.1C summarizes the mean overlap lengths at the level of virus genomes. The peaks in Fig 2.2 do not appear in Fig 2.1C because a majority of 1nt and 4nt overlaps appear in a much smaller number of dsDNA genomes.) We observed +2 overlaps
Figure 2.2: Associations between overlap lengths and frame shifts. (left) Ridgeplots summarizing the distributions of overlap lengths for different frame shifts, where +2 indicates a shift by 2 nt relative to the upstream reading frame, and –2 indicates a 2 nt shift on the opposite strand (note the reverse complement of CAT is ATG). For +2 and –2, we also display the densities after removing overlaps by 1 and 4 nt (dashed outlines), since these predominate the respective distributions. (right) Treemaps summarizing the distribution of frame shifts by Baltimore class. The area of each rectangle is scaled in approximation to the relative frequency of each frame shift.

significantly more often among OvRFs from DNA viruses than from RNA viruses (odds ratio, OR = 3.3; Fisher’s exact test, \( P < 10^{-12} \)). Furthermore, only four out of 29,906 (0.01\%) overlaps by 1 nt involved a frame shift other than +2. These four cases involved –2 shifts where one of the ORFs was initiated by the alternate start codon TTG (e.g., CATTG). Another common type of short OvRF involved –2 frame shifts with an overlap of 4 nt, e.g., CTAA, where the reverse-complement of TAG is CTA. These were predominantly found in dsDNA virus genomes \( (n = 1423, 73.3\%) \). However, a substantial number \( (n = 419, 21.6\%) \) were also recorded in ssDNA viruses in which a complementary negative-sense strand is generated during virus replication, e.g., Geminivirus.

Excluding OvRFs with short overlaps of 1 or 4 nt, the most common type of OvRF involved a shift of +1. These were observed in both DNA viruses \( (n = 34,175 \text{ dsDNA}, 1,448 \text{ ssDNA}, \text{ and } 7,505 \)
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

unknown) and RNA viruses \((n = 40 \text{ dsRNA, 62 ssRNA−, 917 ssRNA+}, \text{ and 170 unknown})\). The median overlap length for \(+1\) OvRFs was 14 nt (IQR 8 to 26 nt). For this type of OvRF, overlaps exceeding 2,000 nt in length were found in ssRNA+ viruses, such as Kennedya yellow mosaic virus (NC_001746) and Providence virus (NC_014126). Overlap lengths tended to be longer in association with \(–0\) (median 114, IQR 27 – 267 nt) frameshifts (Wilcoxon rank sum test, \(P < 10^{-15}\); Fig 2.2).

**Graph-based approach to studying OvRFs**

Quantifying OvRFs by statistics like the number of overlaps per genome, or the mean overlap length, reveals substantial variation among Baltimore groups and different frameshifts. However, our objective is to characterize the distribution of OvRFs among virus genomes at a finer resolution. Specifically, these statistics, which are defined at the level of genomes, prevent us from identifying patterns in the distribution of OvRFs at the level of individual genes. For a meaningful comparison at the gene level among virus genomes, we need to be able to identify which genes are homologs. We decided to pursue our objective at the taxonomic level of virus families, to balance diversity in OvRFs with sequence homology. Identifying homologous genes among genomes at the level of virus families is challenging, not only because of substantial evolutionary divergence, but also because genomic rearrangements that can involve the gain, loss or relocation of ORFs, \(i.e.,\) changes in gene order (synteny). For example, the family Rhabdoviridae is characterized for the loss and acquisition of new genes that overlap with consecutive core ORFs, driving substantial variation in genome size and the formation of new accessory genes families \([31]\).

We used an alignment-free \(k\)-mer-based method \([22]\) to partition all amino acid sequences from genomes in a given virus family into clusters of homology \((2.8\) Fig). In brief, for each virus family we calculated a \(k\)-mer distance \([21]\) between every pair of amino acid sequences. We projected the resulting distance matrix into two dimensions by t-distributed stochastic neighbor embedding (t-SNE), and then applied hierarchical clustering to the distances in the 2D plane (Fig 2.3A). The clustering threshold was determined by balancing the mean frequency of a cluster across genomes
against the mean number of unique clusters per genome (Fig 2.3B).

**Figure 2.3: Adenoviridae family analysis.**

**A.** t-SNE projection of protein sequences from \( n = 71 \) genomes in the Adenoviridae. Each point represents a protein sequence, coloured and numbered by its cluster assignment. Based on our clustering criteria, we identified a total of 39 clusters for this virus family. **B.** A compact representation of reference genomes labeled by genus. Each set of line segments represent the coding sequences of a genome, coloured by cluster assignments and rescaled to a constant total length. White spaces represent non-coding regions. **C.** A hierarchical layout of the adjacency graph for Adenoviridae. Each node represents a cluster of homologous coding sequences, scaled in proportion to the number of sequences in the cluster. Node numbering and colours were determined by the order of appearance of clusters in the data. Directed edges (arrows) connect nodes representing coding sequences that are adjacent in five or more genomes. Edges are coloured blue if the genes overlap and grey otherwise; widths are scaled in proportion to the number of genomes in either case. This diagram was generated using Graphviz and arrows were manually modified in Inkscape.

We propose a graph-based approach to characterize the distribution of OvRFs in the context of coding sequences in the genome. This approach provides a framework for quantifying overlapping regions at a finer resolution within virus families, and is a natural method for visualizing differences.
2.1. **Using networks to analyze and visualize the distribution of overlapping genes in virus genomes**

between them. Each node in the graph corresponds to a cluster of homologous coding sequences. Nodes are connected by two sets of directed edges (arrows; Fig 2.3C). The first set represent the number of genomes in which coding sequences in the respective clusters are located next to each other (adjacency edges). A second set of edges represent the number of genomes in which the adjacent sequences are overlapping (overlap edges). Hence, an overlap edge is never present without an adjacency edge. Because edges are weighted by the number of genomes they each represent, an overlap edge can never have a weight that exceeds the matching adjacency edge.

**Example: Graph-based analysis of Adenoviridae**

Adenoviridae is a family of dsDNA viruses with genomes approximately 32,000 nts in length encoding around 30 proteins. Our clustering analysis of protein sequences in the \( n = 72 \) reference genomes identified 37 clusters (Fig. 2.3A). Fig. 2.3B displays the distribution of cluster assignments across coding sequences in the genomes. We noted that one of the genomes (bovine adenovirus type 2, NC_002513) had an unusually long non-coding region. We subsequently determined that this reference genome record was not completely annotated, removed it from the dataset and repeated our analysis, resulting in 39 clusters.

The adjacency graph for Adenoviridae features a relatively conserved gene ordering that corresponds to clusters 10 to 20 (Fig. 2.3C). In other words, this part of the graph has a mostly linear structure where nodes tend to have one incoming edge and one outgoing edge. Clusters 10 to 20 correspond to proteins encoded by regions L1-L5 (2.9 Fig). For example, cluster 15 predominately maps to the protein names including the term ‘hexon’. The graph also features several ‘bubbles’ in which one of the coding sequences is gained or lost in a substantial number of genomes. For example, some genomes proceed directly from cluster 11 (pVII) to 13 (pX), bypassing 12 (pV). Similarly, cluster 31 (IX or ORF0) is gained in at least 11 genomes. In addition, the graph splits between clusters 19 and 39 as it traverses from cluster 18 to 20. These clusters correspond to mixtures of the proteins 33K and 22K, which are both encoded by alternative splices of the same gene transcript that includes a long intron [32]. This split, as well as self-loop edges on both clusters,
implies that our clustering method can be confounded by inconsistent annotation of such isoforms.

The graph also contains distinct groups of nodes with multiple incoming or outgoing edges, which represent homologous clusters of coding sequences with more variable orderings in Adenoviridae. For example, clusters 8, 34, 36 and 38 generally correspond to ORFs in the E1 region of aviadenovirus (genus of bird-associated adenovirus) genomes associated with gene duplication events \[33\]. Similarly, cluster 25 maps to the RH family of duplicated genes in aviadenovirus and atadenovirus genomes. The presence of homologous coding sequences with potentially common origins in both the E1 (5’) and E4 (3’) regions of the Adenoviridae genomes induces the overall cyclic structure in this adjacency graph.

Figure 2.4: **Adjacency graphs for different virus families.** These graphs were generated from the clustered ORF data, using the same procedure that we employed to generate Fig 2.3 for Adenoviridae. Blue edges indicate overlapping open reading frames, and grey edges represent ORFs that are adjacent but not overlapping. Edge widths were rescaled by a factor of 0.5 for Geminiviridae and Papillomaviridae to accommodate differences in sample size (numbers of genomes) among virus families. Arrowheads were manually adjusted in Inkscape as in Fig 2.3.

In Adenoviridae, OvRFs vary from 1 to 19 nt in length, with a median length of 10 nt (IQR 8 – 12 nt). By visualizing clusters of homologous coding sequences as a graph, we can see that the conserved ‘backbone’ of L1-L5 genes are relatively free of overlaps. In addition, Fig 2.4 dis-
plays the adjacency graphs produced for four other virus families (Rhabdoviridae, Geminiviridae, Coronaviridae and Papillomaviridae, representing four different Baltimore groups. This visual comparison of adjacency graphs not only clarifies the substantial variation in the frequency of OvRFs among families, but also reveals differences in the distribution of overlaps among ORFs. For example, OvRFs in genomes of Geminiviridae tended to be associated with common adjacency edges on the ‘left’ side of the graph, corresponding to homologous ORFs closer to the 5’ end of genomes. OvRFs in Coronaviridae genomes tended to be associated with less common pairs of adjacent ORFs (e.g., 10-7 and 7-3 versus 8-4 and 4-5; Fig 2.4).

Variation among families

To analyze the distributions of OvRFs in the context of adjacency graphs, we fit a zero-inflated binomial regression model to the weights of overlap and adjacency edges for every pair of clusters. For example, out of 69 genomes with adjacent coding sequences assigned to clusters 6 and 7 in the Adenoviridae graph, 57 genomes had an overlap between the sequences and 12 did not. We calculated the number of genomes, degree size, number of triangles, transitivity and Eigenvector centrality as edge-level attributes, and used a stepwise model selection procedure to determine which combination of attributes was best supported by the data as predictor variables. The results of fitting these regression models to each graph are summarized in Fig 2.5. Effect size estimates varied substantially among virus families. For example, centrality was significantly associated with higher probabilities of overlaps for Geminiviridae, Papillomaviridae and Rhabdoviridae, but with lower probabilities in Coronaviridae. A cluster with high centrality is connected to many other clusters that are also of high degree size. In our context, high degree sizes correspond to ORFs with variable neighbours or diverse locations in a genome, e.g., due to multiple gene capture and duplication events [33]. Triangles in the adjacency graph tended to be associated with lower rates of overlap. For instance, gene loss by deletion (from A → B → C to A → C) is more likely to be tolerated if the adjacent ORFs A and C do not overlap with the targeted gene B. For Adenoviridae, transitivity had a relatively slight but significant negative effect on overlap probability (Fig 2.5).
Figure 2.5: Forest plot of zero-inflated binomial regressions on adjacency graphs. Points and lines are omitted for terms that were discarded by a stepwise AIC model selection procedure. Each point corresponds to coefficient estimates from zero-inflated binomial regressions on the probability of an overlap between adjacent ORFs, given the clusters of homologous ORFs from genomes of each of five virus families (see colour legend). Line segments correspond to the 95% confidence interval of the estimate, drawn in bold when the interval does not include zero. Total size = total number of genomes with adjacent ORFs assigned to the respective clusters. Edge degree = total edge degree of the linked clusters. Triangles = total number of triangles involving either cluster. Transitivity = total transitivity of the linked clusters. Centrality = total Eigenvector centrality of the linked clusters.
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

This is consistent with our previous observation that clusters comprising a core ‘backbone’ in the adjacency graph tended to be associated with fewer overlap edges.

2.1.5 Discussion

The number of genome sequences for previously unknown viruses is rapidly accumulating in public databases, driven in part by environmental metagenomic sequencing projects [34] and education/outreach programs like SEA-PHAGES [35]. These data provide a significant opportunity to examine the composition of these genomes to identify large-scale patterns in features such as overlapping open reading frames (OvRFs). The gold standard for collecting and processing such data for comparative analyses is manual curation, which enables investigators to correct frequent missannotations in public databases, even after additional curation by collaborative efforts [11]. For example, Pavesi et al. [36] curated an experimentally-validated set of 80 overlapping genes from virus genomes to examine differences in nucleotide and amino acid composition from non-overlapping ORFs. Rancurel et al. [37] manually curated 1,098 virus genomes to identify OvRFs for a phylogenetic analysis of their role in the de novo emergence of novel genes. However, the scale and rate of growth of these data makes it increasingly difficult to manually curate OvRFs, and it will eventually be unfeasible to curate the full repertoire of virus genomes. While we have focused on NCBI reference genomes that have been curated through community-based efforts [11], implemented automated quality control steps (such as excluding overlaps with lengths inconsistent with other annotation) and manually inspected outliers, we recognize that the resulting database is not as reliable as those obtained through additional expert curation.

Here, our focus was on developing and applying computational methods that can scale with the rapidly growing number of genomes. We have characterized the distribution of putative OvRFs in over 12,600 annotated virus genomes. Beginning with conventional comparative methods, we first confirmed previous findings that overlapping genes are ubiquitous across all Baltimore classes, with examples identified in 50.2% of the virus genomes. We observed that the majority of non-splicing OvRFs are short (e.g., less than 10 nucleotides). However, the small overlaps in our
study were predominantly by 1 or 4 nucleotides, whereas previous work [10] reported peaks at slightly longer lengths (3 and 7 nucleotides, respectively). We also confirmed previous reports [4, 5, 7] that the number of OvRFs increases with genome length, whereas OvRFs tend to be shorter in longer genomes. These trends are consistent with the compression theory that proposes that overlapping genes are a significant mechanism for reducing genome lengths [5]. However, we must be cautious about interpreting these patterns because, like previous work, there is no adjustment for non-independence among observations due to evolutionary homology, *i.e.*, identity by descent. This can be mitigated in part by examining correlations within virus families. At this level, we did not find significant evidence of a consistent association between overlap and genome lengths (2.7 Fig). To assess the sensitivity of our results to missannotations that were not removed by our filtering criteria, we ran a simple simulation experiment by modifying the start and end coordinates of ORFs with random deviates from a uniform distribution $U(-10, +10)$ for 10% and 50% of all genomes, respectively, and repeating our analyses (2.10 and 2.11 Figs). Genome-level trends in the numbers and mean lengths of overlaps were generally robust to this type and extent of missannotation. Misannotations had a more substantial impact on the distribution of frameshifts among overlaps; for instance, spurious overlaps of 1 or 2 nt became appreciably associated with −2 and +1 frameshifts with 50% missannotation; however, the overall trends remained the same.

In the absence of a standard notation for frameshifts in OvRFs, past studies have devised different labeling systems (2.6 Fig). For OvRFs on the same strand, we labeled the frameshift relative to the ‘upstream’ reading frame. Following [18], we used a negative sign to indicate that overlaps involve ORFs on opposite strands. However, we used −1 and −2 to indicate that the codons on the opposite strand are shifted by one and two nucleotides, respectively, relative to the −0 frame, which we consider to be a more intuitive notation. In an analysis of 701 RNA virus genomes, Belshaw and collaborators [5] previously reported that most overlapping genes consist of +1 and +2 frameshifts (+1 and −1 in their notation). We observed similar results in our analysis of 5,972 RNA virus genomes, which we further stratified by Baltimore class (Fig 2.2). We also encountered apparent overlaps between genes in a common reading frame, which we denote as +0. Even though
these OvRFs share codons, they yield different gene products where one is truncated relative to another, which may influence the folding and maturation of the respective proteins. These cases may thereby provide a means of differentiating between the compression [5] and antiredundancy [6] hypotheses of OvRFs in viruses, since +0 overlaps increase the selective burden of the same nucleotides, whereas other frameshifts increase the number of nucleotide sites under purifying selection. On the other hand, +0 overlaps have a much narrower repertoire of protein sequences and structures. Since these cases do not represent true OvRFs, we excluded them from our analysis.

In our data, antisense frameshifts (i.e., −0, −1 and −2) account for only the 6.6% of all overlaps and are primarily found in DNA virus genomes. For example, we detected a total of 14 cases of antisense frameshifts in RNA virus genomes (two instances in −2, four in −1, and one in −0). For example, two −1 overlaps 434 and 44 nt in length are annotated in segment S of the dsRNA virus Pseudomonas phage phiYY (NC_042073). Since these involve hypothetical proteins in a recently discovered dsRNA phage, however, we must be cautious about interpreting these results. In DNA viruses, −1 are slightly more common than −0 and −2 OvRFs (Fig 2.2), especially if we exclude the most common −2 overlap by four nucleotides (i.e., CTA). However, Lèbre and Gascuel [18] recently determined that the −1 frameshift (−2 in their notation) was the most constrained, in that the codons used in one ORF limit the amino acids that can be encoded in the other. It also minimizes the expected frequency of stop codons in the opposing strand, but −1 overlaps were not significantly longer than other types in our data. Moreover, we observed that overlaps of frameshift −0 tended to be longer than the other antisense overlaps (Fig 2.2). A unique property of the −0 shift is that there any combination of amino acids can be encoded without inducing a stop codon in the reading frame opposite [18], due to redundancy in the universal genetic code. Carrying over an example from Lèbre and Gascuel, there is no way to encode two tyrosines (YY) without introducing a stop codon in the −2 reading frame. This property of −0 overlaps may play a significant role in permitting greater lengths.

One of the key challenges for extending our comparative analysis to the level of individual ORFs was the assignment of ORFs into clusters of homology. This is complicated not only by
extensive sequence divergence at the level of virus families, but also the gain or loss of ORFs in different lineages through processes that include gene duplication. Furthermore, the annotation of ORFs in a general purpose public database like Genbank is not sufficiently consistent to rely on these labels. For example, cluster 15 in our analysis of Adenoviridae genomes comprised coding sequences with diverse labels, including ‘hexon’, ‘hexon protein’, ‘hexon capsid protein’, ‘L3 hexon’, ‘II’, ‘capsid protein II’, ‘protein II’, and the ubiquitous ‘hypothetical protein’ label (Fig. 2.9). We also found several examples of genomes in the NCBI Viral Genome Resource in which ORFs were incompletely annotated. The bovine adenovirus type 2 reference genome (NC_002513), for example, has only 11 annotated coding sequences. Adenovirus genomes typically contain about 30 to 40 genes. Since this reference genome lacks coding sequence annotations over a 15 kbp interval, it was apparent that many genes were simply not annotated. We subsequently confirmed this using a gene prediction and homology search analysis and discarded this reference genome from our analysis.

Given the abundance of genomic diversity at the level of virus families, the assignment of ORFs into homologous groups is not unambiguous. Thus, we utilized an alignment-free approach to cluster the coding sequences in the reference genomes for each virus family. There are a large number of alignment-free methods that extract k-mers from two input sequences (see [22] for a recent review). We chose the Bray-Curtis distance (also known as the intersection distance) because it performed comparatively well at the task of protein classification in a recent benchmarking study [22]. However, the classification analysis in that study was performed on protein databases curated to span a broad range of relationships, including both cases of evolutionary and structural homology. While alignment-free methods are generally regarded as more suitable for comparing more divergent viral genomes [38], we note that it is feasible to use a conventional sequence alignment program to generate a pairwise distance matrix for clustering analysis. To illustrate, we generated a distance matrix for the Papillomaviridae genome set by pairwise alignment with MAFFT, and then applied the rest of our analysis to the result. Originally, we obtained 10 clusters using our alignment-free, k-mer-based approach to generate a distance matrix (Fig 2.4). Using a word cloud
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

to visualize the distribution of labels among clusters, we observed a clear separation of E1, E2, E4, E7 and L2 labels among clusters, suggesting that the k-mer method does a reasonable job clustering homologous proteins for Papillomaviridae (2.12 Fig). For example, cluster 2 is predominantly associated with ‘L2’ and ‘minor capsid protein’ labels. Applying the same threshold criterion to the p-distance matrix from pairwise alignment resulted in a similar number of clusters (9). Clusters in this graph each presented an assortment of different labels for proteins that was not consistent with homology (2.12 Fig). For example, cluster 3 carried labels for L1, L2 and E1 at similar frequencies. The adjacency graph from the pairwise alignment method (2.13 Fig) was topologically similar to the graph in Fig 2.4. For instance, overlap edges were predominantly associated with a subset of four to five nodes in both cases. However, the lack of a consistent association of labels with nodes in the alignment-based graph precluded a more direct comparison between topologies.

To extract putative clusters of homologous ORFs, we were required to define some threshold to apply to the hierarchical clustering results. For any given threshold, some number of ORFs will be misclassified into separate clusters (false negatives) or into the same cluster (false positives) — this issue is common to all unsupervised clustering methods. For each virus family, we selected thresholds that minimized the number of duplicate cluster assignments per genome, while maximizing the overall frequencies of clusters across genomes. This criterion assumes that homologous ORFs are represented by a single member in every genome. For instance, viruses in the family Coronaviridae are characterized by five conserved genes — replicase polyprotein (1ab), spike (S), envelope (E), membrane (M) and nucleocapsid (N) — and a varying number of accessory proteins of low homology [39]. The conserved gene order is reproduced in the corresponding graph (Fig 2.4), which comprises a distinct chain of five nodes mapping to the respective genes (2.14 Fig). On the other hand, the remaining six nodes do not readily map to consistent subsets of accessory gene labels, e.g., NS3C, NS7 protein. Although it is intuitive, our criterion is as ad hoc approach that is not supported by an underlying model of genome evolution. Hence, developing improved and efficient clustering methods in this context will be an important area for further work.

To evaluate the role of OvRFs in the emergence of genes de novo by overprinting [7,8], we need
to characterize the distribution of overlaps at the level of individual ORFs. We employed unsupervised clustering of homologous ORFs to identify gene order polymorphisms at the level of virus families. Visualizing the syntenic relationships among clusters at this taxonomic level provided some interesting patterns. For example, members of Adenoviridae have about 16 conserved ‘core’ genes in the middle of the genome that are responsible for DNA replication and encapsidation, and the formation and structure of the virion [33]. These core genes formed a distinct backbone in our adjacency graph of this family that was relatively free of overlaps (Fig 2.3). However, clusters 5, 6 and 7 were connected by wide overlap edges indicating an abundance of overlaps between 5-6 and 6-7. These clusters correspond to highly conserved ORFs that were predominantly annotated as encoding the conserved maturation protein IVa2, DNA polymerase and pre-terminal protein (pTP), respectively. Thus, in some cases conservation of gene order is accompanied by conserved overlaps. On the other hand, cluster assignments tended to become more variable towards the 5’ and 3’ ends of the genome in association with an increasing frequency of OvRFs. Similarly, OvRFs in Coronaviridae tended to be associated with pairs of clusters representing accessory genes that were less frequently adjacent in these genomes (Fig 2.4). As a member of the virus order nidovirales, coronaviruses have undergone extensive selection to expand the repertoire of genes encoded by their relatively long RNA genomes [40]. Accessory genes assigned to clusters with overlaps tended to be found in varying locations in genomes of Coronaviridae, which is more consistent with horizontal gene transfer or duplication than overprinting.

Like any visualization method, there is a practical upper limit to the amount of information that can be represented by an adjacency graph. For instance, we restricted our graphs to employing node colour and size, and edge colour and width, to represent cluster identities, numbers of ORFs, and the type (adjacency, overlap) and frequency of relationships, respectively. While there is a larger repertoire of visual channels, e.g., node shape, adding information would make the graph increasingly difficult to interpret. Nevertheless, there are several other attributes of overlaps that are potentially of interest, such as the average length of overlaps or the predominant frameshift. To illustrate, we provide an alternative rendition of the adjacency graph for the Adenoviridae family.
2.1. Using networks to analyze and visualize the distribution of overlapping genes in virus genomes

in which edge widths represent the mean lengths of overlaps (Fig).

In summation, we have described and demonstrated a new approach to characterize the distribution of OvRF in diverse virus genomes. Adjacency graphs provide a framework for both visualizing these distributions and for hypothesis testing, i.e., effects of gene- or genome-level attributes on the frequencies of overlaps between specific clusters of homologous ORFs. In future work, we will develop comparative methods on the topologies and features of adjacency graphs to identify shared characteristics between virus families at this level. We further postulate that adjacency graphs may provide useful material for extending methods for ancestral gene order reconstruction [41], where the graphs can address the problem of uncertain labelling of genes. Ideally, one would simultaneously reconstruct the phylogeny relating observed genomes. Reconstructing ancestral gene order is already an NP-hard problem [42]. Given the diverse and evolutionarily fluid composition of many virus genomes, however, it is remarkable that the gain and loss of ORFs has not been explored as much as larger organismal genomes.

2.1.6 Supplementary Information

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+2</td>
<td>-1</td>
<td>+2</td>
<td></td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>+1</td>
<td></td>
</tr>
<tr>
<td>+0</td>
<td>0</td>
<td>+0</td>
<td></td>
</tr>
<tr>
<td>-0</td>
<td>rc0</td>
<td>-0</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>rc+1</td>
<td>-2</td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>rc-1</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>3'</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.6: Notation of the 6 possible frameshifts used in this study and two other studies [5] [18].
Figure 2.7: Scatterplots of genome length on mean overlap length by Baltimore class. Each point represents a virus genome, coloured by virus family. Families represented by only one genome were coloured in grey (smaller point size). Results from Spearman rank correlations are summarized in the lower left corner of each plot. The lower-right panel displays a box-and-whisker plot summarizing the distributions of Spearman’s rank correlation coefficients ($\rho$) for genomes within families, grouped by Baltimore class.
2.1. **Using networks to analyze and visualize the distribution of overlapping genes in virus genomes**

---

**A. Creation process of Adjacency plot**

- Multifasta file
- K-mer distance (substrings of length $k=3$)
- t-SNE algorithm (exaggerate distance between proteins to extract functional clusters)
- hclust (hierarchical clustering)
- cutree
- Adjacency plot

**B. Graphic representation of an Adjacency plot**

---

Figure 2.8: **Creation of adjacency plots A.** Steps used to generate the input file for the adjacency plot. First, we downloaded a multifasta file containing the protein sequences of the reference genomes for each species in the virus family. Then, we used a Python script to calculate the *k-mer* distance between proteins followed by an R script to designate each protein to a cluster according to homology. Finally, we used a Python script to generate dot files using Graphviz. **B.** Adjacency plot interpretation. Each one of the proteins that constitute a genome is assigned to a different cluster with homologous proteins from other species. From each cluster, we draw arrows in gray that represent adjacent proteins and arrows in blue that represent overlapping proteins. The width of the arrow is proportional to the number of proteins related between the two clusters. One cluster can have entries adjacent to proteins in different clusters. In this example, cluster 3 has proteins adjacent to proteins in cluster 2 and cluster 5.
Figure 2.9: **Word clouds of protein names mapped to clusters for Adenoviridae.** The size of each word (gene annotation) is scaled in proportion to its relative frequency in association with ORFs in the respective clusters.
Figure 2.10: Random introduction of 10% missanotation on the ORFs database.
Figure 2.11: Random introduction of 50% missannotation on the ORFs database.
2.1. **Using networks to analyze and visualize the distribution of overlapping genes in virus genomes**

**Figure 2.12:** Distribution of protein names across clusters in Papillomaviridae.

**Figure 2.13:** Adjacency graph based on proteins clustered according to a pairwise alignment for Papillomaviridae.
Figure 2.14: Distribution of protein names across clusters for Coronaviridae proteins.
2.1. **Using networks to analyze and visualize the distribution of overlapping genes in virus genomes**

Figure 2.15: Adjacency graph of Adenoviridae family. Overlapping edges are proportional to overlap length.
2.2 Clustering highly divergent homologous proteins: an alignment-free method

**Note:** This section provides a detailed description of the methods we used in section 2.1, which were published independently as a protocols paper [2].

The comparative analysis of amino acid sequences is an important tool in molecular biology that often requires multiple sequence alignments. In comparisons between less closely-related genomes, however, it becomes more difficult to accurately align protein-coding sequences, or even to identify homologous regions in different genomes. In this protocol, we describe an alignment-free method for the classification of homologous protein-coding regions from different genomes. This protocol was originally developed for comparing genomes within virus families, but may be adapted for other organisms. We quantify sequence homology from the overlap (intersection distance) of the $k$-mer (word) frequency distributions for different protein sequences. Next, we extract groups of homologous sequences from the resulting distance matrix using a combination of dimensionality reduction and hierarchical clustering methods. Finally, we demonstrate how to generate visualizations of the composition of clusters with respect to protein annotations, and by colouring protein coding regions of genomes by cluster assignments. These provide a useful means to quickly assess the reliability of the clustering results based on the distribution of homologous genes among genomes.

### 2.2.1 Keywords

Alignment-free methods, protein clustering, bioinformatics, R, Python.

### 2.2.2 Introduction

Many comparative methods for analyzing genomic data are dependent on identifying homologous sets of protein-coding genes. This is a challenging problem for divergent genomes. For exam-
2.2. Clustering highly divergent homologous proteins: an alignment-free method

Comparing virus genomes, which are relatively compact, becomes more difficult at higher taxonomic levels such as families — not only because the gene sequences will have accumulated substantial numbers of mutations, but also because entire genes are more likely to be gained or lost over time. This problem is exacerbated by inconsistent annotation of gene features in virus genomes. Gene sequences that encode the homologous proteins in different genomes may have different labels, e.g., ‘hexon’ and ‘capsid protein II’, or carry no labels at all. In addition, there are more new virus genomes being deposited in public databases than ever before, owing in part to the proliferation of environmental metagenomic sequencing studies [43, 44] and increasingly efficient algorithms for de novo assembly [45]. In many of these new virus genomes, protein-coding gene sequences are not annotated with labels other than ‘hypothetical protein’, even if homologous sequences can be found in other genomes.

To address this problem, we have developed an alignment-free method to generate clusters of homologous gene sequences in different genomes [1]. Like the majority of alignment-free methods [22], we use a k-mer (‘word’ counting) approach to quantify the similarity of amino acid sequences. We focus on protein sequences because they are more likely to retain sequence homology over longer evolutionary distances than the underlying nucleotide sequences, due to purifying selection. Bypassing the step of aligning large numbers of amino acid sequences makes it more feasible to compare large numbers of virus genomes, genomes encoding many genes, or both.

2.2.3 Basic protocol 1: Data collection and processing

To cluster homologous proteins, we require a file with amino acid sequences. Multi-FASTA files can be generated by manually searching for protein sequences and downloading them the FASTA format. To automate this process of retrieving large numbers of sequences with consistent information for all proteins, we wrote a script that uses a list of Genbank accession numbers to query the NCBI database via the Entrez application program interface (API) in BioPython [16]. Our script converts the results into a FASTA file with sequence labels comprising: the accession number of
the genome encoding the protein, product name, strand, and specific location of the protein in the genome (start and end positions). Users can run the script from the terminal by using the Python3 interpreter.

**Necessary resources**

**Hardware**

Computer with internet connection.

**Software**

- UNIX-like operating system (Linux or macOS).
- Python version 3.6 or higher (available at [https://www.python.org/](https://www.python.org/)).
- BioPython (including Entrez, SeqIO, and SeqFeature submodules) (available at [https://biopython.org/](https://biopython.org)).

**Input files**

Table (.txt) with NCBI accession numbers for genomes of interest.

**Optional: Gathering accession numbers**

To retrieve a list with accession numbers associated with an Entrez query, you can access the NCBI website ([https://www.ncbi.nlm.nih.gov/](https://www.ncbi.nlm.nih.gov/)) and select the ‘Nucleotide‘ option to search your sequences of interest. In the left panel, you will find filtering options such as reference sequences, sequence length, or molecule type. When you are done, click the ‘Send to‘ button at the upper right corner of the page. In the drop-down menu that appears, select the ‘File‘ file destination, and the ‘Accession List‘ format. Once your click the ‘Create File‘ button, a list with accession numbers will be created in the default downloads folder of your web browser (Figure 2.16). From here, we
2.2. Clustering highly divergent homologous proteins: an alignment-free method

recommend you change the name of the file to a more informative label, and move it to a folder from which you intend to perform the rest of the analysis.

Figure 2.16: **Downloading list with accession numbers.** From the Nucleotide database in NCBI, download accession numbers for the reference sequences of Adenoviruses.

**Accessing NCBI to obtain a muti-FASTA file with aminoacid sequences**

To automate the data collection, we wrote a Python script (*protein_scraper.py*) that uses the Bio.Entrez submodule to fetch protein information by sending direct requests to the NCBI Entrez API. The script loops through a list with the accession numbers of genome sequences of interest, and retrieves the corresponding records from the NCBI database. The Genbank identification number (gid) is used to fetch a Genbank record with the genome information, including annotations for its coding sequences (CDSs). For each CDS, we retrieve details regarding product, strand,
location and amino acid sequence, and export this information to a multi-FASTA file (Figure 2.17).

**Downloading pipeline**

Using the Terminal application (command-line interface), download the scripts required to run the pipeline by using `git` to clone the PoonLab/ProtClust repository from GitHub (https://github.com/PoonLab/ProtClust):

$ git clone https://github.com/PoonLab/ProtClust

**Running protein-scraper.py from the terminal**

1. Use `cd` to move to the scripts folder, and `ls` to verify that all the scripts for the analysis have been downloaded:

   $ cd ProtClust/scripts

   $ ls

   kmer.py opt_kpca.R protein_scraper.py

2. Download amino acid sequences from the terminal by specifying the interpreter (Python3), the code (protein-scraper.py), the path to input file (list with accession numbers), an email address (required to submit queries to the NCBI Entrez API), and a path to the output file (FASTA file with protein sequences) with the option `--outfile`:

   $ python3 protein-scraper.py <path/to/accno_list> <email> --outfile <Path/to/output/file>

**Sample data**

We provide a sample file (my_acnnos.txt) for this protocol in the data folder. The result from using this accession list as an input is the sample_proteins.fa file. Example of both files are shown in Figure 2.17.
2.2. Clustering highly divergent homologous proteins: an alignment-free method

![Input: my-accnros.txt](image)

Text file with list of accession numbers

**download sequences: run protein_scraper.py**

```bash
laura@eva:~/Projects/ProtClust$ python3 scripts/protein_scraper.py data/my-accnros.txt laura@uwo.ca --outfile data/proteins.fasta
AC_000001.1
AC_000002.1
AC_000003.1
AC_000004.1
AC_000005.1
AC_000006.1
AC_000007.1
AC_000008.1
AC_000009.1
AC_000010.1
AC_000011.1
AC_000012.1
Done
Protein file at: data/proteins.fasta
```

From the terminal, run the script specifying the list with accession number, an email, and the name for your output

**output: proteins.fasta**

```
>"Ovine mastadenovirus A,AC_000001.1,E1A,1,477:907;993:1187"
MRHLRLAFDWRFWEAENLQQDLTHSEDEDFEPLTLQDLVELESVPNAYFFPDADIPQELPTDAAAE
>"Ovine mastadenovirus A,AC_000001.1,E1B 19K,1,1291:1771"
MDLSLQLCEQLSLPLQRVIYYATNRASWWSRTFGGRRLANLVLVTIKVEEAELSQELFNEDEGFQWMFGSG
>"Ovine mastadenovirus A,AC_000001.1,E1B 55K,1,1668:2859"
MLQAQDAAVPAEASGASVRQVQGFHPFARGNLEPNKVTYQQILSEFQADPFFTNDRYDEFHELVTHLLAE
>"Ovine mastadenovirus A,AC_000001.1,IX,1,2953:3304"
MADEGDIRTSFLTARLHRWAGVRRHNAIGSNISGPVSPEIVSTVRNRTDATELTRNITTTTRPAAEQLNLE
>"Ovine mastadenovirus A,AC_000001.1,IvA2,-1,14929:4942;3326:4651"
MEQRAGEPALSHQPPEFKTDPRYEKRPQKRPVHSRRNFNAHSEALERQDPCRHRPPPADALLEKQPKPK
```

Multi fasta file with amino acid sequences of CDSs on query genomes

Figure 2.17: **Download protein sequences.** Use `protein_scraper.py` to download the amino acid sequences encoded in a genome.
2.2.4 Basic Protocols 2: Calculating $k$-mer distances

This protocol describes a simple way to identify homologous proteins between highly divergent and incompletely annotated coding sequences. To do this, we use a script that takes a multi-FASTA file and decomposes every sequence into counts of words ($k$-mers) of one to three amino acids in length, i.e., $k = 1, 2, 3$. We calculate the distance between sequences based on the intersection distance \cite{23} between their respective $k$-mer frequency vectors.

Necessary resources

Hardware

Computer.

Software

- Python version 3.6 or higher
- built-in Python modules csv, math, re, sys, argparse
- custom Python module gotoh2, \url{https://github.com/ArtPoon/gotoh2}

Input files

FASTA file with protein sequences (generated in Protocol 1).

Compute distance matrix

To get a distance matrix from your protein data, run the \texttt{kmer.py} script located in the \texttt{scripts} folder by specifying the location of the FASTA file, the output file with the distance matrix, and whether you want to extract the information on the headers as a separate file as described in Figure \ref{fig:2.18}. To use a string kernel distance (d2s, Luczak et al. \cite{23}) instead of default intersection distance, you need to specify the option \texttt{--kernel}. 

90
Run `kmer.py` from the command line using the FASTA file your created in the previous protocol:

```
$ python3 kmer.py <path/to/protein_file> <path/to/distance_matrix> -header
      <path/to/header_file>
```

**Algorithm description**

The script iterates through all sequences in the FASTA file and decomposes every amino acid sequence \( (s) \) into the set of all substrings (words or \( k \)-mers) comprising one to three consecutive amino acids. We use the set of all words in a sequence \( (W(s)) \) to calculate its frequency of words \( f(s, w) \), where \( w \) refers to a specific word. Then, we calculate the intersection distance between two sequences \( s \) and \( t \):

\[
k(s, t) = 1 - \frac{\sum_{W(s) \cap W(t)} 2 \min(f(s, w), f(t, w))}{\sum_{W(s)} f(s, w) + \sum_{W(t)} f(t, w)}
\]

This distance can be visualized by considering two overlapping histograms. The sum of minimum values in the numerator of the above equation corresponds to the overlapping region. It is adjusted by a factor of two to account for the overlap. The sums in the denominator represent the total area covered by the two histograms, counting the overlapping region twice. Dividing one value by the other results in a similarity measure, which we subtract from the maximal value of one to obtain a distance.

**Outputs**

The distances between pair of sequences are stored in a matrix in a comma-separated values (CSV) format. If you provide another file path to the `--header` argument, then the CDS information — including description, accession number, gene name, forward/reverse orientation, and genome coordinates — are extracted from the FASTA headers and written to this separate file.
Figure 2.18: **Calculate distances between pairs of proteins.** Use kmer.py to find the intersection of k-mer counts between your proteins.
2.2. Clustering highly divergent homologous proteins: an alignment-free method

2.2.5 Basic protocol 3: Extracting clusters of homology

We identify clusters of homologous proteins using the \( t \)-distributed stochastic neighbor embedding method (\( t \)-SNE, Van der Maaten and Hinton [46]) on the distance matrix calculated in Basic Protocol 2. This algorithm maps each of the data points (amino acid sequences) to a two- or three-dimensional space in such a way that preserves their actual pairwise distances in the original feature space. Next, we use a hierarchical clustering algorithm to generate a new distance matrix from the coordinates of the embedded points. Dimensionality reduction and clustering methods are frequently combined in this way because distance measures can have unexpected properties in high-dimensional feature spaces [26]. Finally, we extract clusters by applying a height cutoff to the dendrogram produced by hierarchical clustering.

As we lower this cutoff, genes in a particular genome will increasingly be assigned to different clusters — at the lowest cutoff, every gene will be in its own cluster. We quantify this by the mean percentage of genes with unique cluster indices, averaged across genomes. Conversely, increasing the cutoff will increase the expected proportion of genomes in which a given cluster will be found — at this other extreme, all genes in all genomes will belong to the same cluster. Therefore, we assume that the optimal cutoff minimizes the difference between these proportions. The pipeline to extract clusters of homology is depicted in Figure 2.19.

Necessary resources

Hardware

Computer.

Software

- R version 3.4 or higher

- R package Rtsne (https://github.com/jkrijthe/Rtsne)
Figure 2.19: **Cluster homologous proteins.** Modify `opt_kpca.R` to cluster your proteins from a distance matrix. Use the k-mer distance matrix and the headers file created on the previous step. Find clusters of homology and save your results as a comma separated file.

**Input files**

CSV files generated in Basic Protocol 2:

- Matrix with k-mer distances between proteins.
- Header file with information about your sequences.

**Preparing your R environment**

1. Identify the location of your input files (Figure 2.19). In the terminal, move to the folder where your data is located and run the `pwd` command. You will need this information to set your working directory in R.

   ```
   $ pwd
   /Path/to/distance/matrix
   ```

2. Open the `opt_kpca.R` script in a text editor. In this script, use `setwd()` to specify the
2.2. Clustering highly divergent homologous proteins: an alignment-free method

working directory where your input files are located (distance matrix and header), and read
the files with read.csv(), parsing them as comma separated values (CSV) formatted data
(modify lines 6, 7 and 9):

```r
setwd('/Path/to/distance/matrix') # lines 6-9
km <- read.csv('kmer-distance.csv', header=F, row.names=1)
km <- 1-as.matrix(km)
headers <- read.csv('protein-headers.csv', header=T)
```

Running the clustering algorithm

3. Run t-SNE method to reduce dimensionality of the k-mer matrix. We specify that the input is
a distance matrix, require to see progress updates, and specify that dimensions of the output
should be 2. The returned object includes a matrix containing the new coordinates of the
data mapped to the low dimensional space:

```r
require(Rtsne)
res <- Rtsne(km, is_distance=T, verbose=T, dims=2)
```

4. Use hierarchical clustering using hclust with Ward’s criterion [25]:

```r
hc2 <- hclust(dist(res$Y), method='ward.D2')
```

5. Extract an optimal number of clusters with cutree:

```r
acc <- headers$accession # Genomes labeled by accession number
acc <- as.character(acc)
n.acc <- length(unique(acc))

obj.func <- function(h) {
  clusters <- cutree(hc2, h=h)
  x <- sapply(split(clusters, acc), function(x) {
```
```r
# Chapter 2

```tab <- table(x)
sum(tab==1) / length(tab)

y <- sapply(split(acc, clusters), length) / n.acc
(mean(x)-mean(y))^2
```

```opt <- optimize(obj.func, c(0, 100))
clusters <- cutree(hc2, h=opt$minimum)
```

6. Plot your clustering results:

```gg2.cols <- function(n) {
  hues = seq(15, 375, length = n + 1)
  hcl(h = hues, l = 65, c = 100)[1:n]
}
pal <- gg2.cols(n=max(clusters))
par(mfrow=1:1)
plot(res$Y, type='n')
text(res$Y, label=clusters, col=pal[clusters], cex=0.8)
```

7. Save your results as a CSV file with information for the amino acid sequences including genome name, accession number, product name, whether the protein is encoded in the forward strand, the coordinates of the CDS in the genome, and the cluster it was assigned to (this file will be written to your current working directory unless you specify another path):

```info <- cbind(headers, clusters)
write.csv(info, 'clustered-proteins.csv')
```
2.2. CLUSTERING HIGHLY DIVERGENT HOMOLOGOUS PROTEINS: AN ALIGNMENT-FREE METHOD

Support Protocol: genome plot based on clustering results

To visualize and interpret your results, we provide a Python script that creates a plot where proteins are coloured in each genome based on their cluster assignments (Figure 2.20). In addition, you can generate ‘word clouds’ to identify patterns on the labels of proteins that were assign to the same cluster. This information is useful to detect problems with your data such as incomplete genomes, misannotated proteins, and partial sequences that should be excluded from the analysis. To do this, run the genome-plot.py script on the CSV file generated on Basic Protocol 3 (clustering section), i.e., the output file from opt_kpca.R.

Necessary resources

Hardware

Computer.

Software

- Python version 3.6 or higher.
- Built-in Python modules: argparse, math, csv.
- Third-party Python modules:
  - pandas [47]
  - numpy [48]
  - seaborn [49]
  - matplotlib [50]
  - wordcloud (https://github.com/amueller/word_cloud)
Figure 2.20: **Visualizing the results of the clustering assignment.** Use plot-genome.py to create informative diagrams with your clustering results. In the genome plot, proteins are coloured based on cluster number. Additionally, the script produces a wordcloud to observe the frequency of the annotations (names) of proteins in each cluster.
2.2. Clustering highly divergent homologous proteins: an alignment-free method

Input files

Clustering results from Basic Protocol 3 in CSV format. Each line should have information of a single protein including genome, location, name, and assigned cluster.

Plotting your genomes

1. From the terminal, run `plot-genome.py` by specifying the location of the CSV input file with all relevant information about your proteins. Optionally, you may specify output filenames for the genome plot and the word clouds. If no output is specified, the filenames will default to the ‘None’ prefix:

   $ python3 scripts/plot-genome.py <Path/to/clusters-info> --outfile <label>

2.2.6 Guidelines for understanding results

The purpose of this protocol is to identify homologous proteins using an alignment-free clustering method. From beginning to end, users will generate three main files: (1) a multi-FASTA file with amino acid sequences encoded by their genomes of interest, (2) a file with distance measurements between proteins, and (3) a file where each protein is assigned to a cluster. We have previously used this procedure to analyze the distribution of overlapping genes in the virus genomes belonging to the Adenoviridae, Genimimiviridae, Coronaviridae, Papillomaviridae, and Rabdoviridae families [1]. We noticed that, even though genomes of virus species within a family can differ in the number of proteins, genome length, or genome organization, the number of clusters determined by our selection criteria tended to be similar to the mean number of proteins per genome. In addition, viruses have conserved genes that encode proteins with essential functions for the virus replication cycle. Therefore, the number of clusters and the presence of large clusters with a similar number of members as the overall number of genomes, can be indicators of a reliable analysis.

To identify potential problems with your results, we recommend visually inspecting the genome
In Figure 2.20, we show the results of using the clustering protocol over all reference genomes on the Adenoviridae family available at NCBI on November 7, 2022. In this plot, long blank (white) intervals interrupted by sporadic colored line segments representing clustered genes indicate incomplete genome sequence records. Cases of partial or unannotated entries are not uncommon in the NCBI database, even for reference (RefSeq) genomes. We highly recommend carefully searching and removing such records from your analysis. From the Adenoviridae example, we detected the presence of a set of core proteins common in all species, that are located in the central part of the genome. In contrast, the 5′ and 3′ regions tended to be more variable in their protein composition. We also observed variation in protein annotations based on the wordclouds. For example, cluster 25 is mainly composed of proteins labeled as fiber protein. Similarly, there is a majority of proteins labeled as E4, E3, and ORF3 in cluster 24. However, we did not find consistent labels for genes associated with clusters 14, 6, or 30, for example.

Overall, this protocol provides an automated method to explore the distribution of homologous proteins in a set of genomes. The success of this method relies on the quality of the input database, and your ability to accurately detect corrupted records. Robust results can provide a detailed basis for understanding the protein-coding composition of your genomes. However, users should be aware that our method could be influenced by protein length, number of proteins per genome, total genome length, or number of genomes in the analysis.

2.2.7 Commentary

Background information

The set of protocols described here provide an alignment-free method to detect similarities between genomes, and to explore changes in gene arrangements caused by fast mutation rates, gain and loss of genes, and recombination events. Initially, we created this methodology to study viruses, which are characterized for being rapidly evolving, such that virus genomes can undergo major changes over directly observable time spans. The comparative analysis of the protein-coding components of
2.2. Clustering highly divergent homologous proteins: an alignment-free method

virus genomes relies in accurate annotation of ORFs. However, gene annotation is an increasingly challenging problem. For instance, the number of reference virus genomes in the NCBI RefSeq database has increased more than five-fold between 2000 and 2015, driven in part by the increasing use of next-generation sequencing platforms [11]. Newly discovered ORFs may systematically lack annotation across virus species, even within the same virus family. Our motivation behind developing an alignment-free clustering method was to be able to analyse virus genomes in the same virus family. This procedure allowed us to identify homologous proteins in highly divergent virus genomes, particularly in the absence of reliable genome annotations.

This protocol describes three methods that can be used independently for: automatically downloading proteins encoded in a list of genomes, finding homologous sequences without the need for alignments, and classifying proteins in clusters of similarity.

**Critical Parameters**

This method was intended to classify protein-coding regions from different genomes based on their sequence homology. Consequently, it is unlikely to work properly if applied to completely unrelated genomes (e.g., DNA and RNA virus genomes), or distantly related genomes that do not retain sufficient homology (e.g., virus genomes from different families).

**Troubleshooting**

The best way to assess the reliability of your results is by generating genome plots using the provided methods. In this plot, the protein-coding regions in each genome are coloured by their respective cluster assignments. This visualization is useful for evaluating the distribution of homologous genes among genomes. For instance, it provides a quick method for determining which genome sequences are likely incomplete, or an overview of how proteins assigned to different clusters are distributed among genomes. Additionally, the wordclouds provide a convenient visual summary of the different labels associated with the proteins in each cluster.
<table>
<thead>
<tr>
<th>Problem</th>
<th>Possible cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incomplete genomes</td>
<td>As shown in Figure 2.20, there are incomplete genomes (horizontal lines) with only one protein, or with missing sections. This is usually caused by annotation errors.</td>
<td>Remove the incomplete genomes from the analysis by establishing a threshold for the number of proteins encoded by a genome for it to be incorporated on the analysis. This step can be performed in R before using t-SNE.</td>
</tr>
<tr>
<td>Missing packages</td>
<td>For all your scripts, both in Python and R, it is common to find that some of the packages or dependencies are not installed on your computer.</td>
<td>Read the error messages to detect the missing packages. Refer to the documentation of the package and make sure that you are running your programs under the correct versions before installing them.</td>
</tr>
</tbody>
</table>

Table 2.1: Sources and solutions to potential errors

### 2.2.8 Conflict of interest

The authors declare no competing interest.

### 2.2.9 Data availability

All our scripts and input examples have been released under a permissive free license GNU General Public License at [https://github.com/PoonLab/ProtClust](https://github.com/PoonLab/ProtClust)
2.2. CLUSTERING HIGHLY DIVERGENT HOMOLOGOUS PROTEINS: AN ALIGNMENT-FREE METHOD

2.2.10 Acknowledgements

This work was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC Discovery Grant RGPIN 05516-2018).
Bibliography


Chapter 3

Simulating evolution in overlapping reading frames

Building upon the established ubiquity of overlapping reading frames (OvRFs) in virus genomes, as demonstrated in the previous chapter, the aim of this chapter is to further investigate the impact of OvRFs on virus evolution.

Studying the evolutionary dynamics of OvRFs is challenging because the effect of a nucleotide substitution has multiple contexts. In such scenarios, computer simulations provide a valuable tool for evaluating different models that can explain empirical observations. In Chapter 1, I provided an overview of existing programs that simulate evolution in a sequence. Nevertheless, there is currently no program capable of simulating evolution within overlapping sequences that considers the selection pressures of the various open reading frames (ORFs) to generate a multiple sequence alignment. To address this gap, I developed HexSE, a simulation model of nucleotide sequence evolution along a phylogeny. HexSE incorporates a customized data structure to track substitution rates at every nucleotide site and integrates information about different selection pressures acting on overlapping genes.

This chapter provides a detailed description of the HexSE model, which was previously published in Virus Evolution as a Resources paper.
3.1. Availability and implementation

**HexSE: Simulating evolution in overlapping reading frames**

Laura Muñoz-Baena¹, Kaitlyn E. Wade², Art F. Y. Poon¹²

¹ Department of Microbiology and Immunology, Western University, London, ON, Canada.
² Department of Pathology and Laboratory Medicine, Western University, London, ON, Canada.

Gene overlap occurs when two or more genes are encoded by the same nucleotides. This phenomenon is found in all taxonomic domains, but is particularly common in viruses, where it may provide a mechanism to increase the information content of compact genomes. The presence of overlapping reading frames (OvRFs) can skew estimates of selection based on the rates of non-synonymous and synonymous substitutions, since a substitution that is synonymous in one reading frame may be non-synonymous in another, and vice versa. To understand the impact of OvRFs on molecular evolution, we implemented a versatile simulation model of nucleotide sequence evolution along a phylogeny with any distribution of open reading frames in linear or circular genomes. We use a custom data structure to track the substitution rates at every nucleotide site, which is determined by the stationary nucleotide frequencies, transition bias, and the distribution of selection biases (dN/dS) in the respective reading frames.

### 3.1 Availability and implementation

Our simulation model is implemented in the Python scripting language. All source code is released under the GNU General Public License (GPL) version 3, and is available at [https://github.com/PoonLab/HexSE](https://github.com/PoonLab/HexSE).
3.2 Introduction

Overlapping reading frames (OvRFs) are portions of the genome where the same nucleotide sequence encodes more than one protein. They have been documented across all taxonomic domains including bacteria, vertebrates and fungi [2–5]. OvRFs are particularly abundant in virus genomes and examples can be found in all Baltimore classes [6]. In viruses, OvRFs may provide a mechanism to store more information in smaller genomes, to make purifying selection more efficient at removing deleterious mutations, or to facilitate the de novo creation of genes [7–11]. OvRFs can be classified into six different frameshifts that can be annotated as +2, +1, +0, −0, −1, −2 [12], although other labeling schemes exist. Here, the sign indicates whether the overlap occurs between open reading frames (ORFs) on the same (+) or the opposite (−) strands, and the integer indicates how many nucleotides they are shifted relative to one another. Frameshifts influence the effect of selection in OvRFs. For example, a +0 frameshift can amplify selection on codons that become expressed in multiple contexts without changing the number of nucleotides under selection. However, measuring selection within OvRFs is a difficult problem because the effect of a nucleotide substitution (i.e., whether synonymous or non-synonymous) depends on multiple codon contexts [13].

Computer simulations of biological processes are widely used as a tool to characterize biological processes that are otherwise too complex to represent as a mathematical model for analysis [14]. There is a great diversity of programs that simulate molecular evolution along a phylogeny [15], which are designed to model different aspects of evolution including recombination [16], insertions and deletions [17], variable selection intensities across the sequence [18], and different substitution biases [19]. However, we have not found a publicly available program that can simulate evolution in linear or circularized genomes with an arbitrary distribution of overlapping and non-overlapping genes and variable codon-specific selection pressures, yielding a multiple sequence alignment for a given tree. A standard simplifying assumption in molecular evolution is that nucleotides or codon substitutions are independent and identically distributed outcomes of a
3.3. **Implementation**

continuous-time Markov model. However, a codon within an OvRF is no longer independently evolving, since a nucleotide substitution can change the selective context for subsequent changes at nucleotides of adjacent codons.

Here, we describe a simulation method (HexSE) implemented in Python that simulates molecular evolution along an input tree where the sequence may contain any number of OvRFs. We employ a memory-efficient data structure to track the rates of substitution events at every individual nucleotide of an evolving sequence.

### 3.3 Implementation

#### 3.3.1 Input specification

HexSE takes three input files. First, it requires a FASTA- or Genbank-formatted file containing a nucleotide sequence (minimum 9 nt) to seed the simulation at the root of the tree. Next, the user must provide a configuration file in the form of a YAML-formatted file specifying the location of each ORF, including the parameters of the distribution that will be used to sample the frequency of mutations for each gene. Lastly, HexSE requires a file containing the Newick serialization of a phylogenetic tree. The tree must be rooted, *i.e.*, contain a root node to assign the input sequence. In addition, the tree must contain branch lengths. These lengths can be rescaled by the user by adjusting the global mutation rate parameter. For example, the program will simulate 0.01 substitution events per site on average along a branch of length 1 if the global mutation rate is set to 0.01 (notwithstanding other rate adjustments such as selection).

**Configuration file**

Additional parameters that may be specified in the YAML-formatted control file include: whether the genome is circular or not (default: false); a global transition/transversion ratio $\kappa$ (default: 0.3); the parameters and number of rate categories for discretized gamma or lognormal distributions, which are used to model variation in selection ($\omega$) and mutation rates ($\mu$) for each reading frame;
and the stationary nucleotide frequencies \( \pi \). By default, \( \pi \) is set to the empirical frequencies in the root sequence.

Figure 3.1: **Pipeline overview.** We use an exact stochastic simulation algorithm to simulate the accumulation of nucleotide substitutions in the selective context of one or more open reading frames, some of which may overlap. The simulation is initialized with an input sequence at the root (left panel), and traverses branches of the phylogenetic tree (centre panel) in order to generate a nucleotide alignment at the tips (right panel).

### 3.3.2 Algorithm

To simulate sequence evolution, we use a standard Gillespie [20] algorithm for the exact stochastic simulation of discrete events (nucleotide substitutions). The model is initialized by assigning the input sequence to the root of the tree and calculating the rates of every possible substitution at position \( i \) to nucleotide \( j \):

\[
\lambda_{ij} = \begin{cases} 
\mu_i \pi_j & \text{for a completely synonymous or non-coding transversion,} \\
\mu_i \kappa \pi_j & \text{for a completely synonymous or non-coding transition,} \\
\mu_i \omega \pi_j & \text{for a nonsynonymous transversion,} \\
\mu_i \omega \kappa \pi_j & \text{for a nonsynonymous transition}
\end{cases}
\]

where \( \mu_i \) is the baseline mutation rate. The effect of selection is tracked by a vector \( \mathbf{w} \) corre-
3.3. **Implementation**

sponding to the reading frames \{-2, -1, -0, +0, +1, +2\}, such that \|w\| = 6. The \(i\)-th element of \(w\) is set to \(w_i = 1\) if the substitution is synonymous in that reading frame. If \(w_i > 1\), non-synonymous mutations accumulate more rapidly on average than synonymous mutations, suggesting that the codon is under positive diversifying selection. Conversely, if \(w_i < 1\), non-synonymous mutations accumulate at a lower rate, indicating that the site is under negative (purifying) selection. In the absence of empirical information on how selective effects in overlapping reading frames combine, we assume that these effects are multiplicative. Therefore, the total selective effect on a substitution in HexSE is \(\omega = \prod w\). If a substitution is in a non-coding region or synonymous in all six reading frames, then \(\omega = 1\), i.e., the substitution evolves neutrally.

Next, we draw an exponentially-distributed waiting time to the next event, \(t \sim \exp(-\Lambda)\), where \(\Lambda = \sum_{ij} \lambda_{ij}\). If \(t\) exceeds the length of the current branch, no event occurs and the current sequence is propagated to the next node. Otherwise, a substitution event is drawn with a probability proportional to the associated rate, \(\lambda_{ij}/\Lambda\), and the sequence is updated by that event. Ultimately, the substitution rates are re-calculated for the updated sequence. This process continues by level-order traversal of the phylogenetic tree until every terminal node (tip) of the tree carries an evolved sequence.

### 3.3.3 Simulating a mutation

Computing and storing the rates of every possible substitution is time-consuming, and selecting the next substitution event by drawing a uniform random number would require a linear search of the cumulative probabilities. Since we use a discretized probability distributions for \(\omega\) and \(\mu\), these rates can only assume a finite number of values. Therefore, substitution events can be rapidly selected by traversing a hierarchical data structure that we call an *event probability tree*. All possible substitution events are stored at the tips of the event tree (Figure 3.2). Starting at the root of the event tree, we select the target nucleotide \(j\) with probability \(\pi_j\), moving down to the respective node at the next level. Next, we select the starting nucleotide \(k\) with probability \(1/(1 + 2\kappa)\) if \(k \rightarrow j\) is a transition and \(\kappa/(1 + 2\kappa)\) otherwise. We select the mutational rate category
given the global rate distribution for $\mu$, followed by selecting a particular region of the genome. When initializing the run parameters, we divide the sequence into categories determined by the distribution of open reading frames. For example, we divide the nucleotide sequence depicted in Figure 3.2 into five regions: nucleotides occurring within ORF a only ($p_1$); ORF b only ($p_3$); the overlapping region between ORF b and ORF c ($p_4$); ORF c only ($p_5$), and; nucleotides that are not in any ORF ($p_2$).

The probability of selecting a region is proportional to the number of potential substitution events weighted by the net effect of selection, which is determined by the rate categories (distributions) of $\omega$ associated with every ORF in the region. The weighted sum is calculated as $\sum_k \omega_k N_k$, where $N_k$ is the number of nucleotide sites associated with the $k$-th combination of selection biases $\omega_k$, and the sum is computed over all such combinations in the region. For example, a substitution at position $i$ that is non-synonymous in ORFs a and b with reading frames $-1$ and $+0$ can have a vector $w = \{1, \omega^a_3, 1, \omega^b_2, 1, 1\}$, where $\omega^g_k$ is drawn from a discretized distribution associated with ORF (gene) $g$. The value of $\omega_k$ at position $i$ results from multiplying the values assigned to $\omega^a_3$ and $\omega^b_2$.

Finally, we choose one of the $\omega$ combinations in the region, with a probability determined by the combination of selection rate biases. If a nucleotide is involved in a ‘start’ or ‘stop’ codon, then we do not include it in the event tree to prevent mutations from disrupting open reading frames.

Traversing the event probability tree from the root to a tip resolves the shared characteristics for a subset of substitution events. In other words, the tip stores references to the nucleotide substitution events that have the same probability of occurring. From this subset we can simply select an event uniformly at random, and update the evolving sequence accordingly. Since the composition of event subsets at the tips of the tree are updated with the sequence, we perform a deep copy of the event probability tree at every internal node of the phylogeny.
3.3. Implementation

Figure 3.2: **Traversal of an event probability tree to select mutations.** An event probability tree is a data structure that we used to store nucleotide substitutions with the same probability. Each level of the event tree corresponds to an evolutionary parameter, such as the transition/transversion rate bias. Each branch represents a discrete value associated with the parameter represented by that level. Traversing the event tree from the root (left panel) selects progressively smaller subsets of mutations, as demonstrated in the right panel.
3.3.4 Simulating evolution in the HBV genome

To demonstrate the usage of HexSE, we simulated evolution in the hepatitis B virus (HBV) genome, where around 30% of nucleotides within ORFs are involved in overlaps. Our inputs for the simulation were: the HBV reference genome (accession NC_003977.2) in Genbank format including coding sequence annotations for genes S, P, C and X; a random phylogenetic tree with 100 tips, using the implementation of the Kuhner and Felsenstein [21] algorithm in T-REX [22]; and the substitution model parameters. These parameters included a global mutation rate of 0.05, two rate categories for $\mu$ drawn from a lognormal distribution with two classes and shape 1.0 (resulting in rates 0.026 and 1.39 substitutions per nucleotide per unit time), a transition-transversion rate ratio of 0.3, and varying distribution parameters for $\omega$ in the four open reading frames. A second run was performed under the same parameters, but excluding all open reading frames except gene C (nucleotide coordinates 1,816 to 2,454).

Levels of genetic variation in simulation outputs varied substantially among codon sites, depending on their protein-coding context. We quantified these changes in genetic diversity using the Shannon entropy, $H = -\sum_{i=1}^{M} P_i \log_2 P_i$, where $M$ is the number of residues, and $P_i$ is the relative frequency of the $i$-th residue in the alignment column. This statistic provides a generic measure of diversity that applies to any combination of reading frames. We calculated the mean nucleotide entropy for sliding windows along the alignment with a width of 20 nucleotides and a step size of 1 nucleotide. For nucleotide residues, this statistic varies from zero, for a completely conserved site, to a maximum of $H = 2$.

A comparison of mean nucleotide entropy profiles between two simulation outputs on HBV sequences with one and four ORFs, respectively, is shown in Figure [3.3]. In the simulation results where we included only one gene, we detected higher entropy in the non-coding regions (mean 0.539, IQR 0.475–0.598) which is consistent with the net effect of purifying selection in the coding region of gene C (mean 0.302, IQR 0.254–0.342). We observe similar mutation patterns in the non-overlapping region of gene C, on the alignment produced when all four ORFs are included. This
Figure 3.3: **Simulating Evolution on HBV.** Using HexSE, we simulated evolution in the HBV reference genome, with all four ORFs (blue) or only one ORF (orange). We calculated the nucleotide entropy per site for the resulting alignments to account for the number of mutations accumulated on each position over time. The region encoding gene C has similar entropy values for both simulations. However, in the simulation with all ORFs, the mean entropy falls from a mean of 0.306 to 0.222 in the overlapping region (nucleotides 2,309 to 2,454). There, mutations rarely accumulate, even in comparison with adjacent regions, which are also protein-coding regions. In general, overlapping sites tend to accumulate less mutations than non-overlapping sites of the genome. This pattern is the most evident in the transition from the non-overlapping regions of gene P (nucleotides 837 to 1,376 and 2,454 to 2,850) to the region overlapping gene S (nucleotides 2,850 to 837).

region, comprised between nucleotides 1,840 and 2,309, has a mean of 0.306 (IQR 0.247–0.366), and reaches a minimum of 0.104. As expected, we noticed a significant differences in the entropy levels between overlapping and non-overlapping regions (average 0.304 vs 0.397; Wilcoxon rank sum test, $P < 10^{-15}$). These shifts in mean entropy are also confounded by variation among ORFs in the intensity of purifying selection, as well as stochastic variation. These patterns are consistent
with the cumulative effect of purifying selection in one or more ORFs on limiting diversification at the level of nucleotide sequences.

The average time that HexSE requires to simulate evolution on HBV under our example parameters was 52.74 seconds (interquartile range, IQR: 48.36–58.01s, \(n = 20\)) using four cores on an AMD Ryzen 5 3400g processor, with Python 3.6.9 and Ubuntu 18.04.6. In contrast, the same simulation took about 310.3 seconds (IQR 281.3–308.1) to complete on a tree with 500 tips. In general, running times scaled linearly with respect to the number of tips on the phylogenetic tree (Figure 3.5).

### 3.3.5 Accounting for variation in selection intensities

To assess HexSE’s ability to generate realistic outputs, we compared selection pressures on HBV gene C between two independent HexSE runs and a curated multiple sequence alignment of 170 complete genomes of HBV genotype F [23]. We used the alignment of actual HBV sequences to reconstruct a maximum-likelihood phylogenetic tree using FastTree (version 2.1.10) [24]. Both HexSE runs were initialized with the resulting tree and the consensus sequence of the alignment. We used the accessory script `gb_to_yam1.py` to configure the run specification files based on the annotation of the HBV genome sequence KY382417.1. Thus, both configuration files included the information required to simulate selection in the four open reading frames that encode proteins S, P, X and C in HBV. The global mutation rate was set to 6, and the transition-tranversion rate ratio (\(\kappa\)) to 0.3, for both runs.

To parameterize \(\omega\) in HexSE, users can modify the shape (\(\alpha\)) and scale (\(\beta\)) parameters of the gamma distribution, considering that the mean of this distribution is \(\alpha/\beta\). For the first run, for example, we set these parameters to \(\alpha = 2\) and \(\beta = 0.5\), respectively, and set the number of categories to four in the open reading frame that encodes gene C. The resulting mean is 1 and the values associated with the four equiprobable categories are 0.293, 0.655, 1.069, and 1.981. Hence, only one quarter of sites in the open reading frame would experience positive selection. We followed the same approach for genes X, P and S.
3.3. IMPLEMENTATION

In contrast, the second run was parameterized by using FUBAR [25] to estimate dN and dS values for the real alignment. In HBV genotype F, the open reading frame that encodes gene C (reference coordinates 1,813 to 2,452) overlaps in 25 nucleotides with gene X (coordinates 1,373 to 1838), and in 146 nucleotides with gene P (coordinates 2,306 to 3,215; 0 to 1,623). We fitted a gamma distribution to the selection patterns in the codon sequences of genes X, P, and C, and run the second simulation based on these empirical estimates. Additionally, we introduced variation to the mutation intensities by drawing mutation rates ($\mu$) from a gamma distribution with $\alpha = 2$ and $\beta = 1$ (resulting in four categories of 1.8, 3.1, 4.3, and 6.7). As expected, the distribution of dN/dS values for the second simulation run was more similar to the actual data ($\hat{\alpha} = 0.85$ and $\hat{\beta} = 0.85$; Figure 3.4). Because our simulations constrained dS to 1, however, we also observed a lack of variation in dS values compared to real data that was only partially compensated by adding variation in mutation rates.
Figure 3.4: **Comparison of selection pressures on between gene C of HBV genotype F and two simulated alignments.** We used the FUBAR (Fast Unconstrained Bayesian Approximation) method, to estimate selection intensities in the genome region comprised between nucleotides 1,813 and 2,452 of a real alignment of HBV genotype F, and two independent HexSE simulations. For the first simulation, we selected values for the shape and scale parameters based on the uninformed assumption that the majority of codon sites were under neutral or purifying selection. In contrast, we parameterized the second run based on the empirical estimates of dN/dS calculated on the codon sequences of genes X, P and C.
3.4 Acknowledgements

This work was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC Discovery Grant RGPIN 05516-2018).

3.5 Supporting Information

Figure 3.5: Measuring HexSE computing times. Elapsed times for 20 replicates of HexSE simulation runs on the HBV genome (accession NC_003977.2) using random phylogenetic trees with 50, 100, 200 and 500 tips. All simulations used the same parameters described in our example: global mutation rate of 0.05, two rate categories for $\mu$ drawn from a lognormal distribution with two classes and shape 1.0, and varying distribution parameters for $\omega$ in seven open reading frames.
Bibliography


Chapter 4

Detecting OvRFs from changes in $dS$ estimates

As extensively discussed in Chapter 1, estimating selection in coding sequences (CDSs) with overlapping reading frames (OvRFs) is a complex task due to the potential impact of mutations on multiple proteins. Specifically, mutations that are synonymous in one open reading frame (ORF) can be nonsynonymous in the overlapping coding sequence, and vice versa. Consequently, purifying selection on these mutations in alternate reading frames can cause us to underestimate the rate at which synonymous mutations accumulate over time ($dS$) in overlapping genes.

To characterize the variation in selection explained by OvRFs, in this Chapter we utilize reductions in $dS$ estimates along a coding sequence as an indication of the presence of an overlap in the hepatitis B virus (HBV). Our particular focus lies on gene P, which is the largest ORF in HBV and exhibits overlaps with adjacent genes S, X, and C. We use HexSE to simulate evolution in gene P with and without overlap, and we used reductions in $dS$, identified with a change detection method, as an indication of the presence of an overlap. Importantly, we observed that the accuracy of the change detection method heavily relies on the degree of divergence between the sequences. For instance, the average true positives rate drops from 1 to 0.4 when the global mutation rate is lowered from 1.7 to 0.5.
4.1 Introduction

The prevalence of synonymous substitutions has been used as the baseline for neutral evolution. Since they do not alter the amino acid sequence, they are expected to accumulate in genomes by chance \[1\]. In other words, these synonymous mutations are not expected to be either positively or negatively selected by natural selection. Of course, there are exceptions to this assumption, such as the effects of synonymous mutations on gene expression or the secondary structure of RNA virus genomes \[2\]. Nevertheless, these provide a convenient baseline for estimating the effect of selection on protein coding sequences \[3\].

Since the early development of codon models, researchers have observed that the presence of overlapping reading frames (OvRFs) influences the rate of accumulation of synonymous mutations \((dS)\) in virus genomes. For example, in 1980 Miyata and Yasugana \[4\] compared the sequences of the *Escherichia coli* bacteriophages G4 and φX174 to investigate the rate of amino acid changes. They found that gene A, which overlaps with gene B and K, exhibited lower \(dS\) values (0.505 and 0.301) compared to the non-overlapping regions (0.68, 0.72, 0.74 and 0.68 for genes F, G, H and non-overlapping region of A, respectively). Their research suggested that — given that synonymous substitutions in the parental gene tend to be nonsynonymous in the overlapping pair, and nonsynonymous mutations that change the amino acid residue reflect the functional constraints operating on the amino acid sequence — overlapping sequences will tend to accumulate synonymous substitutions at a lower rate than the rest of the genome.

The effect of reductions in \(dS\) in overlapping reading frames has a significant impact on selection estimates when studying virus evolution. This is evident in the case of the PB1-F2 gene of the influenza A virus (IAV), where a strong signal of positive (or diversifying) selection \((dN/dS = 9.36)\) was initially reported in 2006 \[5\]. However, this signal was later revealed to be an artifact caused by the reduction of synonymous mutations due to an overlap in the +1 frameshift with the PB2 gene \[6\]. In other words, the reduced \(dS\) values skewed the ratio estimates \((dN/dS)\) of selective effects upwards.
Interestingly, the amino acid residues encoded in overlapping regions are not more conserved than those encoded in non-overlapping regions [7]. Instead, overlapping regions are hypothesized to be utilized by viruses to amplify the deleterious effects of mutations, ultimately leading to their removal from the population. This phenomenon was demonstrated by Fernandez and collaborators [8], who investigated the constraints imposed in the regulatory genes tat and rev by their overlap with the envelope gene env. To elucidate the effects of the tat/rev overlap on virus evolution, the researchers modified the HIV-1 genome to encode the same proteins without overlaps and conducted mutational analyses. They evaluated the differences in fitness among the amino acid residues of each protein, and discovered that the presence of overlaps reduced the probability of generating an unfit combination of the overlapping gene pair.

The hepatitis B virus (HBV) also presents an intriguing case of gene overlap that has been extensively studied. This virus possesses a circular, partially double-stranded DNA genome of approximately 3,200 base pairs, consisting of four open reading frames (ORFs) encoding proteins P (polymerase, responsible for reverse transcription), C (core, forming the capsid), S (surface, enveloping the capsid), and X (regulatory protein, facilitating replication through gene expression stimulation) [9–11]. Notably, 63% of the HBV genome exhibits overlaps: ORF S is fully contained within ORF P, while ORFs C and X overlap with P by 23% and 39% of their sequence lengths, respectively [12]. Early studies on HBV revealed that the presence of overlap constrained the evolution of the virus, leading to a reduced number of synonymous substitutions in gene P [7]. These conserved regions have later been hypothesized to be crucial for maintaining essential protein functions [13, 14]. Moreover, overlapping regions in HBV complicates the estimation of its evolutionary history, with divergence estimates differing significantly between overlapping and non-overlapping regions [15].

The objective of this chapter is to investigate the impact of overlapping genes on selection estimates by employing a change detection method capable of identifying alterations in $dS$ when overlaps are present. A change detection method provides an objective, statistical model to determine whether a system has changed its internal state, by examining its outputs over space or time.
We use HexSE to simulate evolution in HBV genomes with and without overlaps, by specifying, or omitting, the location and parameters that define selection in genes S and P. We then applied a change detection method to the simulated alignments and evaluated whether the method identified a change point — points with a significant shift in $dS$ — in each replicate alignment. Results with zero change points as an option were interpreted as no-overlaps. In contrast, results with at least one change points in every optimal solution were interpreted as overlaps. We then assessed the performance of this method by comparing these predictions to the known simulation settings. Our results show that change detection method can predict the presence of an overlap, but it is highly sensitive to mutation rates. Thus, the accuracy of the method increases with the extent of sequence divergence in the alignment, and decreases with low substitution rates due to purifying selection.

### 4.2 Methods

#### 4.2.1 Estimating selection in a real HBV dataset

Although HexSE can generate a sequence alignment under a wide range of parameter settings (see Chapter 3), we wanted to simulate HBV alignments that were as close as possible to realistic genome sequences. To establish the baseline parameters for our simulation runs, we initially estimated the distribution of selection effects across sites using an alignment of 508 curated nucleotide sequences from HBV genotype A [16]. From this alignment, we reconstructed a maximum likelihood phylogenetic tree using FastTree (version 2.1.10) [17]. Subsequently, we employed a Python script to extract all the coding sequences (CDSs) based on the annotation of genome AB205118 for genes S, X, C, and P.

Next, we generated a new phylogenetic tree specifically for the CDS encoding gene P. To estimate the $dS$ and $dN$ values in each codon, we used the FUBAR (Fast Unconstrained Bayesian Approximation) method [3] in the HyPhy software [18]. Rather than attempting to estimate both rates at every codon site in the alignment, FUBAR defines a grid of 20 $dS$ and 20 $dN$ values, and infers the bivariate probability distribution over the resulting 400 combinations of these values.
4.2. Methods

Figure 4.1: Obtaining parameters for HexSE runs from non-overlapping regions of gene P.

HBV is a circular genome that encodes four proteins: P, S, X, and C. To parameterize our HexSE simulation runs from empirical estimates, we used the FUBAR [3] results for $dN$ and $dS$ on gene P to obtain the ‘shape’ and ‘scale’ parameters of a gamma distribution fitted to the estimates on the non-overlapping region (as highlighted in the genome diagram at the top right of the figure).

In HyPhy, the estimated values of $dS$ and $dN$ in every codon are denoted as ‘alpha’ and ‘beta’, respectively. Finally, we used the ‘fitdistr’ function from the MASS package in R (version 3.6.3) [19, 20] to fit a gamma distribution to the non-overlapping regions, from which we obtained the ‘shape’ and ‘scale’ parameters that define the gamma distributions that we use in HexSE to draw the $dN$ and $dS$ values for the simulation (Figure 4.1).

4.2.2 Running HexSE simulation replicates under different coding contexts

To evaluate our ability to predict the presence of an OvRF based on shifts in selection estimates, we conducted 100 HexSE replicate simulations for seven different coding contexts. In all simulations, the stationary nucleotide frequency $\pi$ for all nucleotides was set to 0.25, the transition-transversion
rate ratio $\kappa$ was fixed at 0.3, and the mutation rate $\mu$ was initially set to 1 for consistency. Additionally, we drew eight values of $dN$ and eight values of $dS$ from gamma distributions, using different shape and scale parameters as outlined in Table 4.1.

No overlap scenario: In the first set of simulations, we specified the location and selection parameters only for gene P, which is annotated in the reference HBV genome AB205118 to be encoded from nucleotide positions 2,306 to 3,221 and 0 to 1,623. In other words, this gene spans the origin of the numbering system for this circular genome. These 100 replicates were designed to represent a scenario where gene P has no overlaps with other genes.

Varying sequence divergence: Next, we simulated evolution at three different mutation rates, controlled by the ‘global rate’ parameter in HexSE. To determine the mutation rate that best resembled the real alignment, we simulated evolution at multiple global rates and measured the lengths of the trees created from each alignment with FastTree. We found that a global mutation rate of 1.7 produced phylogenetic trees with lengths (measured in expected numbers of substitutions per site) most similar to the real data. Hence, we used a global mutation rate of 1.7 to create an alignment with levels of diversity similar to reality. Additionally, to assess the performance of the detection method at lower levels of sequence divergence, we also simulated evolution at global rates of 0.5 and 1.0.

Varying selection intensities: To simulate changes in selection pressures in gene S, we varied the parameters of gamma distributions that describe the rate variation in $dN$, while keeping the distribution for $dS$ centered around 1. In the simulations representing neutral evolution of gene S, the $dN$ values were centered around the mean of 1. For simulations representing purifying selection, the mean of $dN$ was set to 0.5, and for simulations representing diversifying selection, the mean of $dN$ was set to 2.5 by changing the shape and scale parameters that define the gamma distribution from where we draw the $dN$ rates on HexSE (Table 4.1).
### Table 4.1: Simulation parameters for HexSE run replicates

<table>
<thead>
<tr>
<th>Run name</th>
<th>global rate</th>
<th>parameter</th>
<th>gene P</th>
<th>gene S</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>p only</strong></td>
<td>1.7</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>-</td>
</tr>
<tr>
<td><strong>reference</strong></td>
<td>1.7</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>1.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>1.15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>3.38</td>
</tr>
<tr>
<td><strong>low divergence</strong></td>
<td>0.5</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>1.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>1.15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>3.38</td>
</tr>
<tr>
<td><strong>medium divergence</strong></td>
<td>1.0</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>1.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>1.15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>3.38</td>
</tr>
<tr>
<td><strong>neutral</strong></td>
<td>1.7</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>1</td>
</tr>
<tr>
<td><strong>purifying</strong></td>
<td>1.7</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>0.1</td>
</tr>
<tr>
<td><strong>diversifying</strong></td>
<td>1.7</td>
<td>shape $dN$</td>
<td>1.03</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dN$</td>
<td>0.56</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shape $dS$</td>
<td>1.15</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scale $dS$</td>
<td>3.38</td>
<td>0.1</td>
</tr>
</tbody>
</table>
4.2.3 Using change points in selection estimates as signals of OvRFs

A brief review of change point detection

Change points occur when there is an abrupt or significant shift in a series of observations over time or space [21]. In this context, ‘significance’ means that the shift exceeds what we would expect to occur by chance. These shifts are generally based on some summary statistic such as measures of central tendency, i.e., mean or median. The points where the shifts occur can be detected using change detection methods that identify changes in the summary statistic for one or more variables, partitioning the observed series into segments [22, 23]. For a time series with \( n \) data points, denoted as \( y_{1:n} = (y_1, \ldots, y_n) \), we may place \( m \) change points at positions \( \tau_{1:m} = (\tau_1, \ldots, \tau_m) \) — where \( \tau_0 = 0 \) and \( \tau_{m+1} = 1 \) — to divide the data into \( m + 1 \) segments. An optimal distribution of change points may be determined by defining segment-specific cost function \( C(y_{s+1:r}) \) for each segment with the data points \( (y_{s+1:r}) \), and then summing the cost functions over the \( m + 1 \) segments. Then, the number and locations of \( m \) change points would be determined by minimizing the total cost over all the segments:

\[
Q_m(y_{1:n}) = \min_{\tau_{1:m}} \left\{ \sum_{i=1}^{m+1} C(y_{(\tau_{i-1}+1):\tau_i}) \right\}.
\]  

(4.1)

The problem with this approach is that the total cost will usually be reduced simply by adding more segments. As a result, there has to be some way of penalizing the algorithm for the number of segments. In other words, Equation 4.1 can be incorporated into a composite cost function that includes a penalty term that increases with \( m \) [21]:

\[
\min_m \left\{ Q_m(y_{1:n}) + f(m) \right\}
\]

(4.2)
4.2. Methods

PELT and CROPS

Suppose that we define $f(m)$ to be a linearly increasing function of $m$, that is, $f(m) = \beta(m + 1)$ where $\beta > 0$. This assumption transforms Equation 4.2 into a penalized minimization problem — a class of optimization problems for which there are efficient numerical methods available [24–26]. In other words, for known values of the linear penalty $\beta$, the cost function becomes:

$$Q(y_{1:n}, \beta) = \min_{m, \tau_1:n} \left\{ \sum_{i=1}^{m+1} [C(y_{\tau_{i-1}+1:\tau_i} + \beta) + C(y_{\tau_i+1;n})] \right\}$$  \hspace{1cm} (4.3)

which can be solved for the optimal locations of $m$ change points.

To enhance the computational efficiency for solving Equation 4.3, Rebecca Killick and colleagues introduced the pruned exact linear time (PELT) method [27]. PELT sequentially discards the values of $\tau$ for which no minimum exists. However, PELT relies on a priori knowledge of the penalty value $\beta$. Consequently, if the same method is evaluated using different penalties, it may lead to the identification of different change points.

To address the challenge of finding the optimal segmentation for Equation 4.3 without prior knowledge of the optimal $\beta$ value, Kaylea Haynes and colleagues developed CROPS (change points for a range of penalties) [21]. CROPS attempts to calculate all optimal segmentations for varying values of the penalty $\beta$ penalty within a user-defined range ($\beta_{\text{min}}, \beta_{\text{max}}$). This algorithm yields a matrix of change points over the range of penalty values. The user is required to visually inspect this output to identify the most suitable setting of $\beta$ for explaining the changes in the data. Although CROPS was proposed as a generic approach to evaluate different solutions to change point detection problems, it was originally used in conjunction with the PELT method [21].

Application to sequence alignments

Initially, we examined whether the change points in the rate of synonymous mutation accumulation ($dS$) were associated with the overlapping regions of gene P in the HBV genome. Since the selection estimates on gene P were skewed to values between 0 and 2 (Figure 4.1), we performed a
log transformation to normalize them. We used CROPS to estimate change points using penalties ranging from 1 to 25, applied to the log-transformed $dS$ estimates. Our analysis included a real dataset of HBV and three simulated alignments: one with specifications for all four ORFs, another with coordinates solely for genes P and S, and a third with only gene P specified (see Figure 4.3) as a negative control.

To assess the change detection ability in accurately predicting overlaps, we conducted simulations with two specifications: (1) gene P only, and (2) genes P and S, as described in Section 4.2.2. Using the FUBAR estimates of $dS$ in gene P, we classified the results into the following categories:

**True positive (TP):** if change detection recognized that there is at least one changepoint in the simulations where S and P were specified.

**True negative (TN):** if change detection identified the possibility of zero change points in the simulations where only P was specified.

**False positive (FP):** if change detection recognized that there is at least one change point in the simulations where only P was specified.

**False negative (FN):** if change detection identified the possibility of zero change points in simulations where S and P were specified.

We calculated the frequency of these conditions for multiple change detection inferences by varying the range of penalty values, specifically by increasing the end value of the penalty range. The start of the ranges was set to 1, while the end values ranged from 10 to 30. To evaluate our ability to accurately classify an overlapping sequence based on the detection of changepoints from selection estimates, we compared the sensitivity (true positive rate, $TPR = TP/(TP + FN)$) of the change detection method against the false positive rate ($FPR = FP/(FP + TN)$) for different end values of the penalty ranges.
4.3 Results and Discussion

Reduced amounts of synonymous substitution rates (\(dS\)) in overlapping regions have been previously reported for viruses like human immunodeficiency virus (HIV) type 1, \(\phi X174\), influenza A virus (IAV), and HBV \([6-8, 28]\). This is assumed to be a consequence of purifying selection in additional open reading frames. However, this effect has yet not been systematically characterized, largely because there has not been a way to simulate evolution in overlapping reading frames with varying rates of mutation and selection among nucleotide sites.

In this study, we used our simulation program HexSE to evaluate whether a conventional change detection method (PELT with CROPS) could classify regions of simulated genomes as overlapping or non-overlapping, based on varying trends of \(dS\) across a reference coding region. From our initial assessment of the ability of change detection to identify overlapping regions based on shifts in the log-transformed values of \(dS\) along the 836 codons of gene P on HBV, we noticed that change points were associated with regions where the overlap started or ended (Figure 4.2). In particular, we noticed that the signal imposed by the presence of gene S was consistently detected in multiple simulations, similar to the signals that we detected from a real HBV alignment.

In the case of the real HBV gene P alignment, the PELT and CROPS methods obtained optimal segmentations at 26 different penalty values. The locations of a subset of the change points were robust to varying penalty values, and generally coincided with the boundaries of overlapping regions (Figure 4.3). The number of optimal segmentations at given penalty values decreased to 13, 4, and 1 for simulations involving four (P, C, S and X), two (P and S), and one ORFs (only P), respectively. In other words, the number of optimal change point placements declined with the number of open reading frames and overlapping regions in the simulated alignments. In addition, we observed that the locations of the change points were concordant with the start and end locations of the overlapping regions. In Figure 4.3 these regions are highlighted in pink for the overlap with gene C, blue for the overlap with gene S, and purple for the overlap with gene X.

To evaluate the robustness of the change detection method to varying conditions, we first mea-
Figure 4.2: **Change point detection in $dS$ estimates on gene P of HBV under varying coding contexts.** Our initial assessment of the change detection method involved evaluating its ability to recover similar change points as those detected in real data. To accomplish this, we compared the change detection method applied to $dS$ rates in the P gene between an alignment of 508 real HBV genotype A sequences, and three HexSE-simulated alignments created under three different scenarios: a genome containing genes P, S, and X; a genome containing genes P and S; and a genome containing only the P gene. Shaded areas in pink, blue, and purple, represent the areas where gene P overlaps with C, X, and S, respectively. Note that the coordinate system is specific to the P gene (codons numbered 1 through 846).

We measured the effects of mutation rates on the classification accuracy. We simulated evolution within the P gene by itself, and with an overlap between P and S genes, under global mutation rates of 0.5, 1.0, and 1.7 nucleotide substitutions per site per unit of time. Estimates of the false positive rates for all three scenarios were below 0.06, indicating that the change detection method rarely places change points under all penalty values in simulated alignments where only gene P was specified (Figure 4.4, *left*). Furthermore, we observed that the true positive rate decreases substantially as the range of penalties expands (Figure 4.4, *right*). This is caused by forcing the CROPS algorithm to solve for optimal segmentations under excessive penalty values, inevitably resulting in one-segment so-
4.3. Results and Discussion

lutions, *i.e.*, no change points. Since we used the conservative criterion that a change point analysis was ‘negative’ if *any* of the optimal solutions consisted of zero change points, expanding the range of penalties drove down the true positive rate.

Additionally, the change detection method is highly sensitive to the level of sequence diversity (Figure 4.4, right). For simulations with low mutation rates (0.5), the highest true positive rate estimates were less than 0.4. In other words, less than half of the simulated alignments containing overlaps were correctly classified. In contrast, the method is quite sensitive (*TPR* = 1) when classifying alignments with overlaps (true positives) at a global mutation rate of 1.7 (which generates

Figure 4.3: Changepoints detected under different penalty values evaluated with CROPS. Under the CROPS method, PELT evaluates multiple penalty values for optimal changepoint detection. This figure illustrates the detected changepoints in estimates of *dS* for gene P in a real HBV alignment and three simulated alignments, each representing different coding contexts. The penalties were set to vary in a range from 1 to 25. Shaded areas in pink, blue, and purple, represent the areas where gene P overlaps with C, X, and S, respectively. The lighter the colour of the point, the higher the penalty value under which it was detected.
Figure 4.4: **Effect of sequence divergence in the classification of overlapping regions.** We analyzed the change detection ability to detect the presence of an OvRF in simulations created under global mutation rates of 0.5, 1.0, and 1.7. (left) True positive rate (TPR) versus false negative rate (FPR) for different global mutation rates. (right) True positive rate (TPR) with respect to the end of the penalty range used to evaluate CROPS. We found that change detection method is very sensitive to sequence divergence, and that, as we expand the penalty range, the CROPS method starts to evaluate stringent penalty values that accommodate for the possibility of no changepoints in the data despite the presence of an overlap.

Having determined that the overall mutation rate has a measurable impact on the performance of the change detection method, we were subsequently interested in exploring the effect of selection in the overlapping gene. We modulated selection by varying the rates of nonsynonymous mutations ($dN$) while keeping the distribution of synonymous mutations ($dS$) centered at 1. For simplicity, we compared alignments where both genes S and P are present (positives) against alignments where only P is present (negatives). In our HexSE simulations, we adjusted the parameters for gene S to represent three selection patterns (see Figure 4.5) to represent:

- **Neutral selection:** In this scenario, the rates of synonymous and nonsynonymous mutations are expected to accumulate at a similar pace over time. To simulate this, we set the shape
4.3. Results and Discussion

Figure 4.5: **True positive rate when varying $dN$ parameters** True positive rates for penalty ranges for 100 replicate simulations run under different parameters for the gamma distributions from which $dN$ values are drawn and scale parameters for $dN$ to 1.

- Positive (diversifying) selection: Here, mutations that introduce changes in the amino acid sequences are expected to accumulate more frequently in the genome, promoting adaptation to different evolutionary contexts. We adjusted the scale and shape parameters for $dN$ to be 25 and 0.1, respectively, resulting in a distribution centered around 2.5.

- Negative (purifying) selection: This type of selection occurs when nonsynonymous mutations are removed from the population due to their detrimental effects. To resemble this scenario, we set the parameters such that the mean of $dN$ values was around 0.5.

We observed that the change detection method performed better in classifying alignments when the overlapping gene was subject to diversifying selection. Under this scenario, the true positive rate consistently approached 1 for penalty ranges up to 25 (green line in Figure 4.5). However, when the overlapping gene was subject to purifying selection, the change detection method failed to identify the presence of an overlap (red line in Figure 4.5). We note that varying selection had
no effect on FPR because the true negative simulations represented alignments where only the P gene was expressed — it is the distribution of selective effects on the second ORF that are being varied.

These results from varying selection are contrary to our expectation that stronger purifying selection acting on the second open reading frame (ORF, i.e., the S gene) will be more readily detectable as a shift in $dS$ rates within the reference ORF (i.e., the P gene). Our interpretation is that diversifying selection in the second ORF induces a readily detectable shift in $dS$ rates upwards of the baseline values. Since diversifying selection means that substitution rates are accelerated, there is a stronger signal for detecting this shift. In contrast, purifying selection in the second ORF will tend to deplete synonymous variation in the reference ORF. This loss of signal in the data appears to reduce the sensitivity of change detection.
Bibliography


Chapter 5

Concluding remarks

My desire to pursue a PhD was driven by my curiosity about evolution, particularly in viruses. Viruses are fascinating entities that have developed some of the most curious and innovative strategies to replicate despite the selective pressures imposed by their hosts’ immune responses. Interestingly, the study of viruses has been the source of evolutionary discoveries that were subsequently observed in cellular organisms. An example of such discoveries are the **overlapping reading frames (OvRFs)** — a unique gene arrangement in which the same nucleotide sequence encodes more than one protein, documented for the first time in the genome of the bacteriophage \( \phi X174 \) (the first organism with a fully sequenced genome). This strategy is hypothesized to increase the information content in virus genomes and to facilitate the creation of new genes. As I conclude my thesis, I would like to provide the keypoints that emerged from these years of using bioinformatic tools to study OvRFs in virus genomes.

5.1 Conclusions

**Prevalence of OvRFs in virus genomes**

To begin with, we evaluated the prevalence of OvRFs in 12,609 reference virus genomes from NCBI. From this study, we confirmed previous findings that overlapping genes are ubiquitous in virus genomes across all Baltimore classes. We observed that the majority of OvRFs in DNA
viruses are short, with a significant proportion being only 1 or 4 nucleotides in length. Additionally, we found that the overlap length decreases with genome length, supporting the compression theory that suggests that OvRFs play a role in reducing the genome size necessary to encode a given number of essential functions. We also demonstrated that different frameshifts involving OvRFs have divergent characteristics. For example, +2 frameshifts are predominantly found in dsDNA viruses, whereas +0 and +1 dominate in RNA viruses, where overlaps are longer and need to accommodate for more mutations without introducing stop signals.

**Network-based methods for the study of OvRFs**

In our pursuit of exploring the distribution of OvRFs at a finer level than Baltimore classes, we proposed to represent genomes within the same virus family as networks, *i.e.*, graphs. To achieve this, we constructed genome networks by clustering homologous proteins based on distance metrics derived from *k*-mer counts. Applying this method to families like Coronaviridae, Papillomaviridae, and Rhabdoviridae, we observed distinct characteristics in the composition of their genomes within each family. For instance, we discovered that the families Geminiviridae, Papillomaviridae, and Rhabdoviridae exhibit a higher number of overlaps among proteins with variable locations in different genomes (*P* < 0.006), whereas Coronaviridae genomes showed fewer overlaps (*P* = 4.9×10⁻⁹). This graph-based approach not only offers a natural framework for visualizing the distribution of OvRFs, but also paves the way for employing network-based statistics to estimate the likelihood of encountering OvRF in genomes within the same virus family.

**HexSE resembles selection patterns imposed by OvRFs**

OvRFs impact selection estimates because synonymous substitutions (that preserve the amino acid sequence) in one reading frame may be nonsynonymous (altering the encoded amino acid) in the second frame. We developed HexSE with the purpose of simulating molecular evolution in OvRFs under different selection intensities. The central novelty of our simulation program is the implementation of an ‘event probability tree’, a custom data structure that provides an efficient and
scaleable means of tracking and randomly selecting substitution events in evolving genomes. We parameterized HexSE with selection estimates obtained from real datasets of HBV, and recovered similar selection pressures across the coding regions when we compare FUBAR results. In other words, we can use HexSE to resemble the selection patterns that we observe in OvRFs.

**Detecting OvRFs based on selection patterns**

Lastly, we used our HexSE simulation method to evaluate the hypothesis that detecting decreasing signals on the rate of accumulation of synonymous mutations in overlapping regions can be used to detect OvRFs. In particular, we used a change detection method to identify shifts in the distributions of synonymous substitution rates, as measured by the normalized quantity $dS$, as we scan across the open reading frame of a ‘known’ gene. We found that change detection was highly effective at classifying simulated alignments in which a second, overlapping reading frame was either present or absent. In addition, we found that the performance of change detection was substantially affected by the amount of divergence among sequences in the simulated data, as modulated by the global mutation rate, and by the strength of purifying selection on the second overlapping reading frame.

In summary, my thesis presents valuable information about the prevalence and characteristics of overlapping reading frames in virus genomes, and contributes to our understanding of virus evolution by exploring alternative ways of assessing selection in these complex genetic features.

### 5.2 Limitations

**Presence of OvRFs based on sequence annotation:** Our analysis of the distribution of genes and overlapping genes in virus genomes depends heavily on the accurate annotation of these reference genome sequences in Genbank. For example, if a gene is not annotated, then it is not available for our clustering analysis as an amino acid sequence. The NCBI Virus
database is a manually curated resource that is meant to be comparable in quality to the RefSeq databases of other organisms, but we cannot rule out the possibility that some genomes are not completely annotated, or that CDS not accurately annotated.

**Prediction of OvRFs from network statistics:** Network-based statistics that relate to the probability of finding OvRFs vary among the genomes of different virus families. For example, only in the Papillomaviridae family the size of a cluster is associated with the probability of an overlap. Edge degree, on the other side, was positively correlated with an overlap in the Coronaviridae and negatively correlated on the Papilloma and the Rhabdoviridae family. However, this statistic was not informative for the Adenoviridae or the Geminiviridae family. Overall, the prediction of OvRFs seems to be a pattern that should be analyzed independently for different families when using a network-based approach.

**Simulating evolution under other complex scenarios:** HexSE can only simulate evolution based on mutations patterns. However, the evolutionary mechanisms that affect virus genomes are far more complicated. For example, viruses undergo recombination, gene shuffling, and insertion-deletion events. HexSE cannot account for any of these scenarios, which limits the analysis of OvRFs to their effect on selection pressures on genomes that conserve the same structure over time.

**Resembling selection pressures from real data:** We anticipate that the presence of OvRFs will introduce bias into the selection estimates. As a result, we are currently parameterizing our HexSE simulations using $dN$ and $dS$ values obtained from non-overlapping regions. In other words, our simulations are expected to resemble the selection patterns of specific segments of the genes. This issue becomes more pronounced in nested genes (such as gene S of HBV) where all codons are part of an overlap. Incorporating selection estimates from overlapping regions into the parameterization of our simulations presents a challenge that we have yet to address.

**Estimating overlaps from changepoint detection methods:** Our approach to using signals of changes
5.3. Future Directions

in $dS$ estimates across a coding sequence as an indication of an overlap, relies on identifying at least one changepoint, regardless of its location in the genome. Furthermore, we currently employ at least one penalty value resulting in no changepoints as an indication of the absence of an overlap. However, this method does not accurately reflect the CROPs results, as might be disregarding changepoints found under lower penalty values. This approach represents our initial attempt at utilizing change detection methods for the detection of OvRFs. I anticipate addressing these challenges and developing a more robust identification method in my future work.

5.3 Future Directions

**Expanding the change detection method:** Our initial investigation into selection pressures in OvRFs indicates that changes in selection intensities resulting from overlaps can be detected using change detection methods. This approach can be further expanded by employing machine learning techniques to analyze the distribution of $dN$ and $dS$ rate estimates in all six reading frames in order to detect OvRFs. Additionally, this method can be evaluated for other viruses that may exhibit distinct selection patterns within their overlapping regions.

**OvRFs at different levels of divergence:** In this research, our analysis focuses specifically on examining the impact of OvRFs on selection within a single HBV genotype, with a very structured genome. However, it is worth considering that a recently emerged OvRF may not undergo strong purifying selection as it may not yet confer an evolutionary advantage for the virus (such as newly created accessory gene). Exploring virus dataset of closely versus distantly related species could provide new insights into how OvRFs evolve in virus genomes.

**OvRFs and recombination:** Are there hotspot associated with overlapping reading frames? Considering that recombination events in OvRFs can potentially disrupt multiple proteins, it would be interesting to investigate the impact of OvRFs on the recombination rates of various virus species.
**Experimental validation of evolutionary hypothesis:** Experimental research on HIV has indicated that, in the absence of overlaps, the fitness of the proteins they encode is diminished. It would be interesting to conduct similar experiments to assess the variation in selection pressures imposed by overlaps, which would be possible by designing genomes that encode the same proteins without overlaps.
Education

Western University
PhD in Microbiology & Immunology
Thesis: Evolution of Overlapping Reading Frames in virus genomes

Universidad Nacional de Colombia
M.Sc in Biotechnology
Thesis: Development of a bioinformatic pipeline for the detection and characterization of plant viruses using Next Generation Sequencing (NGS) data

Universidad Nacional de Colombia
B.S. In Biological Engineering
Thesis: Molecular characterization of Potyvirus causing viral diseases on Solanaceae plants in the Colombian Andes

Publications


Presentations

2023-06-26  Detecting overlapping reading frames by changes on dS estimates across virus genomes
42nd American Society of Virology’s meeting, Athens, Georgia
Oral: L Munoz Baena

2023-04-20  Decoding the effects of overlapping reading frames on selection pressures in virus genomes
30th International Dynamics & Evolution of Human Viruses. Heidelberg, Germany
Oral: L Munoz Baena

2022-07-17  Analysis and visualization of overlapping reading frames in virus genomes: a graph-based approach
41st American Society of Virology’s meeting. Madison, Wisconsin, USA
Oral: L Munoz Baena

2021-05-06  Simulating evolution of Overlapping Reading Frames in virus genomes
28th International Meeting on Dynamics & Evolution of HIV and Other Human Viruses. Online
Oral: L Munoz Baena, Wade Kaitlyn, and A Poon

2020-05-11  The global distribution of overlapping reading frames in viral genomes
27th International Meeting on Dynamics & Evolution of HIV and Other Human Viruses. Online
Poster: L Munoz Baena and A Poon
2017-10-12  Genetic potential of avocado rootstock determined by heritability studies
Seminar for the technical development in the avocado commercial crop in Antioquia. Itagui, Colombia
Seminar: L Munoz Baena, Andres Cortes, Alejandro Navas, Cipriano Diez, Laura Patino

2016-06-29  Development of a bioinformatic pipeline for molecular detection and characterization of plant
viruses using Next Generation Sequencing data
Ninth Latin American and Caribbean Agricultural and Forestry Biotechnology Meeting. Lima, Peru
Poster: L Munoz Baena, H Jaramillo Mesa, D Munoz Escudero, PA Gutierrez, M Marin

2015-09-26  Development of a Platform for Plant Virus Diagnostics and Characterization using Next-Generation
Sequencing
1st Conference on rapid Next-Generation Sequencing and bioinformatic pipelines for enhanced molecular
epidemiologic investigation of pathogens. Washington, D.C, USA
Poster: L Munoz Baena, H Jaramillo Mesa, D Munoz Escudero, M Marin, PA Gutierrez

2010  Harmonia axyridis (Coleoptera: Coccinellidae) development when feed with three different kinds of
prey diets under laboratory conditions
Intel International Science and Engineering Fair. California, USA
Poster: L Munoz Baena, WA Valencia Montoya, M Cano Ortiz

Scholarships and Awards

2022  John A. Thomas Award. Department of Microbiology & Immunology
Western University

2018  Graduate Research Scholarship. Department of Microbiology & Immunology
Western University

2017  Fellowship for young researchers. National Science Foundation of Colombia - Colciencias
Genetic potential of avocado rootstock determined by heritability studies

2016  Fellowship for young researchers. National Science Foundation of Colombia - Colciencias
Development of a bioinformatic pipeline for molecular detection and characterization of plant viruses using Next Generation
Sequencing data

2010  Travel award for young researchers. Explora - Colombian National Science Fair
Harmonia axyridis (Coleoptera: Coccinellidae) development when feed with three different kinds of prey diets under
laboratory conditions

Research

2018 - Present  Modelling the evolution of virus genomes
Founding: Natural Sciences and Engineering Research Council of Canada (NSERC Discovery Grant RGPIN 05516-2018)
Responsibilities: Develop a simulation program to measure the impact of selection on overlapping reading frames.

2017-2018  Development of certify avocado materials for Antioquia with genetic, physiological, and
phytosanitary quality
Founding: Colombian national science foundation - Sistema general de regalías.
Responsibilities: Data analysis (phenotypes prediction based on genetic characterization)

2015-2017  Bioinformatic analysis of the virome of leaf tissue of Solanum tuberosum subsp. Andigena var. capiro
in Antioquia (Colombia)
Founding: International Foundation for Science (IFS, Sweden, Grant: C4634-2)
Responsibilities: Laboratory experiments, sample collection in field, development of bioinformatic tools for data analysis

2013-2015  Determination of the RNA virome of Solanum tuberosum var. Diacol Capiro using Next Generation
Sequencing (NGS)
Founding: Research Vice-Rectory, Universidad Nacional de Colombia
Responsibilities: Laboratory experiments, sample collection in field, data analysis

Teaching

Teaching Assistant - Bioinformatics of infectious disease, MCRIOIM 4750G
Department of Microbiology & Immunology, Western University
Review lab material, guide students through assignments, grade lab reports

Teaching Assistant - Bioinformatics of infectious disease, MBI/MIMM 4750G
Medical Bioinformatics, Western University
Review lab material, guide students through assignments, grade lab reports

Teaching Assistant - Biology of Infection and Immunity, 2500AB
Department of Microbiology & Immunology, Western University
For the Virology section: review lecture materials and hold drop in sessions

AUGUST 30, 2023  LAURA MUÑOZ BAENA. CV 2
Extracurricular Activities

**Editor and Producer at GRADCAST**
The official radio show and podcast of the Society of Graduate Students at Western University
**Roles:** Interview graduate students about their research projects. Record, edit and produce selected episodes

**Under Water Hockey Player**
Currently part of the Canadian Underwater Hockey Women Elite Team

London, ON, Canada  
2019 - 2023

Colombia and Canada  
2014 - Present