

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Finding Insights in Immunological Data Through Interactive Visualizations

Leonor Ribeiro e Sousa Mendes de Freitas



Mestrado em Engenharia Informática e Computação

Supervisor: Professor Alexandre Valle de Carvalho, FEUP and INESC TEC Porto

Co-Supervisor: Professor Uri Hershberg, University of Haifa

Co-Supervisor: Professor Marco Oliveira, INESC TEC Porto

July 20, 2023

Finding Insights in Immunological Data Through Interactive Visualizations

Leonor Ribeiro e Sousa Mendes de Freitas

Mestrado em Engenharia Informática e Computação

July 20, 2023

Abstract

The immune system is constantly changing and adapting to respond to new diseases. B-cells produce B-cell receptors (BCR) and antibodies capable of binding to antigens derived from pathogens and other molecular determinants of the body. How B-cells diversify, mutate, and proliferate is key to the adaptive immune response that allows us to combat pathogens and protects us from recurrent disease. Cells that derive from the same ancestor share the same unique BCR source, encoded by a specific combination of gene fragments, are called clones. During an immune response to disease, B-cells proliferate and die at high rates and mutate their BCR gene, competing to better bind to antigens. This process of diversification and selection leads to affinity maturation of the B cell repertoire, which improves its binding to the disease. From the mutations and differentiation that occur in each clone, it is possible to establish a lineage that helps us understand the clones' evolution.

High throughput sequencing allows us to track tens of millions of individual BCR sequences and millions of clones in a person, as well as track several time points in an immune selection process. This opens the issue of how it is possible to relate lineages from past time points to those collected at later time points. Existing lineage visualization techniques are not built for understanding and representing the evolution of B cell lineages, if single or related lines, across multiple time points.

The current study, supported by recent visualization techniques, proposes a novel interactive visualization solution of B-cell clone lineages across multiple time points and at a specific time point. To achieve this, the user would start by filtering the clones with criteria such as the tissue belongs to or a clone size interval. Then, a stacked line area that represents the size evolution of BCRs would be rendered, displaying both the number of clones in the sample and their presence over sampling time points, represented by the size of the area of each stack. The user could also select a particular clone by clicking on the stacked line area and visualizing the details of its lineage tree at the desired time points. To provide the immunologist with a broader picture of the evolution of a B-cell lineage, the necessity of a window manager was recognized to allow the creation of different modes of visualization for trees at different time points and the comparison of the tree visualizations, both by displaying them side-by-side and embedding visual variables to convey meaningful data such as time. This last component would be obtained by clicking on the desired stack. A prototype was developed to ensure the viability of implementing the presented conceptual solution and, posteriorly, to serve as a platform that enabled performing an evaluation experiment to assess the solution's usability.

The evaluation experiment conducted consisted of a form with a series of questionnaires that assessed both the functionality of the tool and the immunological analysis that it was expected to support. From the results obtained from this experiment, it was identified that improvements should be implemented on the interface components, and there was a clear necessity to provide potential users with documentation on the utilization of the tool. Nonetheless, it was possible to infer that this tool could simplify the process and assist immunologists in analyzing and compar-

ing lineage trees at different time points as well as obtaining an overview of the clone evolution and, therefore, contribute to a better understanding of the evolution of B cell clone populations.

Keywords: visualization, graphical interface, interactive, lineage tree, clone, B cell, immunology, mutations, selection.

Resumo

O sistema imunológico está em constante mudança e adaptação para responder a novas doenças. As células B produzem recetores de células B (BCR) e anticorpos capazes de se ligarem a antígenos derivados de agentes patogénicos e outros determinantes moleculares do corpo. A chave para a resposta imunitária adaptativa que nos permite combater patógenos e nos protege de doenças recorrentes provém de como as células B se diversificam, sofrem mutações e proliferam. As células que derivam do mesmo predecessor partilham o mesmo BCR, codificado por uma combinação específica de fragmentos de genes, chamados clones. Durante uma resposta imunitária a uma doença, as células B proliferam e morrem em taxas elevadas, sofrendo mutações nos BCR, competindo para se ligarem aos antígenos de forma mais eficiente. Esse processo de diversificação e seleção leva à maturação por afinidade do repertório de células B, o que melhora a sua capacidade de ligação ao elemento patogénico. A partir das mutações e diferenciações que ocorrem em cada clone, é possível estabelecer uma linhagem que nos ajuda a compreender a evolução dos clones. O sequenciamento de alto rendimento permite identificar dezenas de milhões de sequências BCR individuais e milhões de clones de um indivíduo, assim como identificar vários pontos no tempo num processo de seleção imune. Este processo levanta a questão de como será possível relacionar linhagens recolhidas anteriormente com as recolhidas posteriormente. As técnicas de visualização de linhagem existentes não foram construídas com o intuito de compreender e representar a evolução das linhagens de células B, quer sejam linhagens únicas ou integradas, em vários momentos temporais.

A presente dissertação, apoiada por técnicas de visualização recentes, propõe uma nova solução de visualização interativa de linhagens de clones de células B que integram vários pontos no tempo e linhagens de um dado momento específico. Para o conseguir, o utilizador poderá filtrar os clones com critérios como o tecido a que pertence ou um intervalo de tamanhos dos clones. Seguidamente, um gráfico de stacked areas que representa a evolução do tamanho dos BCRs seria renderizada, ilustrando o número de clones na amostra e a sua presença nos vários momentos de amostragem, representados pelo tamanho de cada área. O utilizador também pode seleccionar um clone específico clicando no gráfico e visualizando os detalhes das suas árvores de linhagem nos pontos no tempo desejados. Para fornecer a um imunologista um quadro mais amplo da evolução de uma linhagem de células B, reconheceu-se a necessidade de um window manager para permitir a criação de diferentes modos de visualização, para árvores em diferentes momentos, e a comparação das visualizações das árvores, exibindo-as lado a lado e incorporando variáveis visuais para transmitir dados significativos, como o tempo. Este último componente é obtido clicando na área correspondente ao clone desejado. Foi desenvolvido um protótipo para possibilitar a implementação da solução conceptual apresentada e, posteriormente, para servir de plataforma para permitir realizar uma experiência de avaliação da utilização da solução.

O teste de avaliação realizado consistiu num formulário com uma série de questionários que avaliaram tanto a funcionalidade da ferramenta, como a análise imunológica que se esperava que ela suportasse. A partir dos resultados obtidos neste teste, identificou-se quais as melhorias que

deveriam ser implementadas nos componentes da interface, tendo ficado clara a necessidade de fornecer aos potenciais utilizadores documentação sobre a utilização da ferramenta. Concluindo, foi possível inferir que esta ferramenta poderia simplificar o processo e auxiliar os imunologistas na análise e comparação de árvores de linhagens em diferentes momentos, bem como obter uma visão geral da evolução do clone e, assim, contribuir para um melhor entendimento da evolução de populações de clones de células B.

Palavras-chave: visualização, interface gráfica, interactiva, árvore de linhagem, clone, células B , imunologia, mutações, seleção.

Acknowledgments

I would like to express my full gratitude to all the people that helped and supported me during the development of this dissertation and contributed to make it possible.

I would like to start by expressing my gratitude to my supervisors, without whom this work would not be possible, Professor Alexandre Valle de Carvalho, Professor Uri Hershberg from University of Haifa and Professor Marco Oliveira from INESCITEC for all the guidance provided, support and all the patience and friendship. I would also like to acknowledge and thank Areen Shtewe for all the guidance and support with my work. I am extremely thankful for all the learning opportunities and the encouragement in this process that helped me grow into a better student, professional and person.

I would also like to express my gratitude to all the people that dispensed some of their time to help me on the development of this work, as well participate in the evaluation of my prototype.

I also want to acknowledge the University of Haifa and INESCITEC for all the resources made available to me.

Last but most definitely not least, I want to acknowledge my parents, Susana and Miguel, my sister Francisca, and Nuno for the unconditional support and love during this journey and for making life easier so that I could dedicate myself to this project. I would like to thank my parents for all the opportunities that they provided me with. I want to thank you all for being so understanding and supportive. This work is dedicated to you.

Leonor Freitas

Contents

1	Introduction	1
1.1	Context and Motivation	1
1.2	Problem Definition and Hypothesis	2
1.3	Objectives	3
1.4	Document Structure	3
2	State of the Art	4
2.1	Information Visualization	4
2.1.1	Information Visualization Definition	4
2.1.2	Information Visualization Pipeline	5
2.1.3	Information Visualization Data-Based Taxonomy	6
2.1.4	Task-Based Taxonomy	13
2.1.5	Visual Elements and Variables	14
2.1.6	Information Visualization Enabling Technologies	15
2.2	HCI Testing	16
2.2.1	Number of Usability Testers	16
2.2.2	Post-Task and Post-Test Questionnaires	17
2.3	Immunology	18
2.3.1	Immunology Definitions and Concepts	18
2.3.2	AIRR Community	20
2.3.3	AIRR Community	20
2.3.4	AIRR Data Commons	21
2.3.5	ImmuneDB	22
2.3.6	Immcanation and IgPhyML	22
2.3.7	Newick Format Trees	23
2.4	Previous Work in Immunology Visualization	23
2.4.1	AncesTree	24
2.4.2	Ggtree	24
2.4.3	iTol	24
2.5	Summary	24
3	Conceptual Solution	26
3.1	Problem Statement	26
3.2	B-cell Receptor Evolution Visualization Pipeline	28
3.3	Conceptual Proposed Solution	30
3.3.1	Data filtering solution	30
3.3.2	Clone size evolution in time visualization solution	31
3.3.3	Lineage tree visualization solution	32

3.3.4	Metadata visualization	35
3.3.5	Alternative radial lineage tree layout	36
3.3.6	User-Interface layout	37
3.4	Proposed Solution prototype	39
3.5	Summary	40
4	Visualization Prototype	42
4.1	Tech Stack	42
4.2	Prototype Architecture	43
4.3	ImmuneDB Database Structure	45
4.4	Tree Building	46
4.4.1	Data Preparation	46
4.4.2	IgPhyML and Newick Trees	47
4.5	Application	48
4.5.1	Filtering Stage	48
4.5.2	Clone Size Stacked Area Chart	50
4.5.3	Lineage Tree Visualization	51
4.6	Summary	56
5	Evaluation	58
5.1	Objectives	58
5.2	Evaluation Methodology	58
5.3	Testing protocol	59
5.4	Informed Consent Form	59
5.5	Pre-Questionnaire Form	60
5.6	Pre-test Prototype Presentation	60
5.7	Questionnaire Form	60
5.7.1	Functional Tasks	61
5.7.2	Immunology Analysis Tasks	61
5.7.3	Post-Task Group Questionnaires	62
5.7.4	Post-Questionnaire Form	62
5.7.5	User Observation Open Questions	63
5.8	Summary	63
6	Results and Discussion	65
6.1	Overall Participants' Stats	65
6.2	Filtering Stage	65
6.2.1	Results	66
6.2.2	Discussion	67
6.3	Stacked Area Chart (Clone Size Evolution Time Series)	68
6.3.1	Results	69
6.3.2	Discussion	69
6.4	Lineage Tree Visualization	69
6.4.1	Task Results	70
6.4.2	NASA-TLX Post-task Questionnaire Results	71
6.4.3	Discussion	72
6.5	Immunology Visual Analysis	73
6.5.1	Results	74
6.5.2	Discussion	75

6.6	SUS Post-Test Questionnaire	75
6.6.1	Results	76
6.6.2	Discussion	76
6.7	Open Questions	77
6.7.1	Results	77
6.7.2	Discussion	78
6.8	Summary	79
7	Conclusion	81
7.1	Conclusions	81
7.2	Future Work	84
	References	86
A	Questionnaire	90
A.1	Questionnaire	90
A.2	Results	91
B	Prototype	103

List of Figures

2.1	Visualization Pipeline. Adapted from Liu et al. [38]	5
2.2	Classical Tree View. Nguyen, Q. et al. [41]	8
2.3	H-Tree Layout. Nguyen, Q. et al. [41]	8
2.4	Balloon View. Nguyen, Q. et al. [41]	8
2.5	Radial View. Nguyen, Q. et al. [41]	9
2.6	Hyperbolic Tree Browser. Shneiderman, B. et al. [50]	10
2.7	Early version of the Cone Tree. Shneiderman, B. et al. [50]	10
2.8	Initial version of Tree-Maps. Shneiderman, B. et al. [50]	10
2.9	Different types of Line Plots. Aigner, W. et al. [8]	12
2.10	Stacked Graph visualization techniques. Thudt, A. et al. [51]	13
2.11	Visual Variables and their characteristics, adapted from Carpendale [18], Roth [46] and axis map website [5].	15
2.12	Examples of interactive visualizations created using D3. Liu, S. et al. [38]	16
2.13	Illustration of the NASA-TLX scales, as seen in Hart [31].	18
2.14	The System Usability Scale Questionnaire, extracted from Brooke, J. [14]	19
3.1	Adapted Visualization Pipeline	29
3.2	Filtering Stage Developed in the 2021 Artathon.	31
3.3	Time series visualization and interaction first consideration; lineage tree initial visualization consideration from Artathon 2021.	32
3.4	First iteration of the lineage tree mockup.	33
3.5	Color as a time-defining visual variable in a lineage tree.	34
3.6	Lineage Tree Mockup with Time Points identified by color and pattern; Integrated and Single trees.	35
3.7	Possible solution for tree element's detail visualization.	36
3.8	Node and edge selection.	37
3.9	Metadata Visualization in Lineage Trees.	38
3.10	Concept for Lineage Tree Visualization Layout and Solution (Window Manager).	39
3.11	Tree Views side-by-side.	40
4.1	Prototype Architecture	44
4.2	ImmuneDB table structure [45]	45
4.3	Prototype Developed from Proposed Solution	49
4.4	Prototype Filtering Form.	50
4.5	Example of a Stacked Area Chart render.	50
4.6	Tree View Creation Form.	52
4.7	Tree Views Interface.	53
4.8	Example of a Tree View.	53

4.9	Tree radial layout.	54
4.10	Tree visualization with the different view option parameters toggled.	55
6.1	Chart of the results of the first question of the Filtering Stage Section.	66
6.2	Results of the third question of the filtering stage questionnaire.	67
6.3	Success Rate Chart of Tasks LTI1 to LTI3.	71
6.4	Average results chart of NASA-TLX Questionnaire.	72
6.5	Results boxplot of NASA-TLX Questionnaire.	73
6.6	Average Results of the SUS Post-Test Questionnaire.	76
6.7	Results boxplot of SUS Questionnaire.	77

Chapter 1

Introduction

This chapter intends to provide an introductory reflection on the work that was developed, beginning by providing context as well as explaining the motivation behind it in Section 1.1. Section 1.2 describes the problem this dissertation intends to solve, followed by the hypothesis. The goals of this visualization of immunology data are, thereafter, defined in Section 1.3. Lastly, the structure that the rest of the present document will follow is detailed in Section 1.4.

1.1 Context and Motivation

The human immune system evolves and adapts in order to protect us from pathogens and other foreign molecules in the body. Particularly, the adaptive immune response is the part of the immune system that is highly specialized and that suffers changes throughout time.

B-cells are lymphocytes and are a part of the adaptive immune response. B lymphocytes produce antibodies, also known as immunoglobulins and B-cell receptors (BCR). BCRs are capable of binding specifically to antigens, in order to render them inactive or to mark them to be destroyed [6] [9] [36]. Considering the Y shape of the antibody molecule, the differentiation process of single cells occurs thanks to mutations in the gene segments located in the top part of the BCR. This creates a unique antibody, capable of binding to a specific antigen, with a specific combination of gene fragments [6] [9] [36]. The B-cell population can be divided into clones, according to their unique BCR source, with cells that share the same ancestor. In this population, clones will compete and mutate their BCR in order to have a more efficient immune response. By inspecting the BCR sequences resultant from the mutations that occur in clones, lineages trees can be established to follow the evolution of the clones [53].

With the advances in technology, high throughput DNA and RNA sequencing is now possible, allowing the collection of millions of relevant sequences from BCRs and clones in each individual. The great increase in data generation made it possible to create lineages from several points in time, instead of from a single time point, even with the rate at which evolution unfolds [53] [36].

This also made it essential to create standards in how to store and retrieve the information relative to adaptive immune responses. The AIRR community [1] set itself to create such standards, and there are several compliant repositories, including ImmuneDB. ImmuneDB [45] is a database tool, based in MySQL, that allows storing data and metadata on B-cell receptors and T-cell receptors collected from high-throughput sequencing.

Information visualization is a field that has as main goal providing interactive visual representations that help build meaningful insight over abstract data [38]. This is particularly valuable with datasets of large dimensions, such as the ones that contain high-throughput sequencing data.

The lineage visualization techniques that already exist allow for the analysis of clone lineage trees from a single time point. They do not, however, contemplate the large amounts of immune data that is collected with more recent technologies nor the relation of lineage trees across time points [26] [37] [55].

The existence of an interactive visualization that not only demonstrates the evolution of clone presence over time in an immune response but also allows for the interactive display of lineage trees in single or multiple time points (establishing a relation between them in the former case) can help the immunology research community better understand the behavior of human immune responses.

1.2 Problem Definition and Hypothesis

The possibility, with high-throughput sequencing, to extract lineages from different time points in an immune response opens up new issues relating to their analysis.

The main concern would be on how to relate lineages from different time points: how do we relate the lineage from a selected time point to the ones from previous time points?

It is common to detect discrepancies with lineages of members of the same clone between time points. Defining how these discrepancies are approached in terms of lineage construction (a single tree where discrepancies are ignored or separate lineages for each time point, with identical clones identified and connected) can be detrimental to finding the evolution process of B-cell clones.

It is, then, essential to understand the best approach to analyze these lineage trees in order to make correct conclusions on the behavior of an immune response.

With the available large amounts of data it is also necessary to have a general understanding and an overview of the presence and diversity of clones present in a dataset.

Thus, in order to address these identified concerns this study proposes to create a visualization solution with sequential interaction that not only allows the mapping and rendering of trees than include all possible time points but also trees from a specific given time point. In order to achieve that, an initial filtering process is created, from which is rendered a clickable stacked line area that shows the evolution of each clone in time. It should be possible to select a clone and visualize its lineage trees. This solution intends to support a better understanding of the evolution of a B-cell receptor repertoire.

1.3 Objectives

This project aims to help solve the encountered problems described in section 1.2. Therefore, there are three main goals to take into consideration for the approach taken when developing the product: creating an interactive visual narrative of multiple time points, obtaining a meaningful visual representation, and being able to visualize B-cell clone lineages at a single specific time point or at several time points.

In order to achieve these objectives, this study intends to create an interactive visualization of B-cell clone and their lineages, using recent visualization techniques that support an exploratory data analysis tool.

In the visualization, the user should be able to track the presence of each pre-selected clone (after data filtering) over time and select lineage trees to be visualized either at a single time point or at multiple chosen time points.

The obtained result should be efficient at providing an understanding of the abstract B-cell receptor data that it is not possible to attain just by studying the dataset.

1.4 Document Structure

The present document follows a structure that contains 7 chapters.

The first chapter is this introductory one.

Chapter 2 contains an overview of the Information Visualization discipline, taking a particular focus on hierarchical and temporal visualization techniques, describes the process of usability testing, provides information on immunology and immunological research data standards, necessary for the contextualization of this study, and analyses already existing visualization tools for immunology.

Chapter 3 describes the proposed solution for this project. This chapter starts by addressing the stated problem and identifying the research questions this study intends to answer. Then, the conceptual solution is described, as well as the process to obtaining it.

Chapter 4 documents the development of the functional prototype that implements the visualization solution proposed in the previous chapter.

Chapter 5 describes the methodology employed to conduct the evaluation experiment used to test the usability and efficacy of the developed solution and prototype.

In Chapter 6 the results obtained from the evaluation experiment and the developed prototype are presented and discussed.

The final chapter, 7, draws the relevant conclusions for the project, taking into account the developed prototype and the evaluation results described previously.

Chapter 2

State of the Art

This chapter includes the revision of literature of all areas that concern this work. Section 2.1 describes an overview of important concepts in Information Visualization, such as different taxonomies. Section 2.2 reviews concepts and methods of Human-computer interaction Testing. In Section 2.3, the Immunology concepts and tools relevant for this work are presented. Finally, Section 2.4 contextualizes the previous work in Immunology Visualization related to the present study.

2.1 Information Visualization

2.1.1 Information Visualization Definition

Information Visualization, or InfoVis, is a discipline that aims to help users gain meaningful insight into data through computer-generated visual representations that can be interactive [38]. Representation and interaction, the two components of visualization, are connected and affect one another, namely when the user's interaction changes the representation [54].

Information visualization concerns itself with the design, development, and application of graphical representations of information. It uses mainly non-spatial, abstract data that needs to be transformed to be understood more easily. InfoVis aims to visually deliver complex information to users by assigning new meanings to graphic patterns, presenting it accurately and rigorously [19].

It is important to define the scope of Information Visualization and differentiate it from areas like scientific visualization and data visualization. This is not always easy, considering these fields have overlapping subjects, and many experts use these terms interchangeably [19]. Information Visualization uses primarily abstract, qualitative, and non-spatial data, while data visualization, for example, uses quantitative data. Information Visualization also has as a primary concern the output form and information and not as much the input data, like data visualization [19] [34].

The recent expansion of big data has made this a prominent area used in many different applications, such as scientific data, business data, and other fast-rising fields such as search results [38]. A greater understanding of data and its analysis can be central to growth, innovation, and productivity [38].

2.1.2 Information Visualization Pipeline

Visualization can be looked at as the mapping of data into visual forms in a way that allows humans to better understand the underlying information [54]. This process from data to visualization and user interface can be represented as a flow/pipeline, which is described below and shown in Figure 2.1. This pipeline is composed of five stages: data transformation and analysis, filtering, mapping, rendering, and UI controls [38].

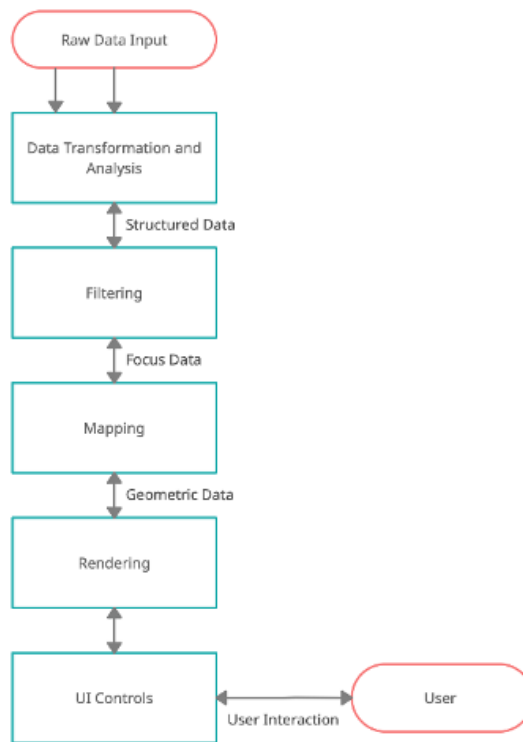


Figure 2.1: Visualization Pipeline. Adapted from Liu et al. [38]

Raw data, usually unstructured, is the input for data transformation and analysis, which suffers a data size reduction if necessary. In this model, several techniques such as clustering and categorization (to extract structured data) and removing noise, interpolating missing values, and correcting possible errors in measurements [38]. There is also the possibility of adding metadata to the original data, describing the information it contains. Transforming raw data can be essential if the data is abstract without a spatial component. This step intends to make it simpler to map data into visualizations [16]. The data transformed in the previous stage is passed onto the filtering stage, where the data to be visualized is selected (focus data) [38]. Several techniques to select relevant data can be applied at this stage, such as details-on-demand, brushing, and dynamic queries [25].

Similarly, as in the previous step, the output obtained is passed onto the following stage: mapping. The filtered data is mapped to graphical/geometric primitives (points, lines) and their attributes [38]. These primitives compose what are called visual structures by being placed onto a

representational space in order to translate information. The mapping should allow fast interpretation, induce the least amount of errors, and be well perceived by the user [16].

In the rendering stage, the previously obtained graphical primitives and visual structures are expressed as image data.

The user can interact with the generated image with UI controls to better understand the represented information [38]. This stage comprises all the interactions that can transform the generated visualization in a way that allows extracting more information. Interaction can affect all of the pipeline stages, whether the mapping of the data or the rendered visual structures. [16]

2.1.3 Information Visualization Data-Based Taxonomy

Taxonomies consist of classifications that allow to better choose the visualization techniques to be used. In the following sections, there are going to be described two different visualization taxonomies: a data-based and a task-based taxonomy.

In 1996, Shneiderman [16] proposed a taxonomy for Information Visualization techniques based on data type.

The proposal is comprised of the following classification:

- One-dimensional data: also known as linear data types; this type of data is organized sequentially; each item of the data collection is composed by a string of characters.
- Two-dimensional data: also known as planar or map data; includes geographical maps; each data collection is composed by several items that make up a part of the total area; each individual item has unique properties such as name, value (task-domain) and color, size (interface-domain); if a multi-layer approach is used to analyze map data, each layer is two-dimensional.
- Three-dimensional data: this type of data respects to real-world objects; this may include molecules or buildings; each item has volume and complex 3-dimensional relationships; the visualization of this type of data must take into consideration of orientation and position, as well as problems of occlusion.
- Multi-dimensional data: multi-dimensional data is present in the majority of relational and statistical databases, where each element of the database, that has n attributes, corresponds to a point in an n -dimensional space; this type of data can be represented with 2D or 3D scatter diagrams.
- Temporal data: data that describes a timeline, which is of great significance for several areas of application, making it important to distinguish it from other one-dimensional data; the fact that items have a start and a finish and may overlap is what differentiates temporal data. Time dependent visualizations are further described in Section 2.1.3.2.
- Tree data: tree structures, also known as Hierarchical structures, are a visualization type in which each data item to be represented (tree node) has a relation of childhood to another

item, with the exception of the item that represents the root node of the tree. Hierarchical/tree structures will be further explained in Section 2.1.3.1.

- **Network:** this type of structure is used when tree structures are not sufficient to represent data and there is a necessity to have relationships between an arbitrary number of nodes/items - graphs; this type of data structure comprises many special cases, for example acyclic or directed vs. undirected; this type of structure can be represented by a diagram of nodes and edges (links) or a square matrix that tracks the existing items and links between them in its rows and columns.

Based on data type taxonomy, other classifications were proposed. For example, the Data State Model [25] is a taxonomy based not only in data types but also in the stage of the data transformation (Value, Analytical Abstraction, Visualization Abstraction and View) and transformation operators (Within stage operators, Data Transformation, Visualization Transformation and Visual Mapping Transformation). This classification gives the possibility of extracting information for each of the visualization pipeline steps in every technique considered, facilitating the design of new visualization techniques.

2.1.3.1 Hierarchical Data Structures

Tree structures and visualizations represent nodes/vertices connected by branches/edges in hierarchies [48]. This hierarchy is a relation in which each node will have only one node as a parent and can have multiple children. Both the elements of the tree and the connection between child and parent nodes can have attributes associated with them. There are several types of tree structures, according to the number of levels of the tree, the number of children of each item, and the type of items on internal nodes and leaves. Special cases like broad (high fanout – high branching factor) and deep (small fanout – many levels) trees require approaching in specific ways. [49]

There are several types of existing tree visualization techniques: classical hierarchical view, the h-tree layout [41], cone trees [50], hyperbolic tree/browser[50][41], tree-maps[50][41], balloon view [41], radial view [41]. Some of these visualization and mapping techniques were landmarks on information visualization research, namely tree-maps, the hyperbolic tree, and cone trees, which will be described in more detail [50].

Classical Tree View (see Figure 2.2) places children nodes below their ancestor, connected to it. There can be top-down, left-to-right, and grid layouts. Since the use of space is not optimized with this technique, there is a great amount of unutilized space. Consequently, the traditional hierarchical view is only adequate for moderately large trees [41].

The H-Tree layout (see Figure 2.3) is employed to visualize binary trees, only performing well with balanced trees, and therefore, it is not suitable for generalized tree visualization [41].

The Balloon View (see Figure 2.4) consists of parent nodes of each sub-tree connected to its descendants, which are displayed in a way that forms a circle. The projection of a cone tree (described in a paragraph below) also results in a balloon view. This visualization does not optimize space [41].

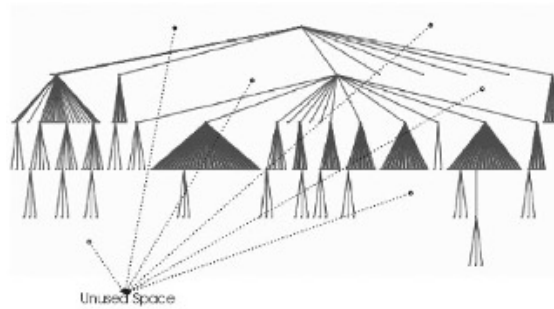


Figure 2.2: Classical Tree View. Nguyen, Q. et al. [41]

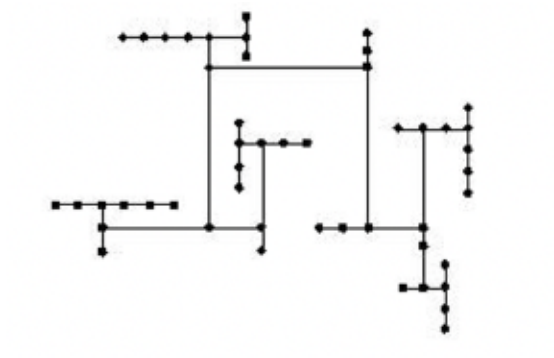


Figure 2.3: H-Tree Layout. Nguyen, Q. et al.[41]

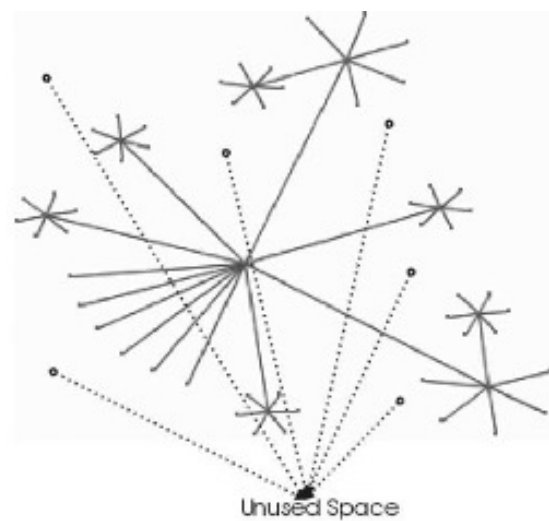


Figure 2.4: Balloon View. Nguyen, Q. et al. [41]

The Radial view (see Figure 2.5) uses an algorithm that places children of a subtree recursively in the shape of a circular wedge. The angle of each wedge is proportional to the number of children in that sub-tree. This technique is also not optimized in terms of space use [41].

The Hyperbolic Tree, as seen in Figure 2.6, is a technique that builds trees in the hyperbolic plane and maps the obtained tree into an ordinary two-dimensional plane.

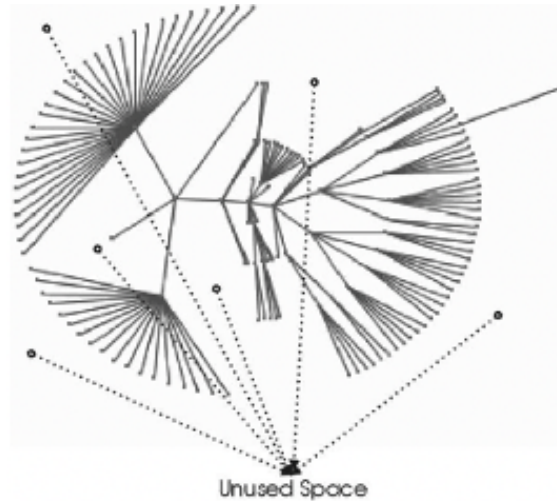


Figure 2.5: Radial View. Nguyen, Q. et al. [41]

This visualization technique places the root node in the center and positions first-level children nodes in a circle around it. Consequent-level nodes are placed in larger concentric circles or ovals, and outer-level nodes' relative size is smaller. Therefore, this representation is in a fisheye layout, with an amplified center and smaller peripheral areas. The hyperbolic tree browser allows any node to be positioned in the center, redrawing the tree. Later versions also included color and node size to represent other attributes. [50], [41]

The Cone Tree, represented in Figure 2.7, is a three-dimensional layout with animated rotation. The root node is on the top of the representation, connected to its immediate children, placed in a lower-level circular configuration – therefore the appearance of a cone. Nodes in lower levels of the representation are also in a cone layout, with the leaf nodes closest to the bottom of the screen. Perspective and shadow effects aided in the visual comprehension of the structure [50].

Tree-maps, of which an example can be seen in Figure 2.8, is a recursive algorithm created by Shneiderman with two characteristics: space-filling, considering it fully uses the visualization region, and space-limited, since a rectangular shape delimits the visualization. The representation shows nested rectangles, each representing a leaf of the tree. The attributes of each leaf determine the area of the rectangle (which represents the relative size of the node) and its color. The first version of the algorithm ordered the children meaningfully but compromised readability due to the aspect ratio of the generated rectangles. Other versions improved the aspect ratio of the nodes and placement according to their area on the visualization region. This is a technique that optimizes the space of the visualization. However, due to the lack of edges connecting the nodes, this representation might make it harder to understand the relation between the nodes and the relational

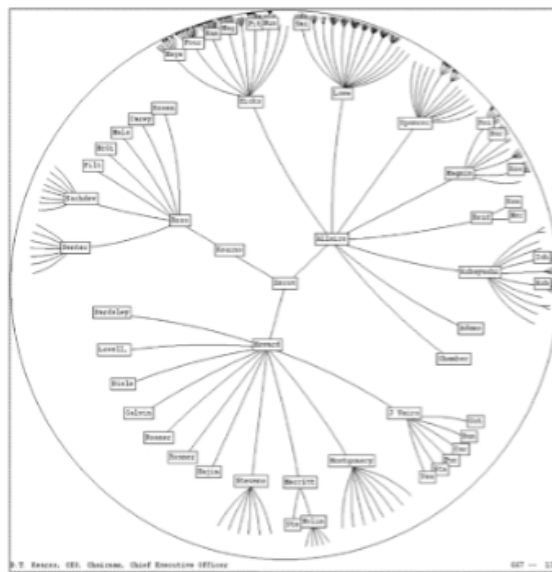


Figure 2.6: Hyperbolic Tree Browser. Shneiderman, B. et al. [50]

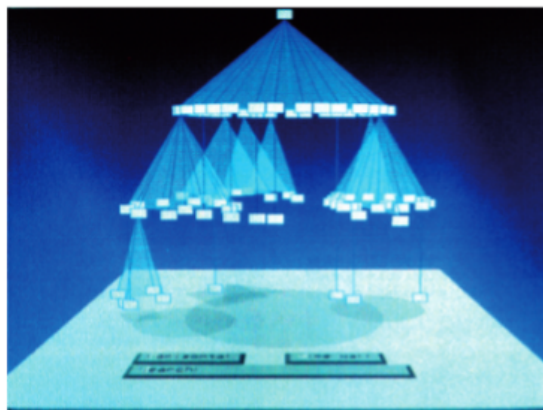


Figure 2.7: Early version of the Cone Tree. Shneiderman, B. et al. [50]

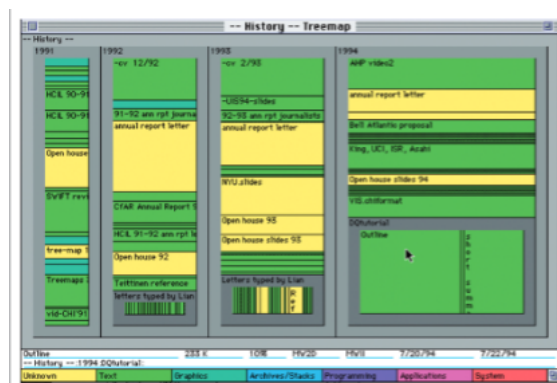


Figure 2.8: Initial version of Tree-Maps. Shneiderman, B. et al. [50]

nature of the data structure [50] [41].

2.1.3.2 Temporal Data Structures

Time series data intends to establish a relationship between the data and time passed and represent the evolution of data over time [40]. According to MacEachern [39], a well-designed time-series visualization should answer the following questions:

- Does a data element exist at a specific time? (Temporal Existence of a data element)
- When does a data element exist on time? Is there any cyclic behavior? (Temporal location)
- How long is the time span from beginning to end of the data element? (Temporal interval)
- How often does a data element occur? (Temporal texture)
- How fast is a data element changing, or how much difference is there? (Rate of Change)
- In what order do data elements appear? (Sequence)
- Do data elements exist together? (Synchronization)

The temporal dimension and the time axis are obvious critical criteria to analyze when approaching time visualization. The taxonomy proposed by Frank [27] describes important concepts that help characterize the formerly mentioned, which includes temporal primitives (time points vs. time intervals) and structure of time (linear vs. cyclic vs. branching).

About the temporal primitives that constitute the time axis, these can be time points or intervals. A time point is described as an instant in time, while a time interval can be defined by two time points, or a time point and a duration. Depending on temporal primitives, different analysis objectives can be achieved. According to Aigner et. al, different structures for the time axis should also be considered: linear, cyclic, and branching [7]. Linear time consists of ordered temporal primitives, the same way we naturally perceive time. Cyclic time structures contain recurring temporal primitives, in a form that each time primitive is preceded and always followed by the same other temporal primitives. It is often convenient to represent cyclic time in a linear time axis. Branching structures can be considered graphs, where each of its vertices represents time primitives, and the edges define their order. Vertices can have multiple outgoing edges, which describe different temporal alternatives. [7]

There are also relevant concepts in the formerly mentioned taxonomy relative to the data associated with the time axis. One particularly relevant distinction concerns the number of variables.

With univariate data, each temporal primitive is linked to a single data value, whereas with multivariate data, each temporal primitive is connected with multiple data values. The latter introduces the necessity of detecting correlations in the visualization.[7] Concerning the representation of temporal data structures, it is relevant to emphasize the nature of the time dependency: whether the visualization is static or dynamic. Static visualizations are represented by a single still image.

Dynamic visualizations express the time dependency of the data through the physical domain, with several images or animations. The presence or absence of interaction does not influence this categorization.

It is also pertinent to mention the concept of dimensionality of the representation, that is, the possibility of representing the data in two or three dimensions. [7] Considering how complex the visualization problem with time and time-oriented data is, there are many different techniques for this type of data. Many visualizations are application-dependent, so it makes sense to focus the review on more classic visualizations applicable to different areas.

Relevant visualization techniques will be described in the following paragraphs.[8] Line plots are the most frequent form of representing time-oriented data. In these representations, data points are connected, highlighting the temporal evolution of the data. Depending on the data used, several types of connections can be applied to the visualization, from straight lines to step lines to Bezier curves (as seen in Figure 2.9). There are also sub-types of line plots, such as band graphs (represented in the bottom-right of Figure 2.9) [8].

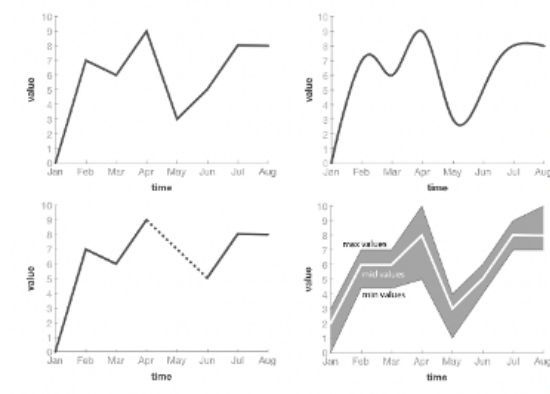


Figure 2.9: Different types of Line Plots. Aigner, W. et al. [8]

Stacked Graphs, and their variations, are a type of visualization that allows the representation of multiple time series in a single visualization by stacking individual time series represented by areas called streams. This technique distorts the baseline of each stream, which motivated various improvements to it, such as the ThemeRiver and StreamGraphs.

ThemeRiver also stacks individual time series but around a central axis, making the outline of the visualization symmetric. This technique is useful for detecting overall changes and evolution of the data but not as advantageous for observing smaller and more detailed changes.

StreamGraphs are a variation of ThemeRiver, reducing the distortion of the layers by smoothing the outlines of each area. Interaction can be essential to improve the readability of these visualization techniques. [51] All three of these techniques are shown in Figure 2.10, using three different datasets.

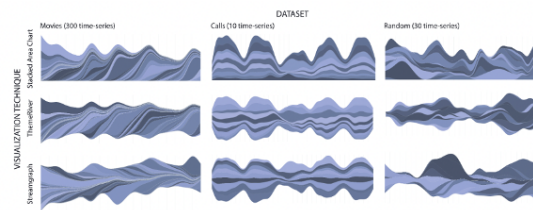


Figure 2.10: Stacked Graph visualization techniques. Thudt, A. et al. [51]

2.1.4 Task-Based Taxonomy

The taxonomy proposed by Shneiderman [16] also contemplated tasks and interaction techniques, namely Overview, Zoom, Filter, Details-on-Demand, Relate, History, and Extracts.

These seven tasks can be described as follows:

- **Overview:** consists of strategies to gain insight over all of the data represented; these include zoomed-out views accompanied by a smaller, more detailed view and the fisheye strategy, where the distortion magnifies one or more parts of the visualization.
- **Zoom:** this technique allows zooming in on data of interest, with tools that help control zoom focus and factor; smooth zooming is essential to maintain the sense of position and context.
- **Filter:** involves filtering out uninteresting data by applying dynamic queries, which are one of the fundamental concepts in InfoVis. **Details-on-Demand:** this task entails selecting an individual or a group of data items in order to access its details, for example, via a pop-up window.
- **Relate:** this technique allows us to observe relationships between data points, for example, by selecting all the elements with a particular attribute or obtaining data that provides additional information related to what was selected.
- **History:** involves keeping a record of interaction actions that allow error correction and refinement.
- **Extracts:** this action permits exporting a selected group of data and its query parameters in order to, for example, analyze it statistically.

In order to allow users to look and interact with information with many dimensions, difficult to fit in an interface at one time, techniques that permit various views with different levels of detail were proposed as an alternative to moving information, through scrolling, paging, and panning. These techniques include Overview+Detail, Zooming, and Focus+Context. Overview+Detail interfaces create a spatial separation, where the user can see both a general view of the information and a more detailed version of a part of it.

Zooming creates a temporal separation since it does not permit the simultaneous visualization of the zoomed-in (magnified) view and the zoomed-out (demagnified) view. Given its separation

limitations, both Overview+Detail and Zooming leave the task of assimilating the global relationship between views to the user. Focus+context techniques integrate data representation in a single display with all parts visible simultaneously. One of the main applications of Focus+context is the fisheye view, which causes a distortion that decreases the size of the areas of the visualization outside the focal area. [21]

2.1.5 Visual Elements and Variables

As already described in previous sections, mapping abstract data into a visual representation is an essential step to obtaining meaningful information.

In Semiology of Graphics [11], Bertin introduces basic visual units that compose full visualizations called marks, which are the simplest graphic element and establish relationships between data [18]. Examples of marks are points, lines, and areas [18]. Visual variables are the possible variations on marks and work as visual metaphors for information [46]. Although visual variables were developed in the context of cartography, their definitions and characteristics are used today across the information visualization field [46]. Creating the best possible visualization solution is significantly based on choosing the correct visual variables to represent the data characteristics [18].

Bertin [11] originally defined seven types of visual variables: position/location, size, shape, value/lightness, color/hue, orientation, and texture/pattern [18] [46].

Visual Variables can have different characteristics and organizational perceptions [46]. The first, selective, occurs when a mark transformed only in a specific variable becomes individually distinct from the remainder of the elements in the space [18]. Associative variables allow for elements changed with it to be perceived as a group. When there are variations according to an ordered variable, the marks become ranked visually, and there is a notion of “less” and “more” [18] [46]. Finally, when a variable is considered quantitative, which in turn is an extension of the ordered perception, it is possible to attribute to the elements and their differences a numeric value [18] [46].

Position is defined as the location of a certain element or mark in the visual space according to a set of coordinates, either two or three-dimensional [18][46]. The location of an element is the most essential of all the visual variables, especially in cartography, as it conveys a certain hierarchy, and the closer to the optical center, a more immediate interpretation by the user. This visual variable is considered associative, selective, ordered, and quantitative since it can easily be used to identify both isolated and group elements spatially and since the coordinates associated with this variable have corresponding numerical values that quantify and order the elements [18][46].

Changing the size of a mark can be done by changing its dimensions and is defined by the amount of visual space that each element takes. The variation of size in an element is considered selective, ordered, and quantitative [18] [46]. The size visual variable is considered a selective variable since it is a distinct characteristic that describes an element and is ordered and quantitative since there is the possibility of attributing a numeric value [18].

Shape is a visual variable that represents the outline and form of an element/mark and can have a broad level of abstraction, starting from simple triangles or circles to, for example, the shape of a recognizable country on a map [46]. A change in shape does not affect the size of the element [18]. Shape is associative, permitting all the elements with the same shape to be perceived as a group.

When the value of a mark is varied, the “lightness” of a color is changed, i.e., how light or dark the element’s color is and is independent of the color hue [18] [46]. Value is a selective variable since, despite changes in other variables, selecting an element with a specific color value remains possible. It is also an ordered variable since the darkness of an element can be “more” than the darkness of another.

Color hue is the variable that represents, as the name indicates, the visible-light wavelength that corresponds to the color attributed to a mark. Color is both an associative and a selective variable since it has the capacity to emphasize a single element and also identify groups independently of other variable changes.

The variation of direction or rotation of a visual symbol in relation to a determined baseline or other elements is denominated orientation. Variation of orientation is also selective and associative, especially if no perspective is being employed and if the shape or pattern used in the mark have a linear nature [18].

Texture variation is based on the different patterns/grains that can fill an area or line. In turn, patterns are composed of a repetition of marks and changes in their orientation and color. Again, this variable is also selective and associative.

Figure 2.11 details these variables and the corresponding characteristics.

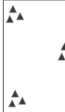





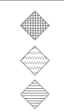
Position	Size	Shape	Value	Color/Hue	Orientation	Texture/Pattern
						
Associative Selective Ordered Quantitative	Selective Ordered Quantitative	Associative	Selective Ordered	Associative Selective	Associative Selective	Associative Selective

Figure 2.11: Visual Variables and their characteristics, adapted from Carpendale [18], Roth [46] and axis map website [5].

2.1.6 Information Visualization Enabling Technologies

Visualization systems and frameworks have been developed to facilitate the development of Info-Vis interfaces. Systems like the InfoVis Toolkit, Protovis, and Document-Driven Documents (D3) aid in creating different applications [38].

The InfoVis Toolkit is a tool built on Java that supports the creation of two-dimensional information visualization applications and components for different data structures. The system relies on the Java Swing GUI and is composed of five main sections: tables, columns, visualization

components, and input/output. The visualization components map data into visual shapes. The InfoVis Toolkit supports specific data structures: tables, trees, and graphs [24].

ProtoVis is an extensible framework with which it is possible to build visualizations by constructing graphical primitives, called marks, including bars, lines, and labels. The existence of these building blocks provides many possible combinations. Marks are generated by mapping the data to visual properties, e.g., position, color. ProtoVis is built with JavaScript and has rendering options for SVG, HTML canvas, and Flash. [12]

D3 is a JavaScript-based library that allows the creation of interactive visualizations on the web. With this toolkit, it is possible to bind selected data to a Document Object Model (DOM), providing “efficient manipulation of documents based on data.” D3 supports large datasets and interaction, with a lightweight and fast end-result. It also supports the reuse of code through the creation of models, both official and created by the community. With this library, graphical resources are built using web standards: HTML, CSS, and SVG. This makes integration with other technologies seamless and gives the ability to, for example, edit an SVG in an external spreadsheet. These characteristics create a flexible and easy-to-use tool to generate data representations.[3] In Figure 2.12, there is a selection of graphics created using D3.

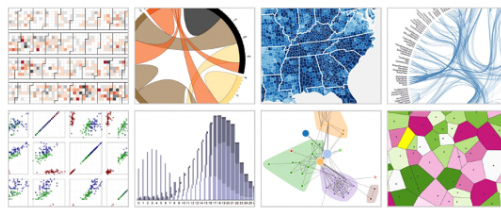


Figure 2.12: Examples of interactive visualizations created using D3. Liu, S. et al. [38]

2.2 HCI Testing

User testing is the kind of usability testing that requires users to engage with an interface directly. In these evaluations, users are required to perform a number of tasks while their interaction with the application is being recorded. Several metrics are considered, such as the time required to complete a task or the number of errors encountered [10].

In order to produce the desired results, it is necessary to plan the experiment carefully. It is necessary to decide on the participants of the tests, measured variables, experimentation method, and others [23].

Interviews and questionnaires consist of query techniques that are used to collect information about tasks and other metrics [23].

2.2.1 Number of Usability Testers

Determining the number of usability testers is an important concern that affects the cost of the study and the scientific results. This translates into having sufficient users in order to cover a

complete evaluation of the desired interface and no users that were excessive to the study [10].

According to Nielsen et al. [42], the relation between detecting usability issues and the number of users tested can be modeled using a Poisson distribution. The obtained model can predict the number of evaluations that enable the best cost/benefit ratio. In a medium-sized example of usability testing, sixteen users would be cost-effective, and the optimal ratio would be at four evaluations.

The type of application and context in which the evaluation is performed can greatly affect the necessary number of users. For example, in the case of websites, it may be required for the users to make several personal choices, which can conduct to many possible pathways and errors to be discovered [10].

Despite the proposed numbers above, this subject still needs further research and for which there is still no certainty [10].

2.2.2 Post-Task and Post-Test Questionnaires

Usability test questionnaires can be divided into two categories: Post-Task Questionnaires and Post-Test Questionnaires. As indicated by the denomination, Post-Task Questionnaires are questionnaires performed by the user just after completing a usability test task. Post-Test Questionnaires are questionnaires performed by the user at the end of the test, after completing all the charges. It can be interesting to use both these metrics with a usability test, as it can help the precision of the results. [35]

There are several relevant questionnaires, such as the SEQ (Post-Task Questionnaire), the SUS (Post-Test Questionnaire), and the NASA-TLX (Post-Task Questionnaire) [35].

2.2.2.1 NASA-TLX

The Nasa Task Load Index is a scale-based questionnaire developed to measure the perceived cost of accomplishing a task, or the workload, by a user right after its completion [31]. Nasa TLX measures this workload with a basis on six different human variable factors, whose scale begins at low and can go up to high with flexible level increments [31]. The first scale, mental load, corresponds to the amount of mental activity, such as thinking, necessary to perform the task. Analogously, the second factor, physical load, corresponds to the required physical effort exerted to fulfill the assignment. The third scale corresponds to the temporal demands of the task, which can be regarded as the time rate at which the task occurred and the relative time necessary for its completion. Frustration is the fourth factor to be measured and can be interpreted as how irritated and demotivated versus content and confident the user feels while completing the task. The fifth scale implemented is effort, in the sense of the difficulty of the task. Finally, the performance is measured in terms of the user's perceived success and satisfaction with the completion of the presented task.[31] [30]

An example of a NASA-TLX can be observed in Figure 2.13.

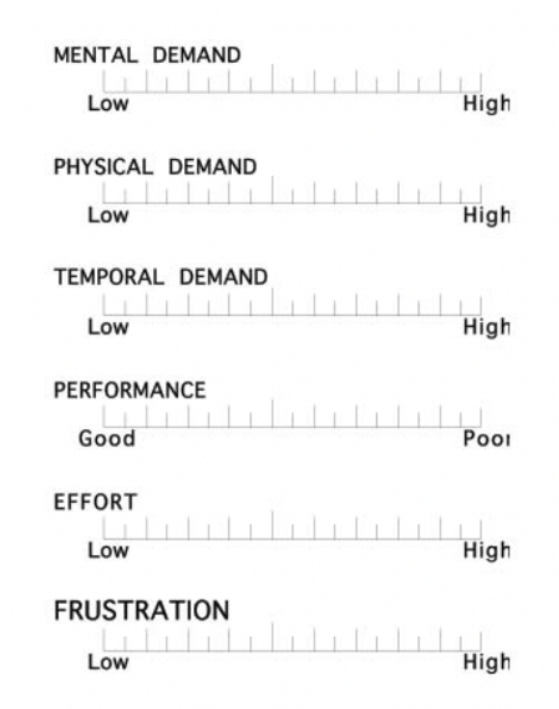


Figure 2.13: Illustration of the NASA-TLX scales, as seen in Hart [31].

2.2.2.2 System Usability Scale (SUS)

The SUS, or System Usability Scale, is a tool developed to assess the global perceived usability of a tool in a short period of time, that being the time of taking a questionnaire [14].

SUS is composed of several statements that use Likert scales to attribute values to how much a participant agrees to them, as seen in Figure 2.14. These statements cover several aspects of the usability of a system, which ensures a high degree of validity for this usability measuring system [15]. This system is meant to be used after participants had a chance to interact and experiment with the tool being tested. There is also a calculation that can be done to obtain a score for each of the SUS questionnaires, which ranges between 0 and 100 [15][14].

2.3 Immunology

2.3.1 Immunology Definitions and Concepts

In this section, there are going to be described immunological concepts and definitions that are relevant to the context of this work.

Our immune system's main goal is to defend our body from infections by identifying and eradicating foreign molecules. It is also responsible for stunting the growth of tumors and repairing damaged tissues. There are two kinds of immunity: innate and adaptive. The innate immune response provides immediate and generalized protection against microbes [6].

	Strongly disagree						Strongly agree
1. I think that I would like to use this system frequently	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2. I found the system unnecessarily complex	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3. I thought the system was easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4. I think that I would need the support of a technical person to be able to use this system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5. I found the various functions in this system were well integrated	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6. I thought there was too much inconsistency in this system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7. I would imagine that most people would learn to use this system very quickly	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8. I found the system very cumbersome to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9. I felt very confident using the system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10. I needed to learn a lot of things before I could get going with this system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 2.14: The System Usability Scale Questionnaire, extracted from Brooke, J. [14]

Lymphocytes are a type of white blood cell and constitute the adaptive immune response, which is more specialized and develops slower [6], [9]. Lymphocytes are divided into B-cells and T-cells. T-cells are responsible for cell-mediated immunity, that is, they interact directly with other cells of the immune system and with infected cells. B-cells are responsible for producing antibodies [9].

Antibodies, also known as immunoglobulins, are proteins produced by B-cells and can exist both as bound to the B-cell membrane (B-cell receptors) and as individual molecules [9], [36]. B-cells produce antibodies in order to respond to an invading pathogen or antigen, the molecules that induce an immune response [36]. They are meant to bind specifically to it in order to either deactivate it or mark it for destruction [9].

The immunoglobulin is a Y-shaped molecule, with the antigen receptors located at the end of both arms. Each arm is composed of a heavy chain and a light chain, which contain gene segments [36]. The variable regions are located at the end of each chain, and the antibody-binding region consists of the pairing of the variable regions of the heavy and the light chain [36], [44]. In addition to the variable regions, light chains contain a constant region, and heavy chains contain three constant regions. The constant region of the heavy chain (Framework region - FWR) has the information that determines the antibody function and, consequently, the antibody's class [36], [53].

The variable regions have three complementarity determining regions, CDRs 1-3, which determine the function/capability of the antibody to bind to an antigen [36], [53]. Genes are regions of DNA (composed of four types of nucleotides - A, C, G, or T) that translate a specific hereditary characteristic, usually corresponding to a single protein or RNA [9]. V(variable), D (diversity), and J (joining) are gene segments present in the CDR regions' heavy and light chains (light chains only have V and J) [36], [53], [52].

In single-cell differentiation, these genes suffer rearrangements (VDJ joining) in order to encode a new heavy-light chain pair which produces a different unique B cell receptor (BCR), or antibody [36], [53]. These rearrangements produce a large and finite number of unmutated antibody structures that correspond to the germlines and compose the repertoire [52].

From the diversity generated through this process, B-cells are selected through their receptor's function, which is determined by the proper folding of the framework region of the BCR and the ability of the CDR regions to bind to an antigen [53].

Clones are groups of cells that derive from the same ancestor and share the same unique BCR (germline). The population of B cells, or repertoire, which is divided into clones, has all its changes and diversification (somatic mutations) occur due to two types of competition: clonal shifting and clonal drift. Clonal shifting is the competition that occurs between different clones. Each clone suffers changes in its original sequence, called mutations, producing mutated cells or mutants. Clonal drift is the competition that occurs between mutated cells that are of the same clone. This competition and mutations constitute the affinity maturation of the B cell population.

The single B cell differentiation, together with the repertoire competition, are the processes behind the somatic selection that composes the adaptive immune response [53].

From the mutations that occur in each clone, it is possible to establish a lineage of B cell receptors that share the same ancestor, which can help us understand the evolution of the clones [32]. These lineages seem to be suitable for phylogenetic analysis and, therefore, can be represented with trees. The root node of the tree would correspond to the germline of that clone, and the nodes that descend from it correspond to the mutated sequences. These nodes are connected by edges or arrows that represent the direction of the evolution and the existence of mutations [32].

2.3.2 AIRR Community

The Adaptive Immune Receptor Repertoire (AIRR) Community is a research-driven group that is attempting to establish, using AIRR-seq data, community-accepted standards for data and metadata to aid in studying B-cell and T-cell receptor repertoires. AIRR-sequencing (AIRR-seq) consists of high-throughput sequencing of DNA and RNA of B-cell and T-cell receptors, allowing to recover immune repertoire data in great detail. AIRR-seq can be the answer to understanding various behaviors of immune responses in situations such as autoimmunity, cancer, infectious diseases, and vaccinology [1], [20].

2.3.3 AIRR Community

The Minimum Information about Adaptive Immune Receptor Repertoire sequencing experiment, also known as MiAIRR, is a standard for data from AIRR-seq experiments that was born of the necessity of data sharing, interpretation, and comparison for complex immunology data-driven research. The MiAIRR standard does not intend to constrain experimental or analytical methods.

It can, however, serve as guidance during the experimental stage in order to understand if it was collected sufficient and correct essential data [13].

The AIRR Community has created for the standard six high-level data sets, each respecting a different aspect of the study and providing information on it. The six data sets are:

- Study, subject, and Diagnosis
- Sample Collection
- Sample Processing and Sequencing
- Raw Sequence Reads
- Processing of Sequence Data
- Processed AIRR Sequences with Annotations

These data sets are in accordance with FAIR-ness and support the publication, curation, and sharing of AIRR-seq data and metadata. The information contained in these six data sets should allow an experienced AIRR-seq researcher to reproduce the results of a study. The MiAIRR sets are organized chronologically (from study design to data generation), and its focus, instead of data analysis, is mainly experimental setup and data generation [13], [2].

2.3.4 AIRR Data Commons

Due to the increase of data generated from high-throughput sequencing of B-cell and T-cell receptors, it was necessary to establish a standard for data and metadata, as well as for supporting data deposit, curation, storage, and use and implementation tools [20].

The AIRR Data Commons (ADC) is a distributed system of repositories that are AIRR compliant, that is, that use a common data model, query language, and share formats for storage, query, and downloading of AIRR-seq data [1], [20]. The ADC creates the possibility of reusing data from AIRR repositories in order to analyze it in new ways and discover new insights about adaptive immune responses [20].

The technical standards for the AIRR Data Commons require the AIRR Data Model and the AIRR Data Commons API for AIRR-seq data repositories.

The AIRR Data Model was created to solve the issues that were not covered by the MiAIRR standard: while it has the defined sets and provides flexibility so the data can be stored in different repositories, there was no structure or schema and no definition of the relationships between the data elements. The AIRR Data Model also specifies a file format, in this case, YAML/JSON. The relationships between MiAIRR objects use standard terminology (e.g., 1-N) [20].

The AIRR Data Commons (ADC) API is a read-only web API that allows access to querying and downloading AIRR-seq data. In order to be AIRR compliant, a repository must ensure that, independently of the internal structure for the AIRR-seq data, it must return it in the AIRR Data Model schema from the API [20]. Several repositories are a part of the AIRR Data Commons and are even managed by iReceptor [22], such as ImmuneDB [45].

2.3.5 ImmuneDB

ImmuneDB [45] is a MySQL-based system for storing large-scale, fully annotated B-cell receptor sequence data, as well as storing and inferring their germlines, clonal groups, and metadata for each study.

This tool supports importing data with raw FASTA sequence files or with other formats such as Change-O or AIRR-formatted pre-annotated sequences. Data can also be exported data in similar formats, like Change-O, AIRR standardized data, or VDJtools. ImmuneDB is, therefore, easy to integrate with other analysis tools and can be used as a part of processes that include them, other than just as a stand-alone system.

The ImmuneDB process follows a pipeline that enables the creation of a database with processed and analyzed study receptor data, where each step is executed through the command line. This pipeline can include raw sequence processing or importing ready data formats, and other processing steps, such as germline and clonal assignment. One of these steps consists of collapsing sequences that only differ from each other in non-relevant positions, resulting in a single set of unique sequences within each sample. Collapsed sequences are considered as duplicate sequences.

The ImmuneDB databases include a web interface that enables browsing the stored data. The interface provides a more intuitive visual analysis, but if a more in-depth analysis is necessary, the command line is the most appropriate option.

The recommended method for installation and usage of ImmuneDB is through the available Docker Image (link) since all the necessary dependencies for generating databases are already included with it.

2.3.5.1 Covid Vaccine Database

The data contained in the covid_vaccine and covid_vaccine_new was extracted from the results exposed in Goel et al.[28]. In this study, an analysis of the evolution over time of antibody and antigen-specific memory B-cells for the SARS-CoV-2 vaccine was conducted on 44 healthy individuals, 11 of whom had a previous infection. Blood samples were collected at four key time points for immunological analysis. The first time point was the baseline pre-vaccine, the second was two weeks after the first dose, the third collection was on the day of the second dose, and the fourth was a week after the second dose. It was demonstrated that the mRNA vaccines administered were effective at producing a strong antibody and memory B-cell response regarding the spike protein and the spike receptor binding domain (RBD).

2.3.6 Immcantation and IgPhyML

Immcantation [4] is a framework that was created in response to the large amount of data from high-throughput sequencing, consisting of a group of tools that enable a complete analysis of AIRR-seq datasets.

One of the packages belonging to this framework, the Change-O standard [29], includes the software IgPhyML [32] [33], which can be used to build phylogenetic trees for clones of B-cell receptors.

IgPhyML [32] [33] allows for the analysis of entire repertoires of B cell receptors that normally contain thousands of lineages, instead of only single lineages. It also implements substitutions that adjust to the context of somatic hypermutation, described in Section 2.3.1, which violates the assumptions of regular phylogenetic programs.

IgPhyML calculates clonal lineage trees using maximum likelihood (ML), with the HLP19 substitution model developed by the authors of this tool, which is the center of this software.

This algorithm sets four groups of parameters that help to differentiate the types of mutations that occurred throughout the development of a lineage and, consequently, aid in the tree building. These seem to obtain, for example, more accurate branch lengths, which represent the number of mutations between two consecutive nodes.

IgPhyML is a software tool that requires significant computational power and is not as effective when running with more than a few thousand sequences. Therefore, it is advised to subsample the dataset being used by filtering and, possibly, dividing it [32] [33].

2.3.7 Newick Format Trees

The Newick format [17] is a standardized system to represent phylogenetic trees. This format simplifies the representation of the tree into a single string that encapsulates the structure of the positions and relationships of the different tree nodes by nesting parentheses and node labels with other punctuation elements. The required elements for each Newick string besides node identifiers are the colon, semicolon, parentheses, comma, and single quote [17] [43].

This format may also contain branch lengths. This string would be obtained by traversing the tree in post order. Therefore, the string should be read and parsed from right to left, where the root node is located.

An example of a Newick string would be: (((One:0.2,Two:0.3):0.3,(Three:0.5,Four:0.3):0.2):0.3,Five:0.7):0.0; [43]. If branch length is included, as seen in the example, each node is represented by a label followed by a colon and the length of the branch. Each pair of parentheses represents a new subtree, and nodes separated by a colon are considered siblings.

2.4 Previous Work in Immunology Visualization

This section intends to briefly describe previous work that has been developed in the field of Information Visualization towards Immunology.

2.4.1 AncesTree

AncesTree is a graphical user interface built as complementary to the trees built with, for example, IgPhyML, that allows researchers to interact and visualize phylogenetic/lineage trees [26].

AncesTree uses two different tools to generate phylogenetic trees: Dnaml and Immcantation. With the Dnaml tool, the Dnaml output text file is parsed by AncesTree. With the Immcantation pipeline, it is necessary to use as input a Change-O file (AIRR format), an IgPhyML file, and the correspondent FASTA file [26].

AncesTree is then capable of processing the input files and generating the correspondent tree, which is visualized in the GUI [26].

2.4.2 Ggtree

Ggtree is a package developed in the R language that allows visualization of phylogenetic trees, while annotating them with data from different sources. Several types of tree formats are supported as input, such as Newick, nexus, NHX, phylip and jplace. This package allows for the users to annotate and color the visualized tree, as well as interact with the render by zooming, collapsing or highlighting parts of the tree. Nodes can be annotated with numerical or categorical data [55].

The phylogenetic tree and data can be stored in a graphical object called ggtree, allowing for reusability and reproducibility. The ggtree object can be rendered into a static image.[55]

2.4.3 iTol

The Interactive Tree of Life (iTol) is a browser based tool for visualizing trees, namely phylogenetic trees. This tool also enables interaction and annotation of trees with several types of data. This tool is implemented with pure Javascript and HTML5. iTol supports the more common formats of phylogenetic trees such as Newick and Nexus. All the data used to annotate trees is provided with plain text files. This tool provides different tree display formats, such as radial or rectangular phylograms, and allows moving and deleting single nodes as well as collapsing or deleting parts of the tree, either manually or automatically, based on defined parameters. Trees can also be pruned based on a text file node data and re-rooted to any node. Tree leaf nodes can be sorted with different criteria. [37]

2.5 Summary

This chapter provides an overview of the existing literature on Information Visualization, on HCI testing, on the relevant immunology context, and on previous work on visualization for immunology. Regarding information visualization, there was first described a definition of this scientific field and explained the information visualization pipeline. Following these sections, there

were explained two taxonomies: a data-based and a task-based taxonomy. There was a particular focus, with the data-based taxonomy, on hierarchical and temporal visualization techniques, which are the most relevant for this study. There was also an analysis of visual elements and variables that can be used to define a visualization solution. There was also a presentation of Visual Variables and their characteristics.

In relation to HCI testing, there was a focus on the number of participants needed for an evaluation process as well as on questionnaires (post-task and post-test) that can be given on this process.

To provide information on the immunology context of this work, the main concepts relevant to understand the background of this project were detailed. There was also a description of the AIRR Community standards, including tools such as IgPhyML and ImmuneDB, which are also important for this study. Finally, the previous relevant work on visualization for immunology, such as AncesTree, was briefly described.

Chapter 3

Conceptual Solution

This chapter describes the problem that is addressed by this visualization solution and the path taken to obtain the final idea to be implemented as a tool.

In Section 3.1, the main questions posed by this study will be described, the origin of the identified problem stated, as well as the requirements that are to be addressed in the solution.

Section 3.2 describes in detail the process and reasoning behind the proposed solution to be implemented as a tool in this work.

3.1 Problem Statement

As mentioned in Chapter 2, the available high throughput sequencing technology generates a great amount of immunological cell data. This creates the difficulty of interpreting and analyzing the evolution of an immune response especially when looking only at a dataset. This work concerns itself particularly with the evolution of B cells. B cell receptors are organized in clones, according to the mutations that occurred in their genetic sequence. The data available represents sequences of B cell receptors of multiple individuals obtained at different time points, resulting in a large and diversified number of clones with distinct characteristics, such as disparate tissue origins. Given the presented difficulties and the importance of understanding and interpreting this data, lineage trees that demonstrate this evolution must be visually represented to make the information more accessible and intuitive. In order to obtain a global understanding of the evolution of a B cell population across time, there are two main concerns to be addressed: lineages must not be isolated and must be visually represented in a way that creates a relationship between trees from different time points and with the single tree that comprises all the time points present in the data, and it is necessary to have an overview by visualizing the diversity of the clones in an immune response, which is an indicator of successful the later was.

Section 2.4 presents several visualization solutions developed by other studies, such as Ancestree [26], that only allow the analysis of isolated lineage trees, mostly for combined time points. These visualizations, in the perspective of this dissertation, fail to consider the evolution of clones across time and are not very clear on conveying where in time a mutation occurred. Furthermore,

not all of these visualizations enable interaction with the lineage tree, or introduce other meaningful information and metadata related to the clone and its mutants, beyond mutation-related data. Moreover, as mentioned previously, the vast majority of visualizations do not succeed in giving an overview of the evolution of sets of related clones (from the same dataset) and consequently do not offer sufficient information to create an understanding of the presence/frequency and the number of clones over time.

Consequently, this work proposes to address the following 3 identified issues that currently are not being considered by existing visualizations:

- **Representing both separate lineage trees for each time point and single all time points lineage trees for each clone, that can be related to each other and compared for analysis.** Considering discrepancies between lineages for the same clone across time points by creating separate lineages for each time point may be detrimental to understanding the evolution of B cells. The previously studied existing visualization solutions only support the visualization and analysis of one lineage tree representation at a time, most cases of all time points of a single clone. I.e., for each clone, only one tree can be visualized and cannot be split across time points. In order to better analyze and comprehend how a B cell is mutating and evolving it is necessary to visualize the lineage trees for each available individual time point. Given how these trees are built, the relationships between mutants of the same clone can become different if a single time point tree is being considered against an all-time point tree. Visually representing similarities and changes in the trees across time points will enable their comparison and easier identification, for a more in-depth analysis of the evolution of a certain clone for a chosen subject.
- **Presenting an overview of clone evolution for an entire or filtered dataset.** The lineage tree visualizations aforementioned allow the representation of single trees that respect to a preselected single clone. The possibility of visualizing the evolution of certain characteristics, such as size, of either a set of clones or the whole dataset has not been found in the existing visualizations studied here. Visualizing several clones in a single view can provide a better perspective about the clones that exist on the dataset and their characteristics, so it becomes easier for a user to select the clones of greater interest for a more detailed analysis, namely of lineage trees. In order to obtain sets of clones, considering the significant size of immunology datasets, filtering options must be provided. These filters should contemplate relevant metadata and other characteristics obtained from the dataset, to aid in the selection of clones that best suit the intended specific analysis.
- **Embedding metadata analysis in the lineage tree visualization.** Each node may also have additional metadata and information associated, that, if represented visually with the tree and its nodes, can convey insight that furthers the analysis beyond mutations and time points. For example, if we take into consideration the tissue/origin of the clone, and visually represent it on the lineage trees of a determined clone, it is possible to understand how the location of the clone also evolved throughout time. Not contemplating more information

other than mutation-related data when analyzing a tree, could hinder the user from getting an accurate perspective on the evolution of B cells.

Considering the identified issues, it is necessary to address the requirements for the visualization solution proposed here. Therefore, to answer the questions that arise from the identified problem it is essential to:

- Create a filtering system to obtain relevant sets of clones to visualize.
- Develop a visual time narrative for a dataset clone presence overview.
- Build and visualize clone lineage trees, both trees with all nodes from a single moment in time and trees that represent nodes from all the available time points from the dataset in use.
- Implement significant visual elements to represent time and other metadata, in order to establish connections between different trees across time.
- Integrate all the requirements above into a solution with a sequential process that enables interactive visual exploration of lineage trees built with data from an ImmuneDB service.

It is important to stress that this work is a first approach to answering the questions set by this study. Conceptualizing and developing an experimental functional prototype has a highly experimental character, which justifies the need for many iterations of improvements to achieve the ideal solution.

From the requirements identified, the authors of this study consider that a solution should have four main focus points: to take a pre-selected dataset, to create a limited number of relevant clone filtering options, to render the evolution and presence of all clones over time, and to render lineage trees that take into consideration discrepancies between different structures for the accounted time points and allow the analysis of related metadata. The selected dataset in this study regards sequencing data from a COVID-19 study [28].

Finally, in accordance with all the previous statements already presented, the solution proposed in this work intends to answer the following research questions:

- How can we effectively create a global overview and understanding of the evolution of a B cell repertoire across time?
- How can we aid reasoning and understanding of the evolution of B cell clones, and the quality of an immune response, through visual representation of both lineage trees for isolated time points and single trees for all time points per clone, since each type of tree is constructed differently?

3.2 B-cell Receptor Evolution Visualization Pipeline

It is largely regarded as of great importance to trace out a conceptual pipeline that describes the process of mapping and rendering initial data into visual forms. As previously presented

and described from different sources in Section 2.1.2, a visualization pipeline should detail each step and transformation taken to obtain the final translation of raw data into a visual mapping. A pipeline description is fundamental to understanding how to approach a visualization solution that addresses the identified research questions and this section presents a version adjusted to the present study and its objectives as a starting point for conceptualizing a solution. This visualization pipeline is represented in Figure 3.1.

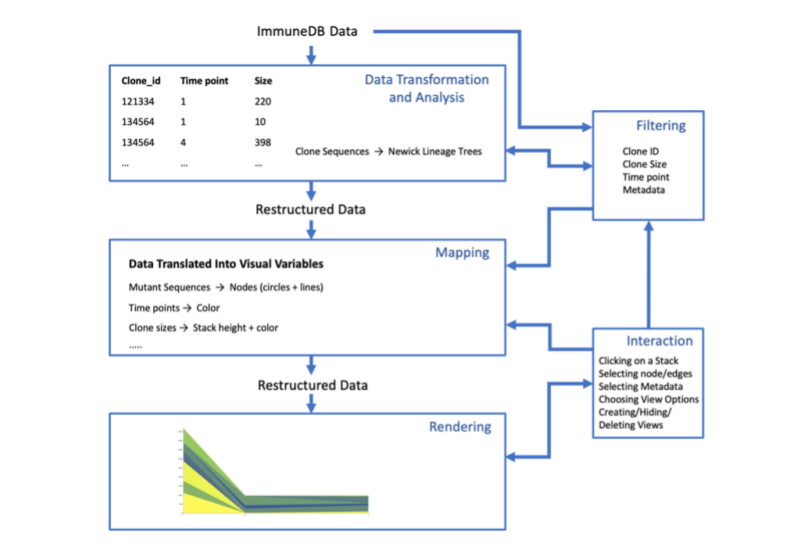


Figure 3.1: Adapted Visualization Pipeline

The initial input of data, or Raw Data as described in Section 2.1.2, is the starting point of the visualization pipeline. The initial immunology data required for this project is obtained from an ImmuneDB database instance, as mentioned in Section 2.3.5 and characterized in Section 4.3. The Raw Data (ImmuneDB data) is in a structure defined by the ImmuneDB dataset schema.

Although a part of the Raw data structure can be directly queried from the database and passed to the next stage - this is the case for the majority of the metadata - part of the structure of the dataset needs to go through the second stage of the pipeline - Data Transformation and Analysis. All data that is mapped into graphical elements, the stacked area chart, and lineage trees, must be restructured and altered to become the input that is transformed into visual structures. For example, the data given as input for the stacked area chart (Figure 3.1) needs to be reorganized into a data structure that maintains direct correspondence between the identifiers of the clones of each stack, each possible time point, and the analogous size. Additionally, the data necessary to build trees with the IgPhyML software (Section 2.3.6), obtained from the data stored in ImmuneDB, goes through several data transformation iterations until it is structured and ready to be used as input.

The filtering stage, contemplated as the third stage of the visualization pipeline in Chapter 2, where the data is categorized and selected for visual mapping can, given the structured nature of the initial data, happen both before and after the second stage, as well as at any point that interaction with the rendered tool creates a data selection and filtering. Some data must be analyzed and transformed in order to be selected, for example, when selecting a clone to be visualized into

lineage trees from the stacked area chart. However, some data should only be converted into a direct mapping of visual elements after going through the filtering step, as seen with the selected clones to be rendered into the stacked area chart. Given the non-linear form of filtering, the visualization pipeline was adapted to better contemplate all the different moments in which this step can occur, as seen in Figure 3.1.

After obtaining the precise and structured data meant to be visualized (Focus Data) the Mapping stage takes place, where, as seen in Section 2.1.2, this input is transformed into visual primitives that compose more complex visual structures (Section 2.1.6). The chosen visual variables, which are detailed in Section 3.3, are meant to create easily interpreted visual metaphors that simplify the process of data analysis. An example of visual mapping in this solution is the transformation of the relationships between the mutants of the same clone being transformed into edges (lines) that represent the mutations, that connect the nodes (circles) that represent the mutated sequences, comprising a lineage tree.

Finally, the abstract visual variables attributed in the Mapping stage are Rendered and expressed into concrete images. After this step, the user can interact with the generated graphics, which can influence most steps of this pipeline. For instance, interaction with the rendered stacked area chart and consequent selection of a clone to further analyze will interfere with all the stages of the visualization pipeline, as filtering, new data structuring, new mapping, and rendering will occur for the lineage trees that respect the aforementioned clone.

3.3 Conceptual Proposed Solution

This section describes the process of development and justification for the chosen visual elements and layout of the proposed solution presented in Section 3.4.

The initial proposition for a solution to the identified problem aforementioned in Section 3.1 was created in the Artathon 2021 (unpublished document), which is an event where immunology-related information visualization solutions are conceptualized, according to the proposed questions/statements. In this edition, the Meeting in Time (unpublished document) visual concept was created as a possible solution to visualize large amounts of immunological data coming from high-throughput sequencing.

Meeting in Time proposed a visualization pipeline comprised of four main parts: (1) a filtering stage, where data would be filtered according to certain criteria and clone characteristics, which would enable (2) the rendering of a visualization of time evolution that could then (3) be interacted with to further filter clones in order to, finally, (4) visualize more clone and lineage tree features.

3.3.1 Data filtering solution

The filtering component comprises both mandatory and optional filters, and the latter can be enabled or disabled. The selected filters are cumulative and when submitted, create a subset of clones from the original dataset. Some of the most important filters, given the objective of visualizing clone presence and evolution across time, are clone size/frequency, time periods/points,

and mutation rate in a clone. A mockup for this stage was developed, with several example filters and their described mechanism, and can be observed in Figure 3.2.

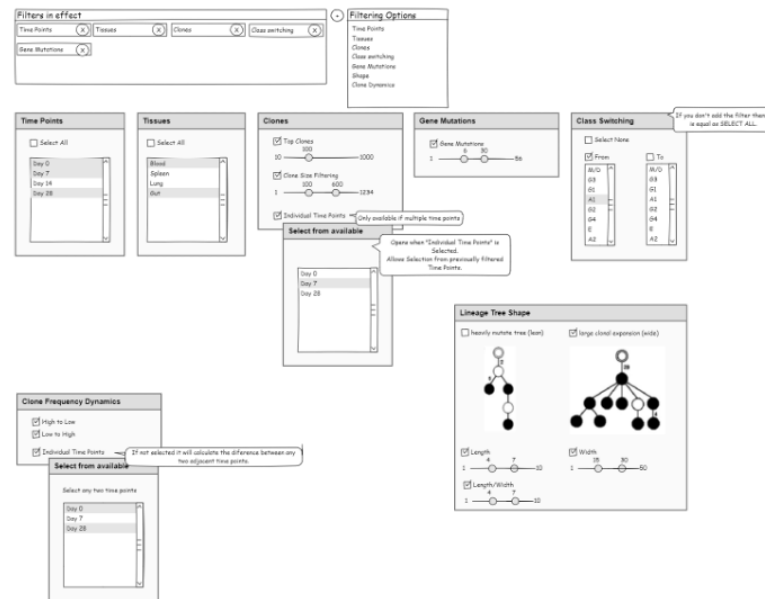


Figure 3.2: Filtering Stage Developed in the 2021 Artathon.

3.3.2 Clone size evolution in time visualization solution

The temporal data visualization is a visual mapping of the set of clones retrieved according to the filters selected in the previous stage. In view of the proposed mapping across time and based on the task-based information visualization taxonomy described in Section 2.1.3, a time series visualization is considered the most fitting type of visualization technique. Since the objective of this stage is to show an overview of the evolution of many clones over time, stacked area charts are an appropriate technique to meet the desired requirements. Stacked area charts (see Section 2.1.3.2) allow having multiple time series in a single visualization by stacking them while sharing the same timeline. Figure 3.3 shows the concept developed for the overview of time evolution. The desired value to evaluate across time, measured by the size of each stack, should be a possible measure of the size of the clones, such as the total number of copies found for a clone. In order to further convey valuable information in this component, not only the size of each stack but also the color is mapped visually to introduce either a more intuitive or more complete analysis. There can be several options for color in each stack, such as grouping clones by metadata parameters, their tissue origin for example, or visually establishing a comparison with relative size between stacks. For the present solution, color is only used to convey size in relation to the size of other stacks. A chosen color (blue) represents the lower size value limit, and another color (green) represents the higher size value limit. Each stack has attributed a color according to the proximity to each end of the color spectrum, obtained by interpolating the colors. The color in each stack reinforces

the idea of size, which can be important in order to select more relevant clones for the following analysis.

3.3.3 Lineage tree visualization solution

The selection of a stack in stacked area visualization triggers the next stage: the lineage trees visualization. A lineage tree is composed of nodes and edges. The first represents either mutated collapsed sequences (mutant) or an inferred mutant while the last represents the mutations that occur between two connected mutants. The germline is represented by the root node. By selecting the clone for a certain time point, a corresponding lineage tree would be rendered. The main thought would be to provide a Focus+Context technique (see more in Section 2.1) to hold all the visualizations in a single view.

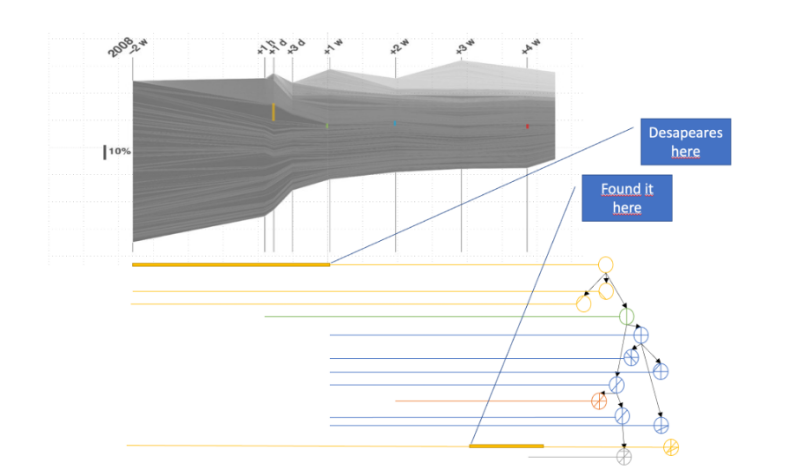


Figure 3.3: Time series visualization and interaction first consideration; lineage tree initial visualization consideration from Artathon 2021.

The concept illustrated in Figure 3.3 shows a single lineage tree that aligns each mutant with its time point. This mockup solution, while it does represent time points through color and provide visual information on which time point a node is first found with the horizontal lines, it does not contemplate the comparison between the structure of single time point lineage trees and all-time points lineage tree per clone. This solution would also introduce considerable visual clutter when analyzing large trees with more clonal expansion.

As a result, the lineage tree concept was further developed. The first iteration passed by building a simple tree with a classic hierarchical shape, with the identifier of each mutant sequence placed next to each node. A visual result of the first iteration of solution development can be observed in Figure 3.4.

In this lineage tree mockup, it is easily identifiable that the sequence names are not readable and that there are no visual cues for the time points to which each node/mutant belongs. Consequently, the next iteration sought to appoint visual elements that conveyed meaningful information about each node's temporal data.

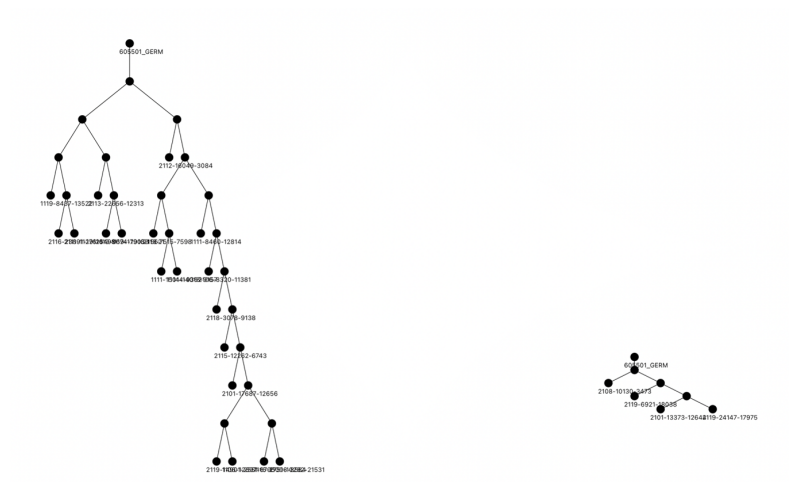


Figure 3.4: First iteration of the lineage tree mockup.

Representing time in each lineage tree is one of the crucial visual elements in this prototype, in order to create a comparison and a relation between lineage trees of different time points and lineage trees that comprise all time points. According to visual element theory (see more in Section 2.1.6), color is often used as a visual variable to categorize and label, since color hue is considered in the literature (Section 2.1.5) as a selective variable, which implies that a natural grouping of elements occurs. In this particular case, it is intended to visually group all nodes that belong to a specific time point, which can be observed in the mockup introduced in Figure 3.5. A color is, then, attributed to each node according to the time point or time points it belongs to. When a node is found in more than one time point, the color assigned to it is an interpolation or average of the colors associated with each of the time points that the node exists in, in order to also provide a visual mapping of its presence in the specific individual points in time. This grouping by time point is the key to identifying tree substructures that may be common between trees easily and finding relevant comparisons in the different trees across time.

Besides the children nodes, color and shape are used to identify the germline in the root node, by making it uniquely distinct from all the other nodes.

In the process of building lineage trees, there are also nodes that are inferred by the tree-building component. These nodes do not have an identifiable sequence associated with them, and therefore, do not have a specific time point either. These nodes, as seen in Figure 3.5 for example, are visualized with a smaller radius and possess the same neutral color that edges do, a color that is not associated with any particular time point.

Color as an identifier, however, has some limitations to be taken into consideration, particularly color blindness.

In a next iteration of the lineage tree development concept, to reinforce the information conveyed visually by color, a pattern/texture was introduced to the nodes to also represent time, represented in Figure 3.6. The chosen pattern to further connect each node to a time point visually consists of lines crossing the circular shape of the node in different directions, in itself also a visual

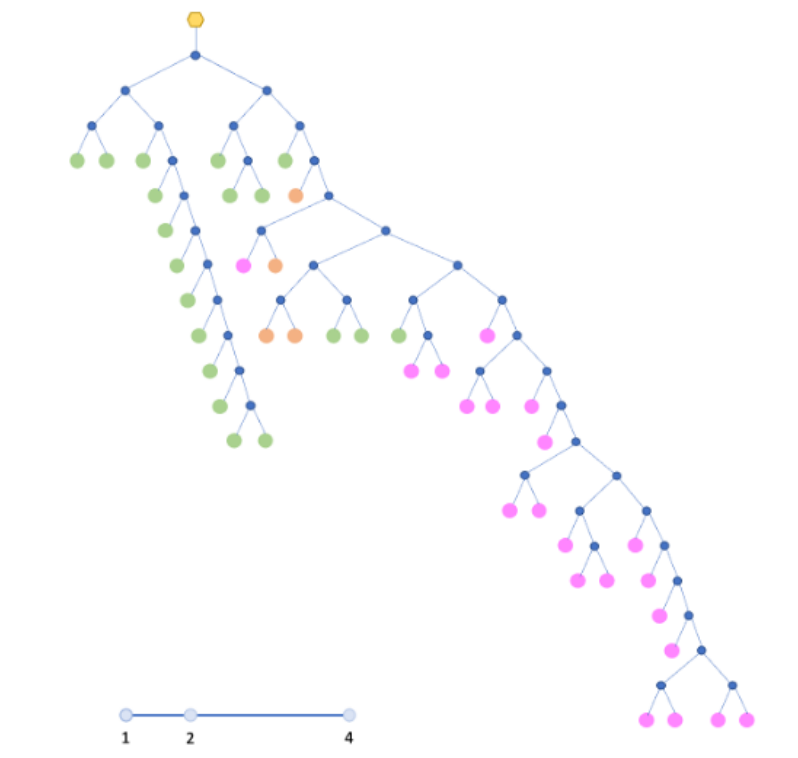


Figure 3.5: Color as a time-defining visual variable in a lineage tree.

variable that can be selective and associative (see more Section 2.1.5). A single line in a specific direction (vertical, horizontal...) is associated with a distinct time point, and consequently, with a color as well. When a node is found at different points across time, there will be one line per time point present in the node. In this particular situation, the presence of a visual pattern is of considerable importance to identify the distinct points a node exists in, in light of the fact that color interpolation is not a very straightforward form of distinguishing the different colors analogous with each time stamp. Both the border of the node and the pattern lines drawn are colored with the same hue as the node color but with a different, darker, value (lightness), to keep a consistent visual mapping.

As mentioned in the previous section, it is of considerable importance to understand the discrepancies that may be present when looking at trees that are built to represent a clone across several or all possible time points or a set of trees that each represent a single time point. In order to achieve this, two possible tree visualizations are considered for each time point: a tree built with only the sequences that appear on the specific time and a tree that comprises all of the sequences for all moments in time, but all the nodes that don't appear on the selected time point(s) are grayed out. This idea is also represented in Figure 3.6. Disabling the nodes that are not a part of that time enables the user to easily find the nodes and tree substructures that are relevant to the intended comparison in the all-time point tree. A tree with nodes from all the time points but with disabled nodes from non-selected time points is called an integrated tree.

It was also relevant to encode visually some essential information about each node/mutant

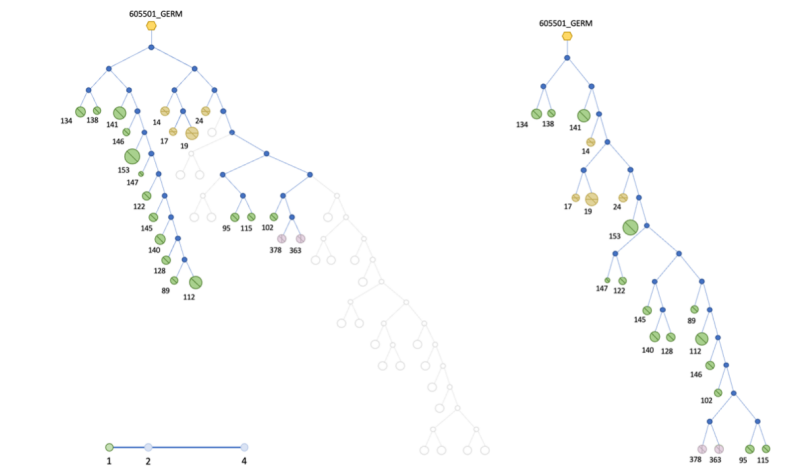


Figure 3.6: Lineage Tree Mockup with Time Points identified by color and pattern; Integrated and Single trees.

and edge visually, beyond time. The most relevant characteristic to represent for each sequence, i.e., for each node, is its prevalence. The most directly correlated visual variable to represent it is size, if accounted that the literature regards this as the only quantitative visual variable ((see more Section 2.1.5). Thus, the diameter of each tree node can be directly related to the number of copies of each sequence, as shown on the nodes represented both in Figure 3.6 and 3.7. Regarding the edges, considering these represent the relation between mutants, the most significant value to map visually is the number of mutations. Given that it is necessary to use a quantitative variable to represent a number, size is once more selected as the most appropriate visual variable. In the case of the edges, the number of mutations that occur between the nodes linked by each is represented by the thickness of the line of the connection.

In order to obtain further information about a certain node or a certain edge, both these elements can be selected. As represented in Figure 3.8, node selection is visually represented by a second concentric black circumference, larger than the node, and it appears each time a node is interacted with. Furthermore, interacting with an edge and selecting it is represented visually by turning this edge black from its default color and transforming the line into a dashed line. Both these visual representations enhance the selected element and give it visual importance in relation to the other tree components. This interaction enables the display of details related to the selected node, such as the collapsed sequences ((see more Section 2.3.5) on that node, or edge, such as the number of mutations. An initial possibility for this representation is presented in Figure 3.7.

3.3.4 Metadata visualization

When describing the stated problem for this study in Section 3.1, one of the identified questions to be addressed consisted of embedding metadata information into the tree visualization. Mapping information into the nodes themselves beyond the data already mapped, would not only not be easy but it would not be visually efficient, creating considerable visual clutter. As a result, the solution

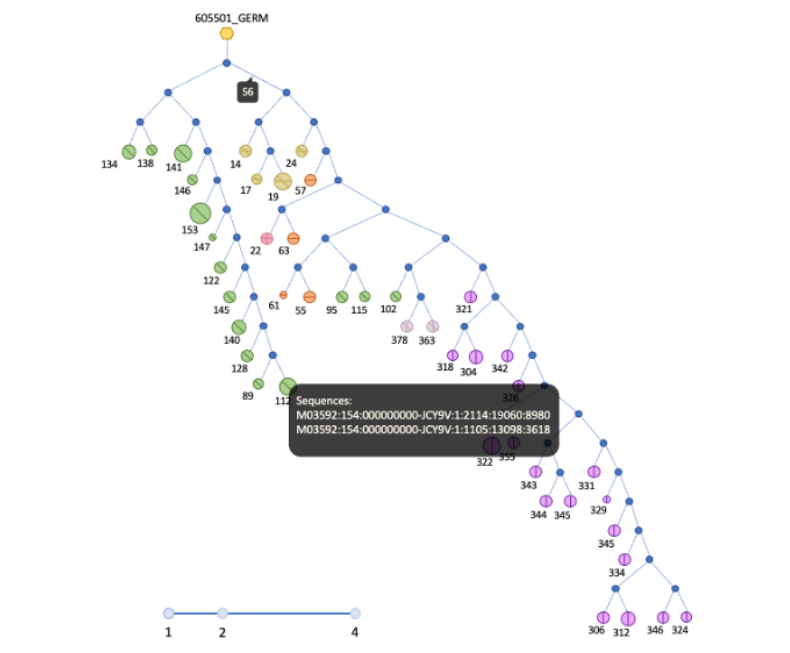


Figure 3.7: Possible solution for tree element's detail visualization.

found to associate the desired metadata to a node was to add up to five symbols with distinct shapes next to each node, as shown in Figure 3.9. Shape (see more Section 2.1.5), according to the visual variables studied in Chapter 2, is an associative variable, permitting symbols with the same shape to be perceived as in the same category, especially when it is the only variable being considered. This characteristic described by the literature, makes shape a fitting variable to convey metadata categorization.

Metadata is composed of different categories, each with several distinct values. In order to correspond each shape to a metadata value, the metadata categories must be chosen initially. Subsequently, there must be an association made between each desired value to be mapped into the tree and a shape. Considering the limitation imposed by the number of symbols, there can only be as many pairs of metadata category-value as the number of possible shapes.

3.3.5 Alternative radial lineage tree layout

In addition to the regular hierarchical shape that has been depicted in the examples mentioned previously in this section, trees can also be visualized in a radial shape. Radial Trees (see Section 2.1.3.1), provide an alternative placement of nodes in the available visual space. Not only can an alternative hierarchical visualization technique provide further and distinct insight into the knowledge conveyed by the tree, but when looking at trees with a very large number of nodes a radial format can provide a better space distribution for a clearer interpretation of the tree structure.

Taking into consideration the importance of establishing comparisons between clone lineage trees from different points in time, identified in the issues that this study intends to fulfill in Section

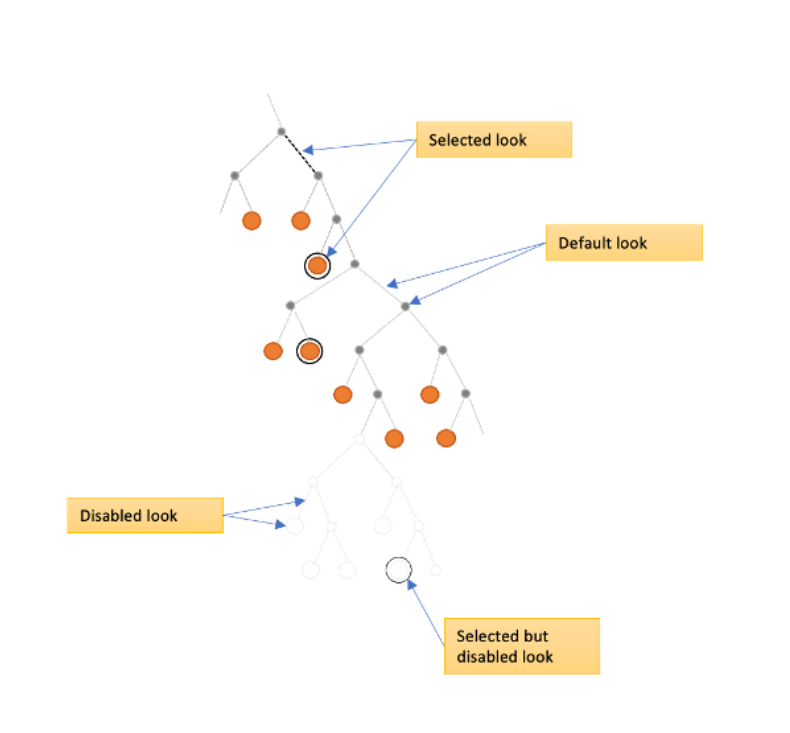


Figure 3.8: Node and edge selection.

3.1, it is important to create a tree layout visualization that enables a visual side-by-side representation of different types of trees on distinct time points. Keeping different options in the same visual space is essential to allow interaction with all the desired trees and establish comparisons between them.

3.3.6 User-Interface layout

Beyond the visualization of the actual lineage tree, and in accordance with what has been described throughout this section about interaction and additional related information, it is also necessary to find a solution that permits altering the overall visual information provided, such as the sequence's number of copies or the number of mutations represented in an edge. The possibility of selection of a node or an edge, mentioned previously, an interaction that displays details from the selected element, also requires that there is dedicated space to exhibit this information that does not cause excessive visual clutter and hinders the analysis.

With concern for the above-mentioned, a conceptual layout was developed to accommodate the requirements mentioned. Figure 3.10 and Figure 3.11 illustrate this mockup.

When the stacked area chart is interacted with and a clone is selected, the user should be redirected to a new interface component, a Window Manager, where new visualizations of the clone's lineage trees for selected time points and with other chosen parameters can be created. Each of these visualizations is represented in a Tree View Window.

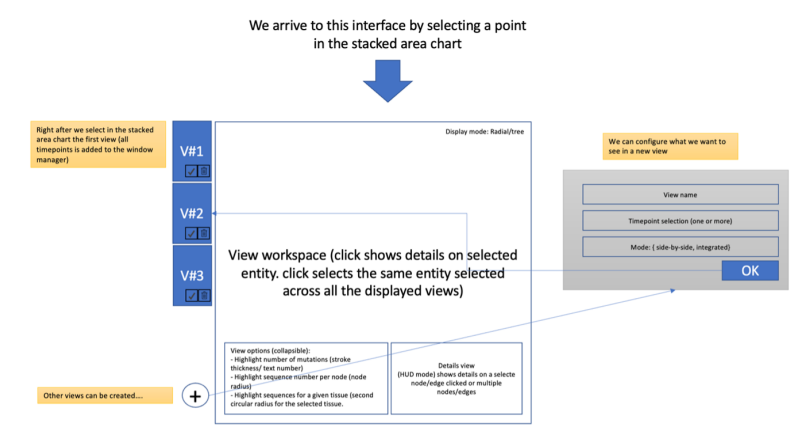


Figure 3.10: Concept for Lineage Tree Visualization Layout and Solution (Window Manager).

is displayed in the Details block of the Tree View. If a node is selected, the Details window names the full identifier for this node in the database, lists the sequences that were collapsed into the main sequence of the node, identifies the size value of this node, and lists all the node metadata pertinent to the selected categories. When in turn an edge is selected, the Details window displays the identifiers of the nodes that the current edge is connecting, the number of mutations that occur between said nodes, and which mutations were identified between the regarded sequences. All of the information displayed in the Details block is necessary for the analysis of the tree and its nodes' evolution, whilst not being possible to be mapped visually into the tree itself.

3.4 Proposed Solution prototype

The problem identified in Section 3.1 and the consequential series of questions that arise from the former, summarized in the hypothesis posed, is solved by the solution consisting of all the conceptual elements described in Section 3.3. Succinctly, the hypothesis states that creating an interactive Visualization of B cell clone lineages at multiple time points, in addition to visualizing them at a given time point, should enable meaningful analysis of the evolution of B cells.

The proposed solution is, therefore, a novel idea for the visualization of immunological data across points in time, that contemplates three components to provide a complete overview and focused exploration. The three components that fulfill the requirements identified in Section 3.1, and described in deeper detail in the previous sections, consist of a filtering stage to pre-select relevant clones, a stacked area chart that provides an overview of clone evolution, and when the latter is interacted with by selecting a stack/clone, a tree visualization interface, or window manager, that grants the ability to create several variations of the lineage trees for the desired time points, enabling a side-by-side comparison. In Section 4.5, Figure 4.3 shows the working prototype version of the proposed solution, with all the previous concepts combined into a single sequential tool.

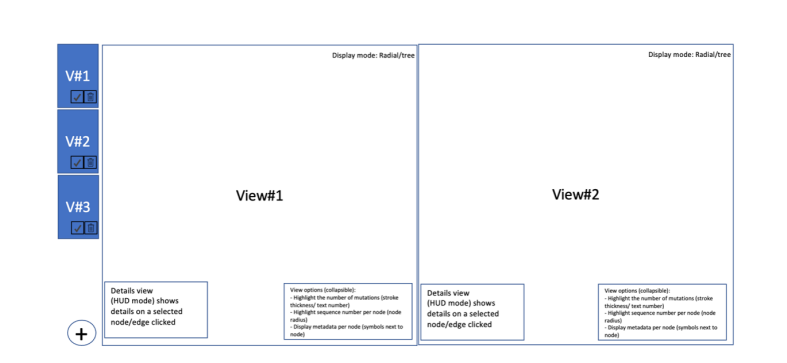


Figure 3.11: Tree Views side-by-side.

3.5 Summary

This chapter details a conceptual solution for the problem presented in this dissertation, which was implemented into a fully functional prototype. This chapter started by identifying the problem to be addressed and the requirements that the concept should fulfill. The identified problem in Section 3.1 consists of two main issues that require a solution: a lineage visualization that creates a relationship between trees from different time points and the tree that comprises all the time points present in the data, and it is necessary to have an overview by visualizing the presence and diversity of the clones in an immune response.

The goal of the solution, based on the specified hypothesis and requirements, is, therefore, to create an interactive visualization that comprises: (1) an initial stage to filter the clones of the dataset in question, based on the desired characteristics, for a more relevant analysis, (2) a time series visualization that enables an overview of the clones across time, and (3) the visualization of side-by-side lineage trees, for each clone, from a single time point or multiple time points, with visual mappings that encode time and other metadata.

From the stated problem, the methodology to reach the proposed solution is presented in this chapter. Firstly, an adapted visualization pipeline is presented in Section 3.2, where the path from data to visualization is characterized for this study. This pipeline begins with ImmuneDB data, which is treated, restructured, and filtered, to then be mapped into visual variables that are rendered into the final visualization. As seen in Section 3.3, the rendered visualization can be interacted with, which can change and filter the data being presented and, consequently, the displayed image. This section subsequently presents all the logic behind the development of the concept, from the filtering stage and the mechanisms and type of filters included, to the time series visualization as a stacked area chart, where each stack represents and is colored in correspondence with the presence of a clone throughout time and to the lineage tree visualization. The lineage visualization is a component comprised of multiple views with varied options for tree visualization for each clone can be created, hidden/shown, and deleted. Each tree is composed of nodes, colored, and patterned according to the time point(s) it belongs to, and edges that represent the mutations between nodes. The tree can be interacted with, in order to display more detailed information, both visually and

in the form of text. Metadata representations are also embedded into the tree, by grouping nodes that possess the same metadata values with matching symbols.

Chapter 4

Visualization Prototype

This chapter presents the process of development of the visualization prototype that implements the conceptual solution introduced in Chapter 3. The first section, 4.1, explains the technology selected for the implementation of the system. The focus of Section 4.2 is the architecture and interaction of the different components of this application. Section 4.3 explains the method and tools used to build the lineages visualized in the last stage of the prototype. Finally, Section 4.4 gives an overview of the implementation of each segment of the application. The prototype and its source code can be accessed through the links in Appendix B.

4.1 Tech Stack

The prototype that fulfills the concept demonstrated in Chapter 3, with consideration for the visualization technologies presented in Section 2.1.5, was developed as a Web Application. Web applications have the advantage of running in browsers, which are supported by a large number of devices and are cross-platform applications, not needing the installation of additional specific software that can cause compatibility issues. Given the designed layout and the flow of the solution presented in Sections 3.2 and 3.3, a Single Page Application (SPA) was the most logical format for the visualization. Thus, the client-side application (Frontend) was developed with the React ¹ framework and NodeJS ², using JavaScript as a language. Considering the browser-supported visualization libraries presented in Chapter 2, the D3.js library ³ was selected for its personalization and interaction possibilities and lightweight format with SVGs. Furthermore, CSS and MaterialUI⁴ were used to customize the appearance of the prototype. Concerning the management and installation of external dependencies necessary to enable the usage of, namely, the D3.js library with the React framework, the Node Package Manager (NPM) ⁵ was used.

¹<https://react.dev/>

²<https://nodejs.org/en>

³<https://d3js.org/>

⁴<https://mui.com/material-ui/>

⁵<https://www.npmjs.com/>

As stated previously, this prototype retrieves B cell receptor data from ImmuneDB [45], which uses MySQL⁶ as Database Management System (DBMS). Since the data obtained from ImmuneDB is of considerable size, it was necessary to use software that aids in data management and manipulation. Pandas⁷ is a popular and well-documented data analysis Python library that satisfies this requirement. This library was used to process and treat data necessary to build lineage trees using IgPhyML [32]. In order to maintain the usage of Python for the RESTful API, the chosen framework was Django REST Framework⁸, which extracts the backend functionalities of the Django⁹ framework. An additional database was used to store the lineage trees in Newick format. This database, matching the ImmuneDB database, is implemented in MySQL DBMS.

Moreover, the application was containerized using Docker¹⁰ technologies – Docker containers and Docker Compose – in order to provide portability and consistency across different hosts. The containerized application includes an Nginx¹¹ server for the client application that connects to the API container, which is served using Gunicorn¹².

4.2 Prototype Architecture

The developed prototype is comprised of three main components: a database layer, an API layer, and a client-side application. A diagram that describes the client-server architecture implemented is presented in Figure 4.1.

From this picture, the database layer is the component that houses the database storing trees in Newick format and the clone, time point, and subject to which each tree belongs. Accordingly, this database has a schema with a single table with four columns: newick_tree, clone_id, time_point, and subject. It also includes the external read-only ImmuneDB database used in this prototype, covid_vaccine_new. The database layer and the API layer communicate through requests from the latter that query the former.

The API layer represents a server component that handles the client requests, applies the necessary logic, and retrieves and returns the data to the prototype's front end. This component has two main Django-based applications: the first handles requests related to the filtering stage and retrieves data only from ImmuneDB, and the second handles requests related to the lineage tree visualization, therefore retrieving data from both databases.

As introduced in Section 4.1, the client application is a SPA (Single Page Application) built using ReactJs, and that constitutes the three interface components of this prototype: the filtering stage, the clone size evolution stacked area chart, and the lineage tree visualizations. These interface components are composed in a sequential form, interacting and sending parameters and data in only one direction, as is depicted in the blue area of Figure 4.1.

⁶<https://www.mysql.com/>

⁷<https://pandas.pydata.org/>

⁸<https://www.django-rest-framework.org/>

⁹<https://www.djangoproject.com/>

¹⁰<https://www.docker.com/>

¹¹<https://www.nginx.com/>

¹²<https://gunicorn.org/>

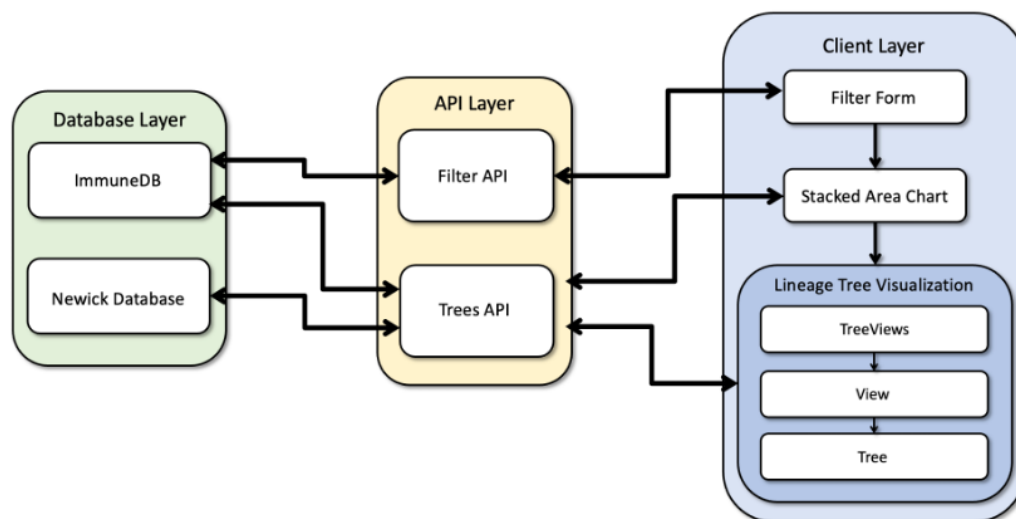


Figure 4.1: Prototype Architecture

The filter form, composed of the Form component, is sent with all the selected filters and corresponding parameters to the Filter API application, requesting as a response all the clones that fulfill the desired parameters.

Every time the form request is sent, the clones obtained are then mapped into the stacked area graph (StackedChart component). In this second interface component, it is possible to select which clone's lineage trees are going to be visualized in the last interface component. In the Stacked Area Chart component, all of the Newick trees that exist for the selected clone are retrieved from the Newick Tree database, as well as all of their sequences' information from ImmuneDB, through the second application of the API. This data is then sent to the TreeViews component.

The lineage trees interface component has three blocks: TreeViews, which consists of several instances of the View component, which in turn each contains a Tree component. The TreeViews block is where all the lineage tree visualizations are created and stored, with a name, selected time points, and visualization modes, as described in Section 3.3.6. Each visualization is represented by a View component, which handles which visualization options are selected, such as the shapes attributed to a metadata value or visualizing the number of mutations, and provides these options to the Tree component rendered inside it. Each Tree is responsible for mapping the Newick format tree and rendering the tree visualization according to all the options passed as parameters.

In the TreeViews block, the metadata categories and values are requested from the second API application.

4.3 ImmuneDB Database Structure

The dataset that contains all the data necessary to develop the desired prototype belongs to an ImmuneDB database. ImmuneDB, as mentioned in Section 2.3.5, is a database-backed system to analyze and store large amounts (terabytes) of high-throughput B-cell receptor (BCR) and T-cell receptor (TCR) data. It relies on MySQL DBMS to maintain a database schema that enables the storage of immune receptor sequencing data. The database schema for ImmuneDB can be observed in Figure 4.2.

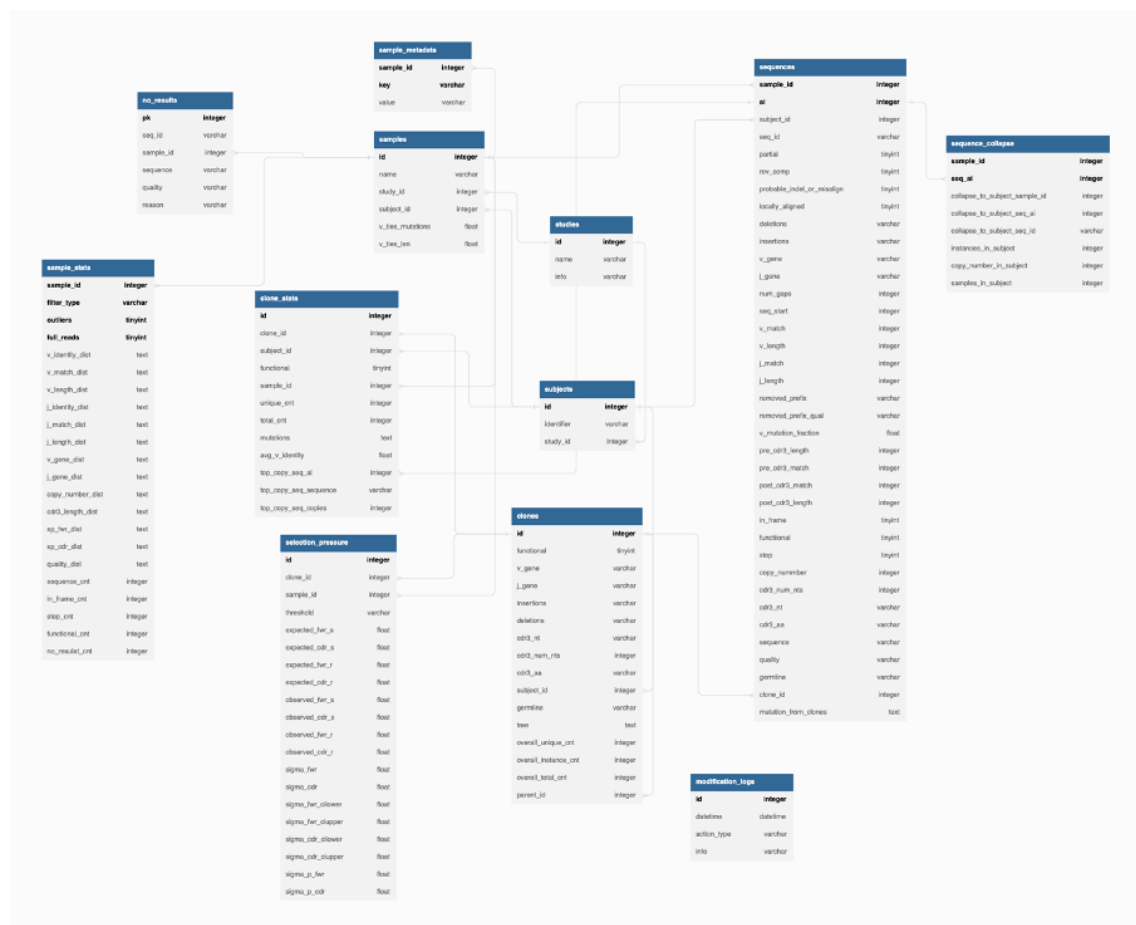


Figure 4.2: ImmuneDB table structure [45]

Since ImmuneDB is a SQL-based tool, it makes it easier to retrieve data by directly querying the dataset. Data obtained from ImmuneDB will be necessary for the three stages of the proposed prototype, based on the hypothesis stated in Section 1.2: the filtering process, the visual time narrative for the clones' evolution overview, and the lineage trees visualization.

The filtering stage would have several filters, either metadata filters or non-metadata filters. All metadata filters will be able to obtain clones based on the sample_metadata table. This table is highly customizable according to the study in question, by entering the desired types of metadata and possible values. For the dataset to be used in this project, the metadata includes parameters such as the time points of sampling, or the tissue where the clones originate from. Non-metadata

filters, such as size-based filters, will need to query tables such as the `clone_stats` table, which stores general information on clones.

The time series overview visualization, which intends to portray the prevalence and number of clones present throughout the considered time, obtains from the database which clones can be observed at each time point and their characteristics, particularly their size/frequency, which can be chosen from different parameters (`total_cnt` from `clone_stats`, for example, is the total number of sequence copies in a clone).

Regarding the lineage trees visualization, given the algorithm with which they are obtained, trees may include branch lengths of zero or estimated lengths/number of mutations. The relationships between clone mutants obtained as output must be crossed with an algorithm for the calculation of mutations, described in detail in Section 4.5.3.3, that compares mutant sequences with germline, retrieved directly from the ImmuneDB dataset, from the sequences table. Other information, necessary to improve the lineage tree analysis provided by this tool, such as sequences collapsed into each mutant sequence or visualizing metadata, is queried directly from the ImmuneDB database.

4.4 Tree Building

Before mapping and rendering trees, lineages need to be constructed from the mutant sequences that belong to each clone. The following sections document the steps needed to obtain the Newick format trees that are parsed, mapped, and rendered into lineage visualizations. These steps include collecting the sequence data in the necessary formats, prepping this data, and inputting it into the IgPhyML tool, configured with the necessary parameters.

4.4.1 Data Preparation

Building trees with the IgPhyML tool requires as input AIRR-formatted sequences that are grouped into clones with computationally inferred germlines (Section 2.3.6).

As shown in Section 2.3.5, ImmuneDB databases provide the possibility of a web interface that allows browsing and visual analysis of the dataset. In order to obtain data compliant with the AIRR format to use as input, data files in TSV format with the desired sequences were downloaded from the frontend interface of the database used in this prototype, `covid_vaccine_new`. Each of the downloaded files contained clonal sequences from a specific sample, subject, and time point and contains all the required columns/fields by IgPhyML.

Once the AIRR files were obtained, it was necessary to incur in the preparation of the data to be used as input for building trees. The Pandas Python library, which allows easy manipulation of very large data files, was employed to accomplish all the actions necessary on the sequence files.

For analysis purposes, and in order to have a tree for each single time point for a clone, the first step consisted in having the acquired files merged in a way that obtains a single file per subject and time point. In addition to these files, all the sequences for all time points for a specific subject

are also merged into a single file. This file is the input used to build lineages that encompass all the time points a clone appears on.

Subsequently, there are many sequences that belong to a single clone, and, therefore, all of them must share the same germline. In order to ensure all sequences from the same clone share the same germline, in each file, the most frequent germline was found for each clone and substituted as the germline for all the relevant sequences. These data files were then ready to be used with IgPhyML, as described in the following section.

4.4.2 IgPhyML and Newick Trees

As demonstrated in Section 2.3.6, the Immcantation's tool IgPhyML [32] is used to build phylogenetic trees for B cell clonal lineages. The IgPhyML software was chosen to build the basis for the B cell lineage trees in this prototype since it simplifies the process of having to fully build a tree-building algorithm. IgPhyML uses clonal clustered sequence data extracted from the ImmuneDB dataset in an AIRR-compliant format that must be treated and analyzed beforehand, as described in detail in Section 4.4.1. There are different forms of using IgPhyML to build trees, from which using the Change-O [29] BuildTrees.py tool and running IgPhyML indirectly with the `-igphyml` option is the most straightforward. BuildTrees.py is run as a command line prompt and, when used as stand-alone software, generates IgPhyML input files from TSV files of clonally clustered sequences. This software also provides options to process the TSV file provided before transforming it into an input file for IgPhyML, such as filtering out non-functional sequences.

IgPhyML was used for this prototype with BuildTrees.py and was run remotely on a server from the University of Haifa, given the computationally demanding tasks from the software. Given that the IgPhyML tree-building process is considerably prolonged when using more than a few thousand sequences, the dataset was divided and the sequences were aggregated with regard to subject and time, as described in the previous section. Having several TSV files allows several processes to be run simultaneously since the installation on the server is containerized.

The basic command used to build trees used for this prototype is:

```
BuildTrees.py -d path_to_sequence_file.tsv --outname tree_output_file --collapse --nmask --minseq 2 --igphyml --optimize tlr --nproc 8.
```

This command receives several parameters that correspond to different options for building the B cell clone lineage trees. The first parameter, `-d`, and the second, `--outname`, refer to the paths of the input TSV file and the name of the output file with the generated trees, respectively. The third parameter indicates that identical sequences are to be collapsed, which is useful for reducing time and computation overheads. The following option, `--nmask`, respects the alignment of sequences. The parameter `--minseq` creates a subsample of the data where clones must fulfill the requirement of, in this situation, having two minimum sequences. The remainder of the options, starting with `--igphyml`, respect the phase when trees are built with IgPhyML. There are several optimization options for IgPhyML, which change the algorithm used to build trees and permit a faster output generation. The core algorithm for the generation of phylogenetic trees in IgPhyML,

as seen in Section 2.3.6, is HLP19. IgPhyML offers several optimization parameter options, including the best-suited option for this study: building using Maximum Likelihood (ML), while still estimating HLP19 parameters, a simpler but reliable approach to obtaining tree topologies. Using the `–optimize tlr` option will use this algorithm for the generation of trees while optimizing their topology and HLP19 model parameters and estimating branch lengths (a value that encodes how mutated the child node is in relation to its parent). Lastly, the parameter `–nproc` specifies the number of threads being used in the process. After running the command for all the files of the dataset, the corresponding output files were generated and downloaded. There are two types of output files generated: files with the fasta extension, which list all the sequence identifiers and corresponding sequences in each lineage, and output files that contain a list of all the generated lineage trees in Newick format (see Section 2.3.7) and its related information. For this prototype, only the last-mentioned files were used, from which the Newick format trees were extracted and stored in the tree database described in Section 4.2. Since IgPhyML requires a minimum of unique sequences, some clones will not have lineage trees for all the time points of the dataset.

4.5 Application

The prototype developed is a subsequential application with a unidirectional flow, where each interface component is reached through interaction and data filtering. According to the described architecture (Section 4.2), these components communicate with the API applications to retrieve the visualized data. The next subsections detail how each of these interface components was implemented, all the interactions that occur between the functional components of the system, and how the data flows through the application.

Figure 4.3 displays the three different components on the final implemented prototype: the filtering form component, the time series clone size evolution visualization and the lineage tree visualization component, with the Window Manager and several Tree Views.

4.5.1 Filtering Stage

Following the concept for the filtering stage presented in Section 3.3.1, the prototype implements an interface component that contemplates a series of filters whose function is to reduce the high volume of data from the entire dataset. As specified in Chapter 3, modern sequencing technologies produce a very large amount of cell data which makes it near impossible for a human to analyze. Furthermore, only a small fraction of the sequences and respective clones are truly relevant in the immune response, and it is important to give them significance visually.

Thus, some essential filters were implemented (Figure 4.4): subject, time point, tissue, and size filters – clone size, and top clones. The subject filter is a mandatory filter that selects one of the study's subjects, whose B cell clones are going to be analyzed. The time points filter allows the selection of which sample collection time stamps are going to be visualized in the next prototype interface components. The time point filter allows for the selection of one or many options, and when more than one time point is selected, the filter result clones will all have to have appeared on

MultiTimeVis

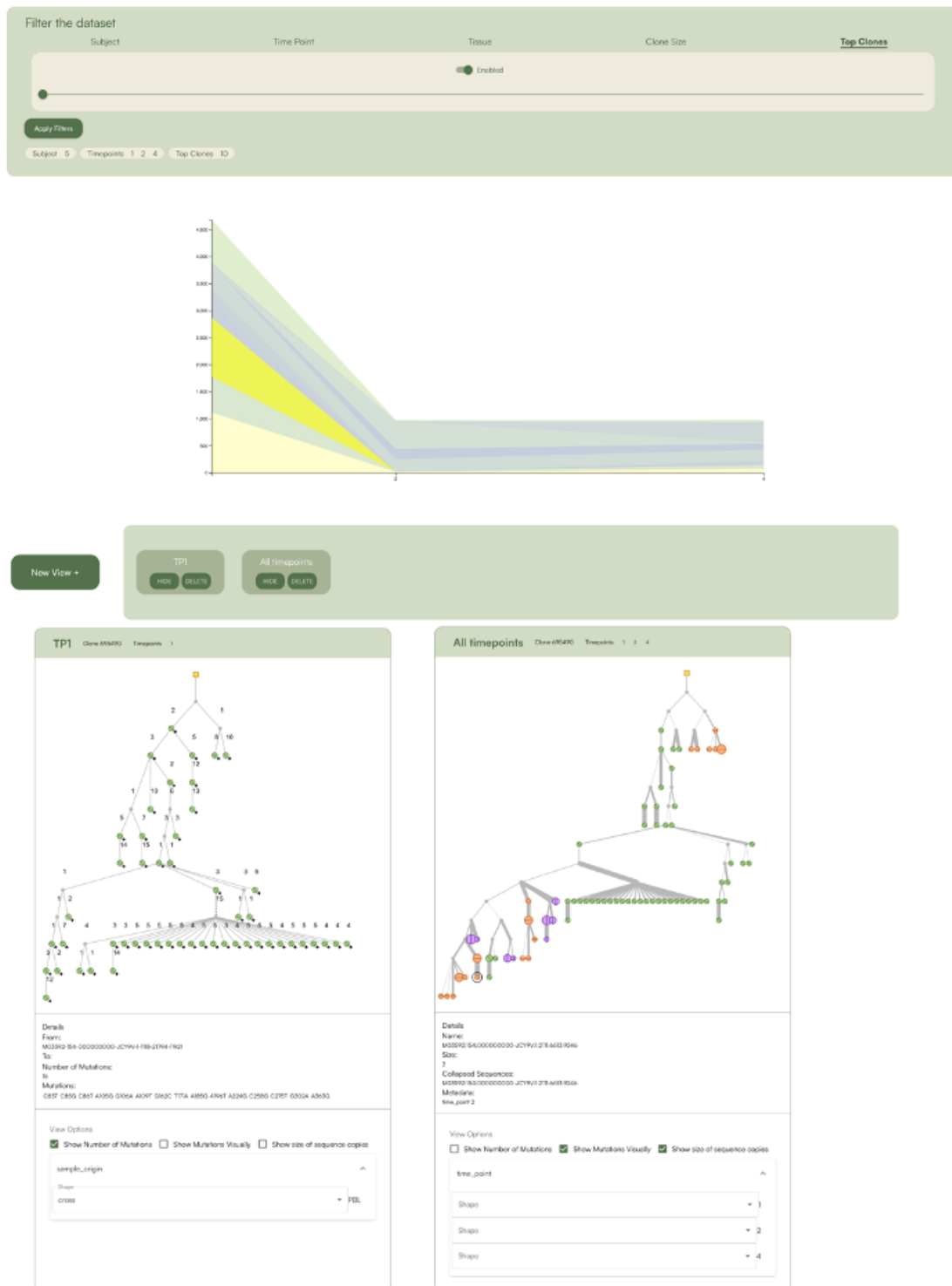


Figure 4.3: Prototype Developed from Proposed Solution

Figure 4.4: Prototype Filtering Form.

all the time points selected. The tissue filter, which represents the organ origin of the sample, also allows the selection of multiple options. However, in the test dataset, all the clones have the same tissue origin and as a result, only one option can be chosen. In the eventuality that the used test dataset had more than one sample origin and multiple selections were made, the clones resulting from this filter would include the sum of the clones that match each of the options. The tissue filter also represents how other possible metadata filters would function. Finally, the clone size and the top clones work to limit the number of clones, namely in order to display the ones more relevant for analysis: larger clones with more clonal expansion. These filters are optional and can be enabled and disabled. The clone size works as a range, where the user can choose a minimum and a maximum clone size, while the top clones are chosen with a numerical value. As mentioned previously in this document, there are several measurements of size, and for these last filters, size is measured according to the total number of copies per clone.

The enabled filters and their assigned values are sent to the API in the form of an HTTP request in order to be used in queries made to the database. There is a dedicated REST API for filters, which enables retrieving the clone IDs that fulfill the filter conditions, as well as their size for each time point, in order to be rendered in the Stacked Area Chart interface component. In the Filter API, the filters are cumulative, i.e., the response result is the intersection of the data that results from the queries of each filter. Only the filters with attributed values are processed in the API. The filter server is prepared to easily integrate more metadata filters since the tables and logic behind the querying are the same for all the filters that fit into this category.

4.5.2 Clone Size Stacked Area Chart

After the submission of the selected filters, as explained in the previous section, results are mapped and rendered in the Stacked Area Chart interface component (Figure 4.5).

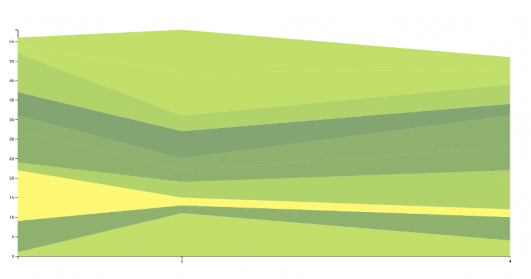


Figure 4.5: Example of a Stacked Area Chart render.

As described in Section 3.3.2, a stacked area chart consists of a visualization of several layered time series that share the same timeline. Each layer in the stacked area chart represents a clone and the evolution of its size value.

The response returned by the API after the submission of the filter form contains a list of all the IDs of clones and their size (total number of copies) at a specific time point where that clone appears. D3.js [3] is the JavaScript library used to map and render this chart, as presented in Section 4.1. In order to build this graph, D3.js receives a data structure that lists all and each clone and creates the correspondence to their size for each moment that it maps into a layer in the graph.

Considering that a clone's size is measured in several samples at each time point (Section 2.3.5), there can be more than one element of the list with the same clone ID and time point. Since there can only be a layer for each clone, and therefore, only one data element for each clone identifier, the data was restructured so that the size, being the total number of copies, of each clone is a sum of all the size entries for each clone-time point pair.

With D3.js, this final data structure is transformed into stacks and rendered as a graph. An example of the rendered chart is represented in Figure 4.5.

Each stack is attributed a color, as presented in Section 3.3.2, that represents its size in relation to the remainder of the clones. The color is assigned according to the larger size of each stack for all the time points and according to its size in terms of what percentage it is of the size of the larger stack. The color yellow is attributed to the larger stack, and the smaller stack has the color blue. All the other stacks will be colored with an interpolated hue between the limit colors, calculated with the aforementioned percentage.

Once rendered, there are two possibilities of interaction: hovering and clicking. When a stack is hovered with the mouse, a tooltip is rendered, presenting the clone identifier corresponding to that specific stack. This tooltip substitutes the need for a legend and aids in selecting the desired clone for the next section's analysis. If a stack is clicked, the clone that it corresponds to is selected to be analyzed in the Lineage Tree interface component (presented in the next section), and the opacity of all the remainder of stacks is reduced to emphasize the clicked layer.

When a stack is selected, the Newick format trees for that clone identifier are requested from the Trees API, as well as information for all the sequence identifiers found in all the trees necessary for the visualization in the Tree Views stage. This information consists of, for each sequence identifier, the sequence, the germline, the size of that sequence (total copy number for the present subject), the sequences that were collapsed into it, and all the metadata associated with the specific mutant. Once all this data is obtained, the prototype renders the stage characterized in the following Section.

4.5.3 Lineage Tree Visualization

The Lineage Tree Visualization interface component corresponds to the last visualization stage of the prototype, which, as stated in the previous Section, is rendered once all the necessary data is retrieved from the databases. In this section, as explained in Section 3.3, lineage trees for multiple

and for single time points can be analyzed and compared. Each part of this component is described in the following sections.

4.5.3.1 Tree Views

The first step in the lineage tree visualization consists of the Tree Views component, which, as described in Section 3.3.6, encompasses the module that creates the layout for the visual analysis and the structure to handle different views of lineage trees.

In order to create meaningful comparisons between lineages in different time points and with different structures, all described in Chapter 3, the user is given the possibility to create different lineage tree visualizations, or Tree Views. Therefore, a window manager was required to manage all the Views. In order to create a new View, the user must fill out a form that presents all the necessary options in order to visualize a lineage: a name, the time points to visualize, the visualization mode (side-by-side or integrated), the tree shape (regular or radial), and which metadata categories should be visualized in the tree (optional). The form can be seen in Figure 4.6.

Figure 4.6: Tree View Creation Form.

When a new View is added, an object with all the selected information, plus a property that indicates that the view is showing or hidden (the default is showing), is pushed into a data structure that keeps all the created Tree Views. All the Views in the Tree Views list are rendered in an interface in the form of a visual list, as shown in Figure 4.7, where each can be hidden/shown and deleted (removed from the aforementioned data structure).



Figure 4.7: Tree Views Interface.

Each View (illustrated in Figure 4.8), with the property showing enabled, renders a draggable window with a header that contains its name, clone identifier, and the time points being visualized, the rendered lineage tree with the details window, and the View Options block. The latter defines a series of visualization parameters passed to the Tree that affect what is mapped and rendered in the lineage visualization.

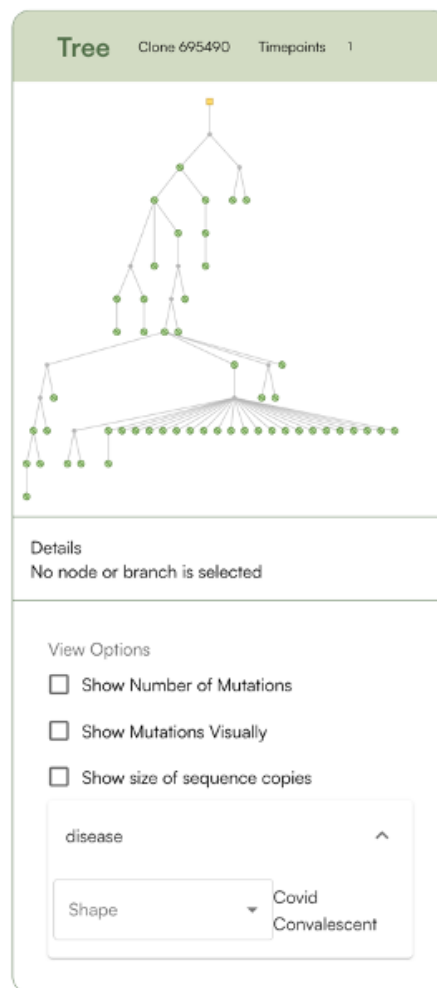


Figure 4.8: Example of a Tree View.

4.5.3.2 Parsing Data and Rendering Trees

When a Tree View is created in the previous segment, it is necessary to map and render the visualization of the desired lineage tree, incorporating the selected visualization options.

The selected time points and the visualization mode (integrated or side-by-side) define the Newick (see Section 2.3.7) tree being parsed for a visualization. When a tree's mode is integrated, the Newick format lineage used is the tree built for all time points, and there will only be a single rendered tree. When side-by-side is selected, there will be a Newick tree for each selected time point, displayed next to each other.

In order to visualize each tree, it is necessary to parse the Newick format into a format that can be used as input for D3js. Parsing a Newick tree consists of analyzing the string in order to retrieve each node element and its relationship (parenthood, siblings, ...) with other nodes and storing them in a nested data structure. The trees obtained with IgPhyML contain inferred nodes (Section 2.3.6) that sometimes have very small distance values to their parent node. This occurs since the ML model assumes, correctly, that only partial mutation information is present. Moreover, there could also be reversion events (i.e. reverse mutations). Thus, the lineage model could suggest branches whose length (i.e. number of mutations) does not exactly match the number of positions that differ between adjacent sequences / cells in the lineage. In order to obtain a more accurate tree, the parsed tree was traversed recursively, and all inferred nodes found with a distance smaller than the defined threshold (0.001) were considered zero-distance nodes, which are then considered equivalent to its parent in terms of mutations suffered. Zero-distance nodes are, thus, removed from trees, and their sequence is collapsed into the parent sequence. The tree structure is readjusted, by having their children become their parent's children together with the node's original siblings.

This structure is next transformed again with the D3js library, which adds information and creates the hierarchical input for the tree visualization. D3js is responsible for mapping and rendering the tree as an SVG, as described in Section 3.3.3, with nodes, color and lines that represent time points. According to the option selected in the form for creating Views, the nodes of the tree can be mapped according to a regular or a radial layout, the latter illustrated in Figure 4.9.

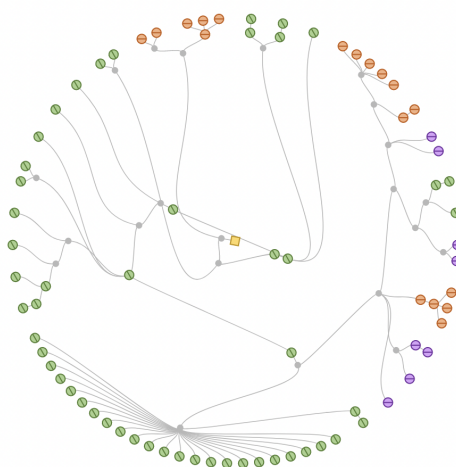


Figure 4.9: Tree radial layout.

View options alter the rendering parameters and change the final result, as illustrated in the example of Figure 4.10. In order to calculate a diameter of a node for visualizing sequence size, a set of size ranges is defined, and each is associated with a numeric value. If this option is selected, the diameter of each node will correspond to the value of the range where its size belongs. If the mutation visualization options are selected, each edge changes by either displaying the number of mutations next to it or by changing the edge's stroke thickness to the number of mutations. When metadata categories are selected, the shapes associated with each attribute will be rendered to the right of all the nodes that possess that specific metadata value. The tree is re-rendered every time any of these parameters changes.

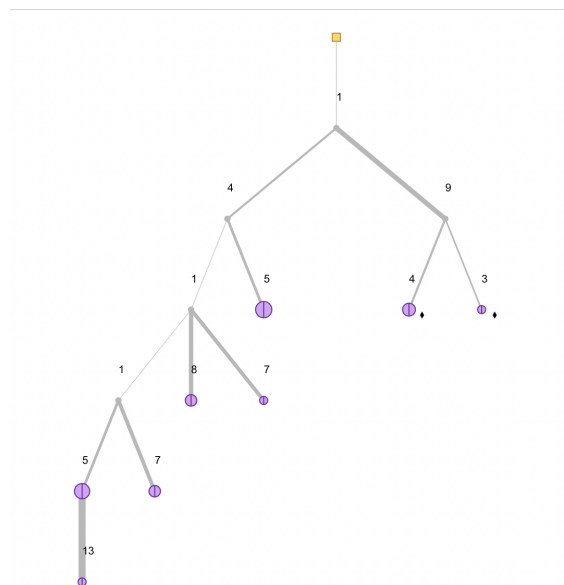


Figure 4.10: Tree visualization with the different view option parameters toggled.

4.5.3.3 Calculation of Tree Mutations

An algorithm for calculating the mutations that occur on all the edges of a tree was developed to understand which sequence changes occur between each pair of consecutive nodes. Knowing these differences is necessary to display visually the number of mutations in each edge and list all the mutations as an edge detail, as well as ensuring a correct analysis of the tree's structure.

This algorithm is composed of two phases:

- The first obtains the mutations for each node in relation to the germline. This step is a part of the Newick tree parsing described in the previous section, where for each sequence identifier, the sequence string is retrieved and compared character by character to the germline string of that clone. A character in a sequence string corresponds to a nucleotide base and can have the values A, C, T, or G (see Section 2.3.1). Each character from the mutant sequence that differs from the germline is considered a mutation, which is stored in the format “germline character – index in string – sequence character”, for example, “A3T”. All the

mutations found per node when comparing the two strings are stored in a list, which is, in turn, stored in the hierarchical nested data structure.

- The second phase of the algorithm, which happens once all the nodes' germline-related mutations are determined, is obtaining the specific mutations for each edge. The mutations in each edge are composed of the mutations that occur in a node but don't occur in its parent. In order to calculate these differences, the tree is traversed using the Reverse Level Order Traversal algorithm, which traverses the tree in the same manner as the Level Order Traversal [47] algorithm but begins with traversing the leaf nodes instead of the root node. This stage of the algorithm is also used to identify which mutations appear on an inferred node. These mutations are found during the traversal by comparing the mutations of the children of the inferred node and finding the common ones. The list of common mutations is attributed to the inferred parent and compared to its parent once it is traversed, as are all of the nodes in the tree. If a node has exactly the same list of mutations as its parent when compared, the edge to that node will have no specific mutations. When that is the case, the node is deleted, and, as with zero-distance nodes during parsing, its children become its parent's children. In addition to this, if the node with no mutations has a known sequence, the sequence's identifier is then considered a collapsed sequence of the parent node.

4.6 Summary

This chapter documents and details the functional prototype (Single page Web application) that instances the solution conceptualized in Chapter 3. As detailed in Section 4.2, this application is, from an architecture point of view, divided into three layers.

Two databases compose the first layer, and, referencing Section 4.1, both the Newick database and the ImmuneDB dataset are based on MySQL. The database layer communicates with the API layer, which has two applications built using Django REST Framework. The final layer is composed of the client application, built with ReactJs and D3.js, that requests data from the API in order to render the visualizations.

In Section 4.3 there is an overview of the data schema used by ImmuneDB and how data is transferred from the database to the visualization pipeline presented in Section 3.2. Required for the last stage of visualization, Newick format lineage trees are built with IgPhyML from AIRR-formatted data that is treated and prepared before being used as input for the software, as seen in Section 4.4.

Section 4.5 focuses on the procedures of the programming development of each of the segments of the web application. The filtering stage is composed of five filters: subject, time points, tissue, clone size, and top clones. The filter parameters selected by the user are sent to the API in order to retrieve the identifiers and size of the clones that correspond to the desired features. Each obtained clone is subsequently mapped into a layer of the stacked area chart, with the sizes at each time point creating the area stack. Each stack, when hovered, displays a tooltip with the clone

identifier. When a layer is clicked and, thus, a clone is selected, the Tree View is rendered. In this interface component, several Views can be created and stored with a form that requests a name and the visualization parameters for a lineage tree. Stored Views can be hidden/shown or deleted. The Tree View component functions as a window manager, where each View can be dragged and rearranged for better comparison of the different visualizations. In each window, several visualization options that change the lineage visualization, such as changing the node's diameter according to their sequence size, can be selected. The lineage tree visualization render depends on the parsing of the Newick format lineages into a nested hierarchical structure that D3js can use. Furthermore, a two-phase algorithm was developed to calculate the mutations in each edge to visualize and display them. This algorithm also analyzed and rearranged the final tree structure by removing nodes that aren't mutated when compared to their parent. This rearrangement ensures a consistent lineage tree analysis.

Chapter 5

Evaluation

This chapter describes how the evaluation of the functional prototype was conducted. Section 5.1 presents the main objectives of this evaluation artifact and, ultimately, the desired features for the developed tool. Section 5.2 focuses on what was the adopted methodology for this process. The test script itself, provided to the participants of this experiment, is detailed in Section 5.3.

5.1 Objectives

The conducted experiment aimed to assess the usability and effectiveness of a human-computer interface process composed of filtering tasks and visual analysis tasks towards the analysis of B-cell evolution lineage trees. The functional prototype was developed (Chapter 4), and this experiment intends to understand if it fulfills its purpose both from a functional and an analytical point of view.

Specifically, the objective is to assess the following features:

- The user's ability to go through the necessary process of filtering data to reach a lineage tree visualization of and interact with this view to obtain useful information about it for the immunology analysis.
- The capability of visually extracting meaningful information from analytic features on the evolution of the B-cell across time through the lineage tree visualization.

5.2 Evaluation Methodology

Both features detailed in the previous Section were evaluated by requesting users (participants) to perform different filter tasks on a well-known dataset and several visual analysis tasks over the lineage tree visual representation.

This experiment used formative assessment for usability problems in the form of questionnaires for each participant. Both objective and subjective data were collected throughout the

evaluation process in the form of answers to tasks, post-task, post-test questionnaires (see Section 2.2.2), and open perception questions, whose goal is to allow the participant to explain in free form their perception of the tool.

This experiment needs to be conducted with participants with adequate knowledge in immunology due to the necessity of assessing the tool's capability of conveying valuable information related to the field. In order to have a sample large enough to infer meaningful conclusions but considering the necessity of specialized knowledge for this test, the considered minimum number of participants was 10. Demographics should be well balanced, especially gender.

Considering the need for participants with experience in the immunology field and, therefore, the need to have testing processes with participants located in different countries happening in the same time period, the experiment was held remotely and was unmoderated. Thus, Google Forms was the tool selected to orchestrate the script and provide forms to the participants. Furthermore, the participants had access to the functional prototype through the browser in order to perform the tasks.

Prior to the realization of the experiment itself, a pilot test was run. Pilot tests help to rule out or correct any questions or tasks that may be ambiguous or redundant, or not clear enough for a participant to perform, especially due to the fact that the experiment was held remotely and unmoderated. A pilot test was conducted with a person that was familiar with immunology and with the prototype.

5.3 Testing protocol

As mentioned in Section 5.2, each participant was given a script with a questionnaire in order to evaluate the developed prototype. The testing script consists of the following 6 sections: Informed consent form, Pre-questionnaire form, Pre-test prototype presentation, Tasks questionnaire form, which includes Post-task questionnaires, Post-questionnaire form, and Open questions. The structure and contents of the testing script are detailed in the next Subsections. The Google Forms script can be seen in Appendix A.

5.4 Informed Consent Form

Prior to the beginning of the experiment, an initial form is necessary in order to obtain consent from each participant to the collection and use of personal data and data from the experiment. Therefore, an informed consent form section was presented, through Google Forms, for the person executing the usability test to read about the experiment and goals. In order to give their consent, a brief description of the experiment, along with what personal data was retrieved during the usability testing process, is presented. Without agreeing with the informed consent form, the participant did not go further in their own test. Although the test collects data on personal information, the answers, and any other provided data are given anonymously.

All the steps in the experiment are compulsory for the participation of the user to be considered.

The contacts of the person responsible for the experiment, Leonor Freitas, were also provided to the participants.

5.5 Pre-Questionnaire Form

Prior to the experiment itself, it is necessary to collect the relevant data on the participant for the results' analysis. The personal data gathered consisted of the participant's country of origin, gender, as well as their level of knowledge regarding both immunology or phylogenetic tree analysis and information visualization tools, using a 5-degree Likert scale. This data was kept for the duration of the experiment plus the time necessary to perform the relevant analysis.

5.6 Pre-test Prototype Presentation

Before starting the experiment, a brief introduction of the main blocks of the application is presented so that the participants can familiarize themselves with the prototype interface blocks. The application has three main blocks: the filters, the stacked area chart visualization (on clone size over time), and the tree visualizations (clone lineage trees). For each of these blocks, the presentation contains a short text describing it, as well as a few illustrative figures.

5.7 Questionnaire Form

For the concrete experiment, each participant follows a list of tasks to perform while using the functional prototype, which is accompanied by a questionnaire relating to the tasks as they were performed. This questionnaire served as a way of both confirming the success/failure of the task and as a way of collecting information on smaller sections of the prototype.

Given the previously defined goals, the tasks are divided into two main groups:

- Functional tasks are divided into three subgroups according to the three main blocks of the prototype interface (filter area, stacked area chart visualization, and tree visualization).
- Tasks that relate to immunology visual analysis relying on tree visualization, where the information on the evolution of the B-cells is more meaningful.

The selected tasks are performed over a well-known dataset (covid_vaccine, Section 2.3.5.1), for which the outcomes of the filter and the conclusions from the visual analysis are known a priori by the experiment evaluator but not by the participants. Therefore, the expected result could be recognized, and a success criterion could be established. The majority of the lineage tree visualization tasks presented are performed on the trees of clone 695490, a large and heavily mutated clone.

5.7.1 Functional Tasks

Functional tasks consist of simple tasks performed on each of the interface blocks. These tasks were used to confirm that the participant gets acquainted and understand the structure of the application, as well as to demonstrate the usability of each block. These tasks have a binary success criterion (either successful or not) that is used to measure the error rate.

A series of functional user goals are determined to better define which tasks should be performed with the application, relating to the interaction with the rendered graphics in order to perform filter tasks. Examples of these goals are the ability to select filters and submit them to obtain the stacked area graph or to select a node from a lineage tree visualization in order to visualize its details. The functional tasks section is divided according to the three stages of the prototype. Consequently, the first section requests four filtering tasks (F1 to F4) from the participant, each corresponding to the desired selections on each filter (subject 5, time points 1, 2, 4, and top 10 clones) and their submission. Following these tasks, three questions are asked on the performed tasks. The first two questions have the objective of confirming the success of these tasks. The success of the tasks is based on the characteristics of the graph obtained from the data that corresponds to the selected filters. Thus, the participant is asked to present the number of stacks visible on the screen and to select the correct statements from a series about the visualization's characteristics, such as variation in size. The third question's aim is to understand the usability of the section obtained after submitting the filters by requesting the user's opinion on which information is successfully transmitted with the Stacked Area chart visualization. The Stacked Area Graph section gets a single task (SAC1) from the participant and a corresponding question to confirm its success: it is requested that the participant interacts with the rendered visualization by hovering in each stack to reveal the matching clone identifier and, consecutively, select the desired clone (number 695490). In order to confirm that the interaction is as intended, the participant must select the correct option for what appears on the interface.

Lastly, the Lineage Tree Visualization Tasks combines the description of the task with its confirmation question. There are six tasks in this section, from LTI1 to LTI6. The first three tasks require the participant to create three new lineage tree views, and the ensuing question for each is a simple success verification based on either the number of trees in each view or the color of the nodes. Tasks LTI4 to LTI6 use the views created in the previous tasks to test the interaction usability and the user's ability to obtain detailed information on the tree's nodes and edges.

5.7.2 Immunology Analysis Tasks

Visual analysis tasks are implemented to assess the extent to which the proposed visualization provides insightful information to a user versed in the Immunology field. Besides the collection of objective data such as completion success, the questions related to these tasks may be more open-ended to better understand the analysis made using the tool. These tasks focused on the tree visualization interface block since most of the visual analysis is to be done with its views. The questions in this section intends to demonstrate if the tool is effective in providing support to the

analysis of selection processes and the evolution of B-cells by obtaining meaningful, although possibly distinct, responses about the comparisons established. Since these questions have more complexity than the functional tasks questions, only three questions are posed to the participants. The first two questions (VA1 and VA2) were based on a task that requests the participant to obtain the lineage trees for all possible time points (1, 2, 4, and all) for clone 695490. Question VA1 asks the participant to consider the lineage tree visualizations for the single time points 1, 2, and 4 and to evaluate whether there is consistency when predicting past selection processes when looking at a lineage from a later point in time. Question VA2 requests the participant to compare between the structures of the single time point trees and the tree that includes nodes from all time points and to clarify what happens to the relationships of nodes from each time point when considering them in the tree that includes all. The last question, VA3, is the same as question VA2 but has different lineages to be compared and intends to be a quicker analysis. Therefore, instead of being required to perform a task to obtain visualizations, the participant is provided with an image of the lineage tree for time point 4 of clone 683843 and an image for the lineage of all the time points for the same clone.

5.7.3 Post-Task Group Questionnaires

Since the questionnaire pertains to the different sections of the prototype and encompasses a large number of tasks and questions, it is logical to present post-task group questionnaires instead of post-task questionnaires to the participant. The reason for this is that although it is necessary to perform questionnaires throughout the experiment so that the participant's memory is fresh and they can give more specific feedback on the recently performed activities, this way, it would not overload and tire the participant performing the test as it could happen with questions on every single task. A single answer is requested, as well to confirm the completion of each task, and this answer corresponds to the objective of the activity in question.

Considering the simplicity of the tasks for the filtering and the stacked area chart tasks, a NASA TLX questionnaire [31] is requested to be filled by each participant after the lineage tree visualization section tasks to assess the subjective task load of the activities proposed to the participant.

The post-task questionnaire helps collect both quantitative and qualitative data about the tasks the participant is asked to perform.

5.7.4 Post-Questionnaire Form

A final SUS questionnaire (see Section 2.2.2), is presented to each participants after all the tasks and the post-task group questionnaires are completed, in order to understand how the participants perceived the usability of the app as a whole.

5.7.5 User Observation Open Questions

Taking into account the difficulty already referred to conducting one-on-one interviews with the participants post-experiment, a section with open-ended questions is provided, to further understand the way a participant uses the prototype and perceives its usability. Two long-form questions are necessary to further understand if participants recognized the main objective and real usability of the prototype as a new tool and how the application could be improved in order to better fulfill its purpose. This can be central to, in the future, adapting the solution and prototype if the perceived rationale has a valid alternative to the one currently used in the prototype. The first question is about what future features the participant thought should be included in the prototype. The participant is also asked to specify in which part these new features should be implemented and to provide a short explanation as to why these suggestions would improve the usability of the prototype. The second question, regarding the aforementioned reasons, asks the participant to comment if this solution helps improve the analysis of the evolution of B-cell lineages and to provide justifications for their rationale.

5.8 Summary

This Chapter documents the developed evaluation questionnaire for the implemented prototype as well as the chosen methodology employed to create it.

In Section 5.1 the objectives of this evaluation are described: determining the user's ability to go through the necessary process of filtering data to reach a lineage tree visualization and interact with it to obtain useful immunology information about it for the analysis, and determining the participants' capability of visually extracting meaningful information from analytic features on the evolution of the B-cell across time through the lineage tree visualization.

Section 5.2 details the methods and process of the evaluation. The participants were asked to perform a series of tasks on a well-known subset of data and answer questionnaires where both objective and subjective data was collected. It was necessary to perform this questionnaire with at least 10 participants, some of which with prior knowledge of immunology. The experiment was held remotely and unmonitored, through a Google Forms questionnaire and a browser access to the prototype. Pilot tests were also conducted before the launch of the experiment.

In Section 5.3 the structure of the questionnaire form is introduced, which had six sections: an informed consent form, a pre-questionnaire form, Pre-test prototype presentation, Tasks questionnaire form, which includes Post-task questionnaires, Post-questionnaire form, and Open questions.

The Informed Consent Form, documented in Section 5.4, was presented at the beginning of the form in order to obtain consent from the participants to collect and use personal and questionnaire data.

The next segment of the questionnaire, the Pre-Questionnaire form, was presented in Section 5.5. This questionnaire was used to collect relevant personal data from the participants. The

personal data collected was the country of origin, gender, proficiency in digital visual analysis tools and proficiency in the field of Immunology.

Section 5.6 focuses on the Pre-Test Prototype presentation, which consists of a brief presentation of all the components of the prototype interface.

The questionnaire itself, with the tasks and questions required from the participants, is described throughout Section 5.7. The questionnaire contains five components. The first, functional tasks, respects the simple tasks and associated questions that intend to assess the usability of each interface block and if the users understand the structure of the prototype. The second contemplated questionnaire component was the immunology analysis tasks. These tasks are focused on the lineage tree visualization interface block and aim to extract the relevancy of the tool for analysis of the evolution of B-cell lineage trees. Following this component, the Post-task group questionnaire (NASA-TLX) being used after the functional tasks for the lineage tree visualization block, and the Post-Test Questionnaire (SUS) are detailed. Finally, the last component of the form, which consists of open questions for user observations are presented, and these present an alternative to an interview by allowing the participants to express suggestions and opinions on the usability of the tool.

Chapter 6

Results and Discussion

This chapter describes the results and answers obtained from the questionnaire and the raw answers presented in Appendix A. This chapter provides an analysis according to the different sections of the questionnaire, having, therefore, seven sections and a summary.

6.1 Overall Participants' Stats

The first block of the usability test form asked participants to provide relevant personal information to this study: gender, country, proficiency with visual analysis tools, and proficiency with the immunology field. From the obtained answers, it was possible to determine that 53.8% of participants were female and 46.2% were male. Participants were distributed between the United States of America, Israel, Portugal, and the United Kingdom.

There were 61.5% of participants who considered their proficiency in digital analysis visual tools 3 or larger on a Likert scale from 1 to 5, and 38.5% considered it to be 4 or 5. Regarding proficiency in the immunology field and lineage tree analysis, 61.5% of the participants identified themselves as a 3 or larger on a scale from 1 to 5, while 30.8% placed themselves on a 4 or 5 level.

6.2 Filtering Stage

In order to assess the perceived usability of the filtering block, three questions were asked in relation to the obtained results from the presented F1 to F4 tasks. Their results are presented in Section 6.2.1 and discussed in Section 6.2.2. The filtering tasks performed on the filter form were:

F1 - Select subject 5.

F2 - Select clones from timepoints 1, 2 and 4.

F3 - Select top 10 clones.

F4 - Combine all these filters and obtain the stacked area chart.

The questions asked to evaluate the success of these tasks were performed on the obtained clone size time series chart (stacked area graph), to ensure it matched with the expected outcome. The three questions on this questionnaire section were:

1. Select the correct statements about the obtained stacked area chart.
 - The total sum of the size of the chart increases exponentially from time point 1 to 4.
 - Clone 691147 increases its size significantly from time point 1 to time point 2.
 - There is a clone that keeps a constant size throughout time (throughout the evolution of the chart).
 - In this chart, yellow represents larger stacks and blue represents smaller stacks.
 - None of the above is correct.
2. Count the number of stacks you see in the obtained stacked chart and provide it below.
3. In your opinion, the stacked area chart:
 - Helps understand the evolution of the clones' presence over time.
 - Gives a good overview of the size of each clone, in order to choose one for analysis.
 - None of the above.

6.2.1 Results

Select the correct statements about the obtained stacked area chart.

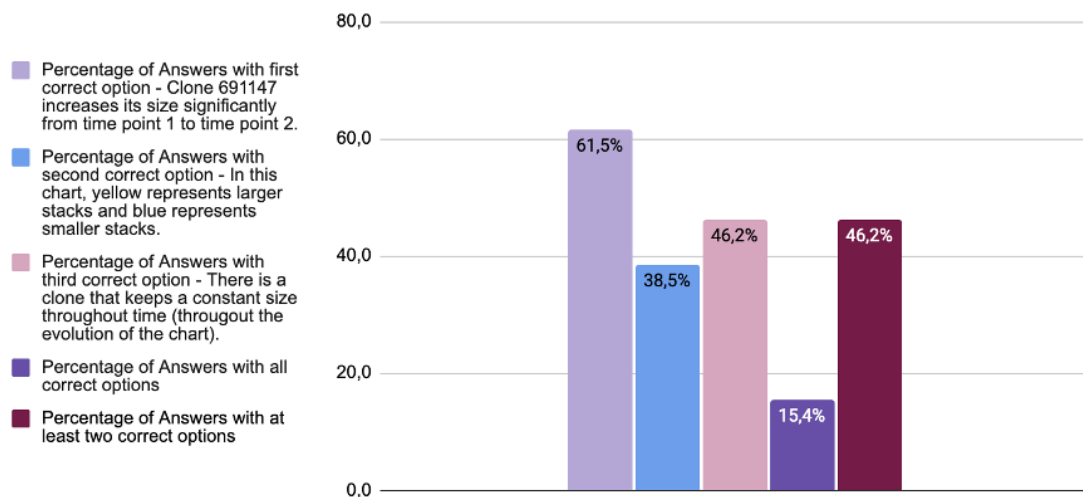


Figure 6.1: Chart of the results of the first question of the Filtering Stage Section.

The first question was a multiple-selection question, providing three correct and two incorrect options, one of which was that no option was correct. This question intended to assess the interpretation of the obtained graph after completing tasks F1 to F4. As seen in the graph of Figure 6.1, most participants selected the first correct option, and a decrease in selection was observed in the second and third options. A single participant selected the option that none was correct, and

no one selected the incorrect option. From all the registered answers, 15.4% of the users selected all the correct options, and 46.2% selected at least two correct options.

The second question required a short format answer on the number of obtained graph stacks in order to confirm the success of the previously completed tasks. The expected input was ten stacks. Of all the participants, 38.5% identified the correct number of stacks, while 46.2% of the answers were 9. One participant identified 8 stacks and another 5 stacks. One of the participants that identified the ten stacks on the chart stated that they were only able to do so by slowly hovering the mouse over the smaller stack, identifying 9 stacks visually.

The last question on the aforementioned tasks was of subjective nature in order to understand the overall perception of the usability of the time series visualization. This question was presented with two checkbox options plus an extra to select none of the other possible ones. The results documented in the chart of Figure 6.2 show that a majority of the participants consider that the clone evolution stacked area chart provides a meaningful overall representation of the clones' size evolution throughout time. 15.4% of the participants considered that the time series visualization did not fulfill any of the provided options.

In your opinion, the stacked area chart:

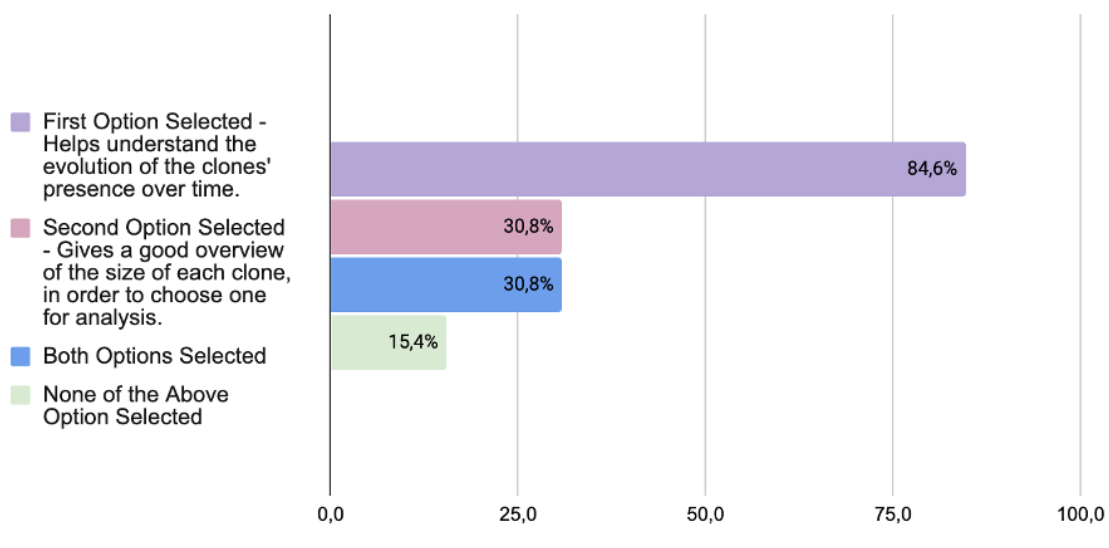


Figure 6.2: Results of the third question of the filtering stage questionnaire.

6.2.2 Discussion

Given the results of the first question and the low success rate, a generalized difficulty in fully analyzing the stacked area chart can be deduced. Although most participants seemed to be able to identify the first correct option, only less than half of the user pool was able to identify the third correct option, and a small minority the second correct option. As seen in Chapter 2, stacked area charts are complex graphs that are more difficult to interpret and require more previous expertise in visual analysis. The vast majority of participants that selected this option had previous expertise in

visual immunological analysis, which correlates to the aforementioned characteristics of the graph. The low selection of the second option can also be attributed to the fact that only an inferred visual assessment is being made without resorting to concrete numerical values, which drives to a much more subjective perspective. The fact that only less than half of the users were able to establish an association between stack size and a certain color (yellow for stacks with the larger maximum size values and blue for stacks with smaller maximum size values) could be happening due to the variation of the size of each stack. Consequently, there is not a single numerical value to attribute directly to a color, and it is not as easy to establish a one-to-one connection. In order to aid in the interpretation of this and other visual variables, particularly since people are not familiar with this type of visualization, it may be necessary to provide additional textual information, such as a legend. Nonetheless, since most users selected at least one expected option, it is possible to conclude that the desired graph was being displayed.

Regarding question 2 of this section, there was a significant number of participants that were not correct in their answer (10 stacks) only by not being able to identify a smaller stack. This characteristic of the obtained results can be explained by the fact that, in the specific test scenario, the obtained graph has a visual emphasis on the portion between the first two time points, and the stacks were therefore being counted from it. In this section, one of the stacks is barely visible and increases as time (x-axis) increases. Taking this into consideration, it can still be acknowledged that the users were correctly identifying the stacks and obtained the intended visualization from the selected filters. The last question of this questionnaire section was more directed towards a subjective interpretation of the information conveyed by this interface component. The majority of the users considered that the graph was helpful in providing an overview of the clones' size throughout time. This matches the notion that stacked area charts seem to be more useful when providing overviews and establishing comparisons between the stacks than individually analyzing each stack. There were still some participants that also considered that the visualization aids in determining the size and evolution of each clone in order to make the necessary selection for further analysis. Since this is a necessary characteristic for this prototype component, this perception could possibly be improved if more information was added, for example, when interacting with the chart, and the specific sequence count number was displayed for each stack per time point.

6.3 Stacked Area Chart (Clone Size Evolution Time Series)

In order to test the usability specifically of the interaction options of the time series stacked chart, a single task, SAC1, was required of the participants. A follow-up question was presented to confirm the success of the mentioned task, whose results and corresponding discussion are detailed in this Section.

The SAC1 task was:

Hover over the stacked area chart for the corresponding clone id to appear over each stack. Click on the stack of the clone 695490. (Please note that after clicking on a stack, it may take some minutes to fetch all the information needed for the next stage).

The follow-up question asked about this task was:

After you clicked on clone 695490 in the stacked area chart, a waiting message appears. Once this message is hidden:

- A button to add a new visualization appears, and when clicked a form is presented.
- A form to create a new tree view appears immediately.
- I obtained nothing after selecting the stack.

6.3.1 Results

In the stacked area chart tasks questionnaire section, the only question was a multiple-choice question where the user could select one of three possibilities. The first and correct option, “A button to add a new visualization appears, and when clicked a form is presented.”, was chosen by 84.6% of the participants. One participant selected the option that stated that a form appeared immediately, and another selected the option that indicated that nothing happened after the task.

6.3.2 Discussion

The majority of participants were able to obtain the correct option, and therefore, it can be concluded that the interaction with the clone evolution stacked graph was overall successful.

6.4 Lineage Tree Visualization

The lineage tree visualization tasks section was developed to test the ability of the users to use the window manager and create new tree view windows, as well as to test the usability of the lineage tree interface for analysis. The results obtained from the tasks in this questionnaire section and their corresponding questions are presented in Section 6.4.1. In Section 6.4.2, the results of the post-task group questionnaire are documented. Finally, in Section 6.4.3, the discussion of the results shown in the previous sections is detailed. The tasks and respective questions asked in this questionnaire section were:

- LTI1 - Create a view for the current clone (chosen in stacked area chart) with the name “Integrated TP 1 & 2”, for time point 1 and 2, in integrated mode and with a regular tree layout. Select also the option to visualize age metadata. When the tree view appears, select the correct option:
 - There are two trees.
 - Some of the nodes are green.
 - I obtained nothing after creating the view.

- LTI2 - Create a view for the current clone (chosen in stacked area chart) with the name “TP 1”, for time point 1, in integrated mode and with a radial tree layout. When the tree view appears, select the correct option:
 - There are only purple nodes on this tree.
 - There are green nodes and greyed out nodes on this tree.
 - I obtained nothing after creating the view.
 - There are only green nodes on this tree.
- LTI3 - Create a view for the current clone (chosen in stacked area chart) with the name “TP 4”, for time point 4, in side-by-side mode and with a regular tree layout. When the tree view appears, select the correct option:
 - The view has two trees side by side, one with different colored nodes and another with only one color nodes.
 - There is a single tree with all the colored nodes purple.
 - I obtained nothing after creating the view.
- LTI4 - Get the collapsed sequences for the node whose name ends in 8243 for the tree view "TP 4". Please write the last set of digits on the name (4 or 5 digits).
- LTI5 - Get the target node name for all branches with more than seven mutations on the view "TP4". Please write the last set of digits on the name (4 or 5 digits). In the case there is a node that does not have a name, add "black node" to the list.
- LTI6 - Find the name of the node(s) with the largest number of sequence copies for the tree for time point 4 clone number 695490. Please write the last set of digits on the name (4 or 5 digits).

6.4.1 Task Results

Tasks LTI1 to LTI3 were created to assess the success rate of the creation of new tree view windows using the window manager. In order to measure this success rate, participants were provided with a multiple-choice question about the obtained result for each task, where out of three options for LTI1 and LTI3, and four options for LTI2, only one option was correct, and one was for not obtaining any results. The measured rate is presented in Figure 6.3 for each of the task-question pairs. For LTI1 and LTI3, the participants that did not select the correct option selected the one for not obtaining any visual result from the task. In LTI2, a small percentage of users, 15.4%, considered there were only green nodes instead of both grayed-out and green nodes.

The second portion of tasks in this section, LTI4 to LTI6, intended to assess the participants' ability to utilize the generated tree visualizations for analysis. The questions associated with each task prompted the participant to provide information extracted from the obtained lineage trees according to the task's instructions.

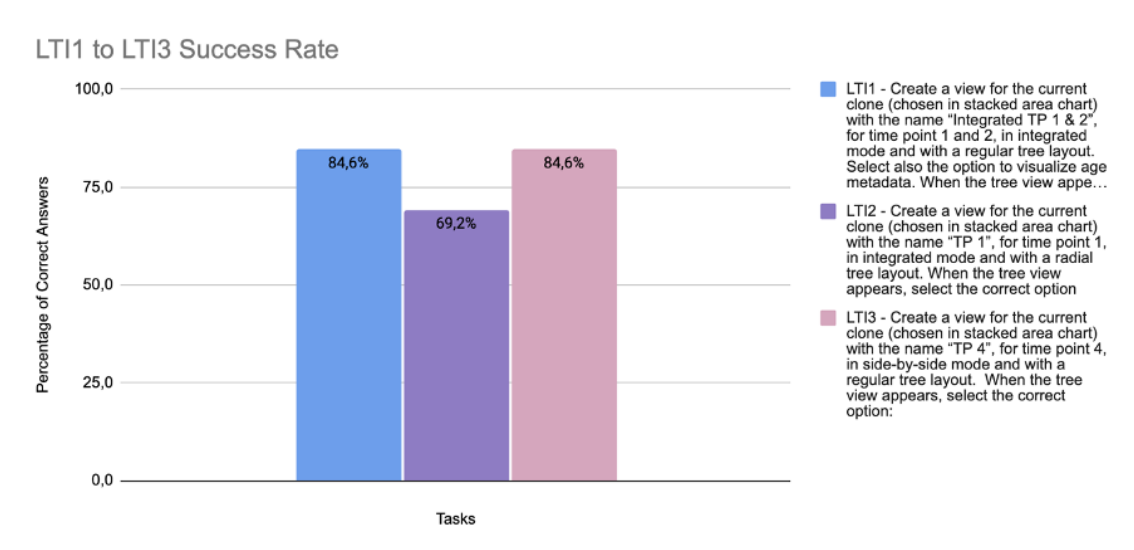


Figure 6.3: Success Rate Chart of Tasks LTI1 to LTI3.

For LTI4, the expected answers were 8243, 11814 and 3170. In the obtained answers, 23.1% of the participants provided all the correct sequences, and 69.2% provided at least one correct answer, either the first or the last listed in the details block of the tree view window.

Regarding the task LTI5, it was expected for the participants to provide the following last digits of sequence identifiers: 19406, 8243, and "black node". The percentage of fully correct answers was 30.8%. Some users, 15.4%, also provided in the answer the target nodes of edges with exactly seven mutations: 18472 and 19088. There were 53.8% of participants who provided the expected identifiers but did not include "black node". Two participants provided empty answers.

In task LTI6, the correct requested last digits of the node identifier were 11740, which comprised 61.5% of the supplied answers. For 7.7% of the participants, this was one of the provided answers based only on visual analysis with no resource to the actual numeric value found in the details block of the tree view window.

6.4.2 NASA-TLX Post-task Questionnaire Results

Following the tasks and questions regarding the lineage tree visualization, including the window manager and the tree view windows, a NASA-TLX questionnaire (ref) was required to evaluate the perceived effort to perform all of the tasks LTI1 to LTI6. The average answer for each question was calculated and is displayed in Figure 6.4. A chart (boxplot) that displays the minimum and maximum values, the first and third quartile and median can be seen in Figure 6.5.

The results are similar when considering all the participants and just the participants with more proficiency in immunology.

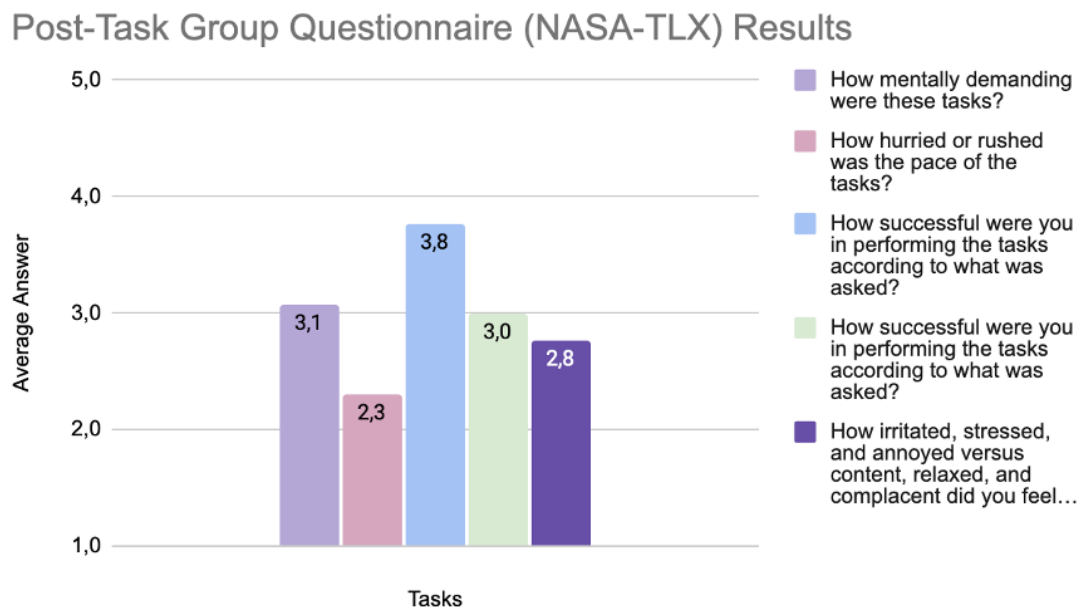


Figure 6.4: Average results chart of NASA-TLX Questionnaire.

6.4.3 Discussion

Regarding questions LT1 to LT3, the high rate of correct answers indicates that participants were able to correctly create the new tree view windows and corresponding visualizations and therefore use the interface's window manager. The identified failures in these tasks were exclusively connected to some users having technical difficulties and not being able to retrieve the data necessary to produce the desired visualizations, which applies to all the following questions.

With question LTI4, although the success rate regarding having all the correct answers was not very high, the remainder of the obtained answers indicate that people were selecting the first or the last presented collapsed sequences for the desired node instead of all. This can be interpreted as that people could still obtain the correct node and locate the information to a certain extent, which fulfills a good part of the objective of the task.

Although only roughly 30% of the answers provided on the LT5 question contained all of the nodes that fulfilled the required criteria, the vast majority of the participants were able to provide the names of the nodes with concrete identifiers, some of them foregoing only the inclusion of the inferred node. This indicates that most of the users were able to perform the intended analysis given the task's requirements.

For the LTI6 task, the expected rationale was for participants to have first obtained the visual option for sequence size and then, from the visually larger, obtain the numeric value of the sequence size of the larger one. While most people seem to have followed this rationale and obtained correct results, one participant explicitly detailed not being able to find the numeric value in the details block of the tree view window, and the option of another (the second larger node) indicated that the user also resorted to visual analysis only. From these results, we can infer that all

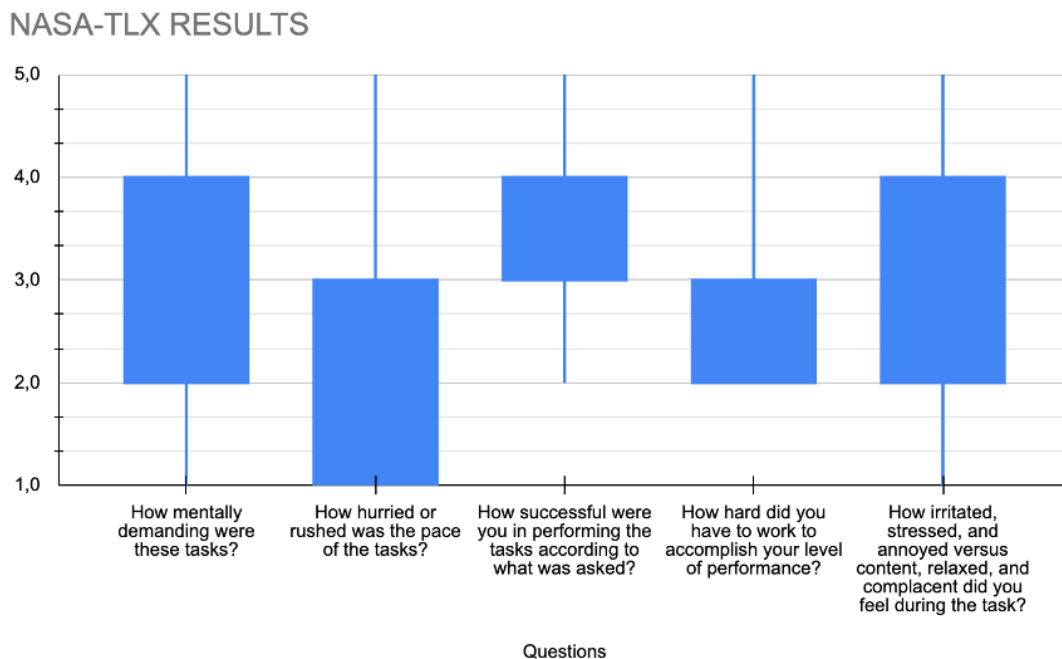


Figure 6.5: Results boxplot of NASA-TLX Questionnaire.

the participants that obtained a visualization were able to perform the visual analysis by selecting the option that allows visualizing the sequence count in each node, and a minority was not able to conduct the full analysis by not locating the numeric value in the details block.

Considering the NASA-TLX obtained averages, these six tasks' perceived effort and success markers convey mostly positive and/or expected results. The average mental demand and the performance-effort questions were around the middle of the scale, as expected. The analysis tasks/questions (LTI4 to LTI6) needed self-learning and tool discovery to be performed, even with the initial prototype presentation, especially for users with no previous experience with visual analysis tools for Immunology. A low score for the time constraint question was also expected since the tasks were not timed. The frustration and stress markers can be derived from the waiting times in order to obtain the tree visualization interface component. The high average score on the perception of task performance success indicates that participants were able to complete the tasks and found this interface component to have a fairly high usability score. From all of these results and their analysis, it can be concluded that the tool performs well in terms of the usability of the lineage tree visualization component, even if some assistance or training may be needed for the analysis portion of the prototype.

6.5 Immunology Visual Analysis

This questionnaire section aimed to understand how users would interpret the obtained visualization regarding immunology analysis. This section consisted of three questions with this goal,

VA1 to VA3, which all asked for a longer, free-form answer. The first two questions were asked after performing a base task:

For the following tasks, please answer freely after obtaining the following views for clone 695490: - an integrated view of all the time points in a regular mode, called "All timepoints" - a side-by-side view of each time point (if not created before), in regular mode, each called "TP1", "TP2" and "TP4" respectively.

The two questions following this task are:

- VA1 - Consider the side-by-side views of each time point. Are the selection processes we see in the lineage from an earlier time point consistent with the past predicted by the later lineage?
- VA2 - After obtaining the views mentioned above, explain what might be happening to the structure of the tree in each of the single time point views when comparing to each corresponding subtree when integrated with the other time points, in the "All timepoints" view.

The last question was based on a screenshot of time point 4 and an integrated tree with all time points for clone 683843:

VA3 - Based on the image provided above and below of the trees of another clone, what could be happening to the structure of the single time point tree when integrated with the rest of the time points?

6.5.1 Results

Regarding question VA1, 30.8% of the provided answers were simply yes, not detailing an explanation, while 38.5% of the participants were unsure, given the considered test scenario and question. Similar results were obtained when looking only at answers from participants that considered themselves proficient in immunology (3 or larger on the Likert scale). One participant mentioned possible discrepancies between the structure of the single trees and the integrated tree due to the presence of sequences from other time points.

In the answers provided to the VA2 question, most participants identified that the structure of the subtrees was maintained in some nodes, but others suffered some small alterations. One particular participant with experience in Immunology analysis mentioned that they found changes in structure for time points 2 and 4 and provided an analysis answer that suggested that some of the sequences found in the second time point tree that were not expanded may have been already present in the previous time point but were not sampled. They also identify the location of the subtrees for the later time points in relation to the earlier in the integrated tree. Another participant with no experience in Immunology analysis was capable of identifying that the parenthood relationships between some nodes changed when looking at them in the single time point visualization versus the integrated view. More than one participant considered that drawing conclusions from establishing comparisons between the structure of single time point trees and the integrated tree can lead to misleading results.

When considering the responses to the VA3 question, the ones that contain an analysis answer belong in the vast majority to participants with proficiency in Immunology and lineage tree analysis. Participants mostly refer to how the single time point tree visualization nodes were integrated into the tree that comprises all the time points, separating the nodes and placing them together with other time points' nodes. One participant predicted that the tree-building system was placing the nodes in different places in the single time point tree when compared to the integrated tree. Other participants stated that the only change was an increase in information and displayed data.

For all of these tasks and corresponding questions, two users were unable to obtain any visualizations.

6.5.2 Discussion

Although these questions required long-form answers, the results of the first question VA1 were inconclusive. The answers specified by the participants did not, in their majority, deliver the necessary information to extract meaningful results. Plenty of participants were not able to understand how the question applied to the presented scenario, and a wider test set, with more clones and more diversity of trees, would have produced more significant analyses.

Question VA2 's results show that with the help of the tool and by interacting with it, both users with strong Immunology knowledge and with no prior experience are capable of analyzing and extracting information from the lineage tree visualizations, albeit with different depths. A large number of the provided answers corresponded to possibilities of expected analysis.

With question VA3, the users did not need to perform a task and create a new tree view window and were only provided with screenshots of the lineage tree visualizations. Consequently, the users were not able to provide as much insight and did not have the same capability to produce an actual analysis when compared to the previous question, where users could explore and click on the nodes and analyze them. A considerable number of users with no previous experience with the field were not able to provide any answers without access to those capabilities. Users with more experience in the Immunology field, however, were still able to provide, at the very least, predictions that were close to some of the possible expected answers, such as the way trees are built (even if built with maximum likelihood and not with parsimony as mentioned by one participant) affecting the structure of the subtrees and changing the placement of nodes.

6.6 SUS Post-Test Questionnaire

The main goal of the SUS questionnaire (ref) is to assess the perceived usability of each participant with an accessible and fast approach. Therefore, this questionnaire was presented after all the required tasks to make that assessment. The following Sections present the results obtained with the SUS tool and their discussion and interpretation.

6.6.1 Results

The average answer for each question of the SUS questionnaire was calculated and is displayed in Figure 6.6. In Figure 6.7, a box plot chart illustrates the median, minimum and maximum values, and lower and higher quartiles.

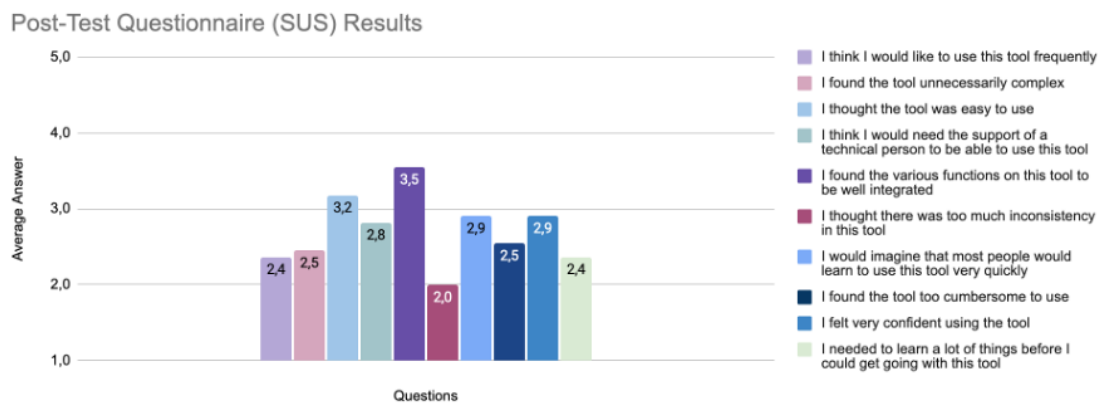


Figure 6.6: Average Results of the SUS Post-Test Questionnaire.

6.6.2 Discussion

With the obtained averages for each task of the questionnaire, some conclusions can be derived on the overall usability of this tool, even if based on subjective metrics.

The first metric measured consists of how frequently a user would use the presented tool. The obtained value for all of the participants was a low score on the scale. This can be attributed to, beyond the fact that some participants are not in the Immunology field, this tool not being as relevant to the person's own specific research and not the relevancy of the tool for the field.

The second, third, and fourth metrics are deeply related to one another. The calculated average answers for each, respectively 2, 3, and 3, reveal that although the complexity of the tool is adequate for the tool, the users perceived its use to have some degree of difficulty. Taking into account the previously presented results, it is likely that some interface components are more intuitive than others, particularly in relation to the interpretation and analysis of the rendered visualizations. It is important to take into consideration that this average includes the answers of participants with no previous experience in Immunology who are likely to have a greater need for technical support.

The following item assessed the integration of the prototype, which received an average high score on the Likert scale. The participants averaged answers that would assert that this is a consistent tool overall. These are important characteristics for this prototype to have since integrating all its components well is crucial to obtaining the full intended analysis.

In terms of the learning curve for the tool, the obtained average puts it around the middle of the scale. This, again, is consistent with the previously presented results across the different components of the interface and varies greatly, having participants put it on both ends of the scale independently of their proficiency in the field.

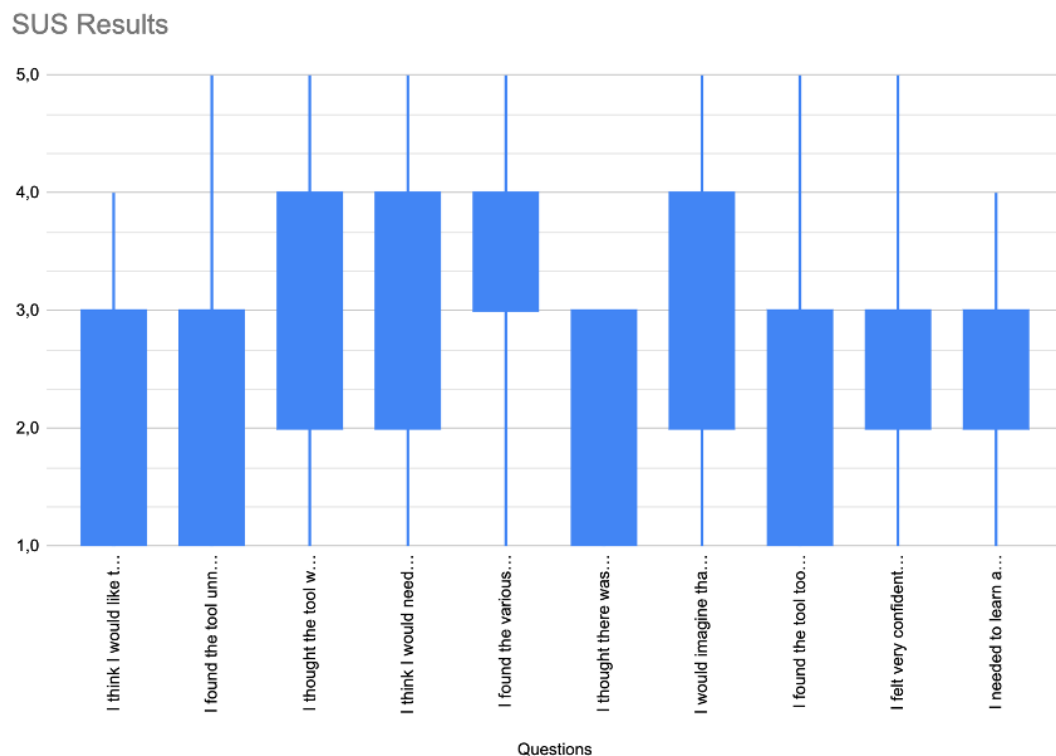


Figure 6.7: Results boxplot of SUS Questionnaire.

Just as with the previous item, the next two metrics have a great variation, with participants attributing values from one end to the other of the scale. Since the SUS intends to measure perceived usability, these values are highly subjective. Other metrics with more concentrated values can, therefore, provide a more certain insight into the usability of the prototype.

The average low score on the last item shows that, although some analysis needs previous specific knowledge, even a user with little to no specific scientific knowledge can use the tool and extract correct information from it.

6.7 Open Questions

The last section of the questionnaire, whose results are detailed in the following sections, prompted the participants to supply their personal suggestions and opinions on the visualization solution prototype. This questionnaire section consisted of two questions.

6.7.1 Results

The first question's answers provided suggestions based on personal preference and necessity. Nevertheless, there were similarities found between the answers and consistent suggestions across participants. The more frequent suggestions consisted of the ability to export/download trees both

in visual and Newick format (23.1% of the answers), for there to be available documentation to learn how to use the tool (23.1% of the answers), and the improvement of data requests' waiting time or loading indicators (30.8% of the answers). There were also some interface improvements suggestions, namely improving, in the filtering stage, the way the selection is made for clone size and top clones, so it is easier to select smaller-sized clones or specific sizes, and improving the way metadata is selected/unselected in the tree view window form. There were several suggestions for improving the interaction and presentation of the time series stacked chart visualization. Suggestions included displaying numeric values for clone sizes in each time point so they would be easily identified, visualizing, and selecting smaller stacks more easily, for example, by having an alternative table chart with the same clones that could also be displayed side-by-side, visualizing clone size in different ways, namely as percentages of the dataset. There was also a suggestion to add the possibility of visualizing mutation substitutions and another to add a "filter the dataset" feature, which would allow a user to directly visualize a clone by inputting its identifier.

In the second question, users expressed their personal opinion on the usefulness of the tool in terms of the analysis of B-cell lineage trees across time. Overall, participants considered the tool to be helpful, particularly in the provided context with databases and with the addition of the improvements suggested in the previous question. Mostly, participants with previous experience with Immunology tools were the ones able to provide significant opinions on the relevancy of the tool. However, a participant with little to no experience in the field stated that even with no training, they were capable of extracting information and performing an analysis on the lineage trees. Another participant considered that to truly conclude the tool's adequacy, it would be necessary for a benchmark comparison with other visualization tools. Two participants with Immunology proficiency considered that the tool would not be useful for their research or context. Several participants considered that introducing an interactive tool such as this makes it easier to analyze datasets, namely for people with little experience. One of these participants mentioned how, with the proper improvements, this tool would remove the need for coding knowledge to extract visualizations from immunological data.

6.7.2 Discussion

The results from these final questions were highly subjective, given their free-form and opinion-based nature. These answers provide significant suggestions that can be implemented in future work in order to perfect this prototype, as well as a good understanding of the usefulness of a tool of this kind for Immunology specialists. Many of the suggestions obtained can be the answer to improving the results presented throughout this Chapter, namely providing documentation for the usage of the prototype.

The main takeaway is that this is a prototype that, by implementing the necessary improvements to the interface, provides a tool with a well-rounded embedded analysis, simplifying the process of obtaining visualizations for the evolution of B-cell clones.

6.8 Summary

Chapter 6 details the obtained results from the evaluation performed with several participants and analyzes and discusses them to understand the usability and performance of the developed prototype.

In Section 6.1, an overview of the participants' personal information was documented. The participants were distributed fairly equally between male and female gender and came from Portugal, the USA, the UK, Spain, and Israel. 61.5% of participants considered themselves proficient in visual analysis tools, and 61.5% considered they were proficient in Immunology or lineage tree analysis.

Section 6.2 presents the results for the functional tasks and corresponding questions of the Filtering Stage of the questionnaire. There were four tasks, F1 to F4, and three questions that intended to assess the success rate of these tasks and the usability of the interface component obtained from them, the clone evolution time series visualization. From the answers registered to these questions and the results extracted from them, it was possible to understand that the users successfully used the filtering stage, but it was necessary to document well the usage of the clone evolution stacked area chart component in order to simplify its usage and interpretation.

Through the results presented in Section 6.3, the correct option was chosen by 84.6% of the participants. Thus, it was deduced that the interaction with the clone evolution stacked graph was overall successful.

In Section 6.4, the lineage tree visualization tasks' results are presented and discussed. This section presents both the results from the functional tasks LTI1 to LTI6 and the results of the post-task questionnaire (NASA-TLX). In questions LT1 to LT3, the high rate of correct answers indicates that participants were able to correctly use the interface's window manager and create new tree view windows and tree visualizations. For questions LTI4 to LTI6, although the success rate for all the correct answers was not always very high, the other answers indicate that people could still fulfill the tasks at least partially and could understand the expected rationale. Regarding the NASA-TLX answer's averages, the perceived effort and success markers for the six tasks convey mostly positive and/or expected results.

Section 6.5 regards the results from the Immunology Visual Analysis tasks, VA1 to VA3. The results for VA1 were inconclusive and did not provide the necessary insights. For question VA2, by interacting with the tree visualization, were able to assert different conclusions on the tree structures, even without previous Immunology knowledge. However, on question VA3, since users did not have access to the interaction and were drawing conclusions only from an image, the provided insights were not as meaningful.

The results and discussion of the SUS Post-Test Questionnaire are presented in Section 6.6. Although this assessment is highly subjective, it was possible to extract information on different parameters of perceived usability of the tool, such as the fact that the tool is adequately complex but would benefit from learning aids and that the tool is well integrated.

Finally, in Section 6.7 the results of the open questions asked in the last section of the questionnaire with the objective of gathering suggestions and insights on the tool. The more frequent suggestions for improvement of the prototype were to implement an export function, both for trees and tree images, to provide documentation or tutorials on how to use the tool and improving the waiting times for the data requests. Overall, the participants considered that this tool was helpful in the analysis of B-cell lineage trees for different time points.

Chapter 7

Conclusion

This chapter details the conclusions taken from the work developed for this study. In Section 7.1, the overall conclusions from all the stages of research are documented, and the obtained results are tied in with the identified issues and the hypothesis postulated at the beginning of this work. The propositions for possible improvements and new features that would further this visualization solution are presented in Section 7.2.

7.1 Conclusions

This work intended to address the gap in immunological data visualizations of B-cell receptor evolution throughout time. The advances in technology that made high throughput sequencing possible created very large volumes of sequence data from BCRs, clones, and correspondent lineages for each studied individual at many time points. Consequently, it was recognized, as detailed in Section 3.1, that this volume of data creates the necessity of exploring datasets differently and representing the evolution of BCR repertoires across time through visualization.

In order to have a global understanding of the evolution of the B cells through time, it was established that two issues needed to be addressed: visualizing not only isolated lineage trees but representing them in a way that enables comparisons between the different points in time and creating a visual overview that demonstrated the diversity of clones and their presence across time in an immune response.

According to the reviewed literature presented in Chapter 2, several existing visualization solutions enabled only the visualization of interactive isolated lineage trees. However, these software tools did not contemplate visualizing an overview of the evolution of a B cell repertoire nor separate lineage trees for different time points, and only some allowed for integrated metadata analysis. Therefore, it was found that none of the studied tools fulfilled the identified issues. Through the study of information visualization state-of-the-art and the necessary immunology context, it was possible to design and implement a visualization solution that attempted to respond to the previously recognized necessities. By understanding the immunology context and knowing available analysis tools and data standards, it was possible to choose IgPhyML software to build lineages

from the AIRR-compliant sequence data extracted from ImmuneDB. The revision of information visualization data-based and task-based taxonomies, together with the analysis of possible suitable visual variables to represent desired BCR and clonal information, made it possible to later design a visualization solution that was based on the developed hypothesis of this work.

The established hypothesis, in accordance with the identified problem and literature review, considered that a visualization solution for analyzing the evolution of B cell clone lineages would consist of an initial filtering process that resulted in a set of data rendered into an interactable stacked area chart that displays the evolution of the presence of clones in time. Clones selected by clicking on a stack should produce lineage tree visualizations for each and all the possible time points. These lineage tree renders could also be interacted with and analyzed.

A novel conceptual solution that intended to corroborate this hypothesis was therefore created, and it encompassed the three main components described above. The proposed solution followed a visualization pipeline adapted to the present work, which helped establish the data flow and structure of the eventually developed prototype. The development of this solution also contemplated which and how visual variables should be used in the stacked area chart and the lineage trees components, such as color together with a pattern as an identifier of time for the nodes of trees.

From the designed concept, a prototype was implemented to facilitate testing the proposed hypothesis and solution. The prototype implemented a database layer with two databases, a server layer with the APIs, and a client layer that created the interface that renders the solution. While developing the lineage tree interface component and building lineage trees with IgPhyML, it became clear that it was necessary to implement algorithms that helped ensure that the rendered trees had the desired structure, such as an algorithm that calculates mutations for each tree branch. The interface of the prototype included all the components presented in the conceptual solution.

As mentioned in the previous paragraph, a test experiment was developed to assess the usability of the implemented prototype and attempt to prove or refute the proposed hypothesis. This experiment consisted of a form that contained a group of tasks and questions that were meant to evaluate the usability of each interface component detailed in Chapter 3 and questions to evaluate if the solution fulfilled the necessary requirements to support the necessary comparisons and analysis between lineage trees at different time points or that include all time points. This experiment was conducted remotely and unmoderated in order to include participants from different countries and sufficient knowledge of both digital visual analysis tools and the immunology field.

The results obtained from the aforementioned experiment demonstrated that if the relevant improvements were made to the interface, this would be a useful tool to allow a fast and detailed analysis of the evolution of a repertoire of BCRs. It was also assessed that this tool could also remove the necessity of possessing coding knowledge to extract meaningful visualizations of immunological data.

After obtaining the clone evolution stacked chart from the filters tasks, some of the presented questions had a lower success rate, which has led the authors of this study to believe that this is the component of the solution which needs more additional features and improvements to create a truly

useful tool. However, from the questionnaire section that followed, it was possible to understand that users had no problem interacting with this graph, and, therefore, it was a successful way of selecting clones in order to visualize their lineage trees. The lineage tree tasks and questions demonstrated that users were capable of successfully using the window manager by creating tree-view windows with distinct characteristics. It was also possible to conclude that users were also able to interact successfully with the tree visualization in order to obtain more detailed information, although there was an increase in difficulty in understanding how to obtain certain parameters.

Considering the results obtained both on the functional tasks and on the immunology visual analysis tasks and the work developed to produce the conceptual solution and the functional prototype, possible answers to the research questions introduced in Section 3.1 are detailed below:

- **How can we effectively create a global overview and understanding of the evolution of a B cell repertoire across time?** The clone evolution stacked chart was implemented to provide an overview of the presence and diversity of a specific set of clones throughout time. This graph, as reviewed in the literature, enables the comparison between the different stacks, being possible to identify more significant members. However, as shown in Section 6.2 and in this Section above, this is a complex graph that can be difficult to interpret. It may also not be the most suitable for analyzing separate clones, which is important in order to select a clone to further analyze with the following interface components. Nevertheless, this can still provide a meaningful global view of the repertoire and filtered subsets, especially if complemented with documentation and tutorials and/or additional textual information on or next to the chart.
- **How can we aid reasoning and understanding the evolution of B cell clones, and the quality of an immune response, through visual representation of both lineage trees for isolated time points and single trees for all time points per clone, since each type of tree is constructed differently?** In order to create a better understanding of the evolution of B cell clones, it was important to create a relation between elements of different lineage trees from distinct time points, particularly when relating a selected time point with the set of previous time points. It was also fundamental to detect the discrepancies between the structure of single time point trees and a tree with all the time points for a specific clone. This was achieved first by creating a window manager that enabled the creation of different lineage tree visualizations, both of single and multiple time points, that could be compared side-by-side. In the visualization itself, color and pattern were used to visually map different time points, which aided the recognition of specific subtrees and their structure of a determined time point both in the single time point tree and the integrated tree. The representation of metadata visually and textually for each tree node further helped group nodes into significant categories that can also help understand the clonal evolution by other measures than time. Beyond the visual variables used, the interaction with the nodes and edges of each tree enabled the display of detailed information, such as identifiers, that provided a more concrete analysis of these trees. The results presented in Section 6.5 indicate that these

features aid in the analysis of the tree, particularly the interaction features, which enabled even inexperienced participants in the field of Immunology to extract information, namely on the tree structure, and establish comparisons between the trees. The results obtained from Chapter 6 overall strongly indicate that this tool and its components and features are helpful for the evaluation of the quality of an immune response.

7.2 Future Work

Given that this work consists of a first approach to a novel visualization solution of the evolution of B-cells in an immune response, there are several improvements and new features that can be explored in the future. Consequently, the following points were identified as possible work to be conducted in future iterations of this study:

- **Creating a more extensive filtering solution.** In the prototype developed for this solution, only a few essential filters were implemented. It would be useful, for a more detailed analysis, to have more filtering options that produced results that follow distinct criteria that are not possible to obtain with the current tool. An example of a possible filter that could help display more meaningful results would be a filter that allows selecting filters by a minimum number of unique sequences at each time point. This would create the possibility of never having clones with a big presence at a specific time point that completely disappear in another.
- **Improving the clone presence evolution chart.** As observed in Section 6.2, there are difficulties in interpreting stacked area graphs, given their base complexity. For example, single clones are harder to analyze with this graph, and color alone was insufficient to convey information. Other types of visualization techniques could be explored for presenting the clone repertoire evolution overview, or supporting text and secondary graphs could be added to improve the usability of this chart.
- **Implementing an export option for trees.** As suggested by the evaluation experiment's participants, after creating a tree view window that contains a lineage tree visualization, a feature could be implemented to export trees both in their Newick format and in a visual format, such as an SVG.
- **Exploring different ways of establishing relationships between lineage tree visualizations of the same clone.** Considering one of the more important questions identified in this study was how to establish relationships between lineage tree visualizations of different time points, this would be an interesting feature to develop further. In the present work, lineages could be compared side-by-side, the nodes' color encoded the time points it was found in, and the nodes and branches of the tree could be interacted with to display more detailed information on them. Additionally, a visual link between similar nodes or subtrees could be a possible option to explore.

- **Exploring different layouts and the prototype's interaction sequence.** Other layouts of the interface could be explored in order to provide the user with a full picture of the visualization, and not only section-by-section. Also, implementing other types of interaction, such as zooming both on the stacked area chart and trees and collapsing certain desired branches/subtrees, could improve the analysis by focusing on more specific areas of the visualization. In addition, it would be useful to create a way of going directly to the creation of the lineage tree visualizations section when the user intends to visualize a specific clone.
- **Creating an improved evaluation experiment.** There were some limitations to how the evaluation experiment could be conducted due to the necessity of having Immunology specialists among the participants. These limitations consisted mainly of the experiment being conducted remotely and unmoderated. Since not holding the experiment remotely would be extremely difficult, conducting this experiment with a moderator present would enable not only employing other metrics, such as time of task completion but would also make interviews with the participants possible, and it would be possible to clear any doubts present in the participants. The presence of a moderator would also ensure that the answers to longer-form tasks/questions would be sufficient and enable more meaningful results. Another feature that could be added was establishing comparisons and benchmarking specific features of the present prototype with features of other existing tools when adequate.

Other aspects of the overall system could be improved, such as the way requests are made to the server in order to minimize the waiting times and create a more seamless experience or improving the placement of information and metadata in the lineage tree visualization.

References

- [1] AIRR Community. URL: <https://www.antibodysociety.org/the-airr-community/>.
- [2] AIRR Community Data Standards. URL: <https://github.com/airr-community/airr-standards>.
- [3] D3 - Data-Driven Documents. URL: <https://d3js.org>.
- [4] Immcantation Portal. URL: <https://immcantation.readthedocs.io/en/stable/about.html>.
- [5] Visual variables. URL: <https://www.axismaps.com/guide/visual-variables>.
- [6] Abul Abbas, Andrew Lichtman, and Shiv Pillai. *Basic Immunology E-Book: Functions and Disorders of the Immune System*. 6th edition, 1 2019.
- [7] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. Visualizing time-oriented data-A systematic view. *Computers and Graphics (Pergamon)*, 31(3):401–409, 2007.
- [8] Wolfgang Aigner, Silvia Miksch, Heidrun Schumann, and Christian Tominski. *Visualization of Time-Oriented Data*. Human-Computer Interaction Series. Springer London, London, 2011.
- [9] B Alberts, A Johnson, and Lewis J. *Molecular Biology of the Cell*. Garland Science, New York, 4th edition, 2002.
- [10] J. M.Christian Bastien. Usability testing: a review of some methodological and technical aspects of the method. *International Journal of Medical Informatics*, 79(4), 4 2010.
- [11] J. Bertin. *Semiology of Graphics: Diagrams, Networks, Maps*. ESRI Press, 2011.
- [12] Michael Bostock and Jeffrey Heer. Protovis: A Graphical Toolkit for Visualization. Technical report.
- [13] Felix Breden, Eline T. Luning Prak, Bjoern Peters, Florian Rubelt, Chaim A. Schramm, Christian E. Busse, Jason A. Vander Heiden, Scott Christley, Syed Ahmad Chan Bukhari, Adrian Thorogood, Frederick A. Matsen, Yariv Wine, Uri Laserson, David Klatzmann, Daniel C. Douek, Marie Paule Lefranc, Andrew M. Collins, Tania Bubela, Steven H. Kleinstein, Corey T. Watson, Lindsay G. Cowell, Jamie K. Scott, and Thomas B. Kepler. Reproducibility and reuse of adaptive immune receptor repertoire data. *Frontiers in Immunology*, 8(NOV), 11 2017.
- [14] John Brooke. SUS-A quick and dirty usability scale.
- [15] John Brooke. SUS: A Retrospective. 8:29–40, 2013.

- [16] S. Card, J. Mackinlay, and B. Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.
- [17] Gabriel Cardona, Francesc Rosselló, and Gabriel Valiente. Extended Newick: It is time for a standard representation of phylogenetic networks. *BMC Bioinformatics*, 9, 12 2008.
- [18] M. Sheelagh T. Carpendale. Considering visual variables as a basis for information visualisation. 2003.
- [19] Chaomei Chen. Information visualization. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):387–403, 7 2010.
- [20] Scott Christley, Ademar Aguiar, George Blanck, Felix Breden, Syed Ahmad Chan Bukhari, Christian E. Busse, Jerome Jaglale, Srilakshmy L. Harikrishnan, Uri Laserson, Bjoern Peters, Artur Rocha, Chaim A. Schramm, Sarah Taylor, Jason Anthony Vander Heiden, Bojan Zimonja, Corey T. Watson, Brian Corrie, and Lindsay G. Cowell. The ADC API: A Web API for the Programmatic Query of the AIRR Data Commons. *Frontiers in Big Data*, 3, 6 2020.
- [21] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys*, 41(1), 12 2008.
- [22] Brian D. Corrie, Nishanth Marthandan, Bojan Zimonja, Jerome Jaglale, Yang Zhou, Emily Barr, Nicole Knoetze, Frances M. W. Breden, Scott Christley, Jamie K. Scott, Lindsay G. Cowell, and Felix Breden. ireceptor: A platform for querying and analyzing antibody/b-cell and t-cell receptor repertoire data across federated repositories. *Immunological Reviews*, 284(1):24–41, 2018.
- [23] Alan Dix, Janet Finlay, Gregory D Abowd, and Russell Beale. *Human-computer interaction*. 2004.
- [24] Jean Daniel Fekete. The InfoVis Toolkit. In *Proceedings - IEEE Symposium on Information Visualization, INFO VIS*, pages 167–174, 2004.
- [25] Brian David Fisher. Illuminating the Path: An R&D Agenda for Visual Analytics Personalized health visual analytics View project HOT Admin Security Project View project. 2005.
- [26] Mathilde Foglierini, Leontios Pappas, Antonio Lanzavecchia, Davide Corti, and Laurent Perez. Ancestree: An interactive immunoglobulin lineage tree visualizer. *PLoS computational biology*, 16(7):e1007731, 2020.
- [27] Andrew U Frank. Different Types of "Times" in GIS.
- [28] Rishi R Goel, Sokratis A Apostolidis, Mark M Painter, Divij Mathew, Ajinkya Pattekar, Oliva Kuthuru, Sigrid Gouma, Philip Hicks, Wenzhao Meng, Aaron M Rosenfeld, Sarah Dysinger, Kendall A Lundgreen, Leticia Kuri-Cervantes, Sharon Adamski, Amanda Hicks, Scott Korte, Derek A Oldridge, Amy E Baxter, Josephine R Giles, Madison E Weirick, Christopher M Mcallister, Jeanette Dougherty, Sherea Long, Kurt D'andrea, Jacob T Hamilton, Michael R Betts, Eline T Luning Prak, Paul Bates, Scott E Hensley, Allison R Greenplate, and E John Wherry. Distinct antibody and memory B cell responses in SARS-CoV-2 naïve and recovered individuals after mRNA vaccination. 6:6950, 2021.

- [29] Namita T. Gupta, Jason A. Vander Heiden, Mohamed Uduman, Daniel Gadala-Maria, Gur Yaari, and Steven H. Kleinstein. Change-O: a toolkit for analyzing large-scale B cell immunoglobulin repertoire sequencing data. *Bioinformatics*, 31(20):3356–3358, 06 2015.
- [30] Sandra G Hart. NASA-TASK LOAD INDEX (NASA-TLX); 20 YEARS LATER. 2006.
- [31] Sandra G. Hart and Lowell E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. pages 139–183. 1988.
- [32] Kenneth B. Hoehn, Gerton Lunter, and Oliver G. Pybus. A phylogenetic codon substitution model for antibody lineages. *Genetics*, 206(1):417–427, 5 2017.
- [33] Kenneth B Hoehn, Jason A Vander Heiden, Julian Q Zhou, Gerton Lunter, Oliver G Pybus, and Steven H Kleinstein. Repertoire-wide phylogenetic models of B cell molecular evolution reveal evolutionary signatures of aging and vaccination. *Proceedings of the National Academy of Sciences*, 116(45):22664–22672, 2019.
- [34] Andy Kirk. *Data Visualisation: A Handbook for Data Driven Design*. 2 edition, 2019.
- [35] Page Laubheimer. Beyond the NPS: Measuring Perceived Usability with the SUS, NASA-TLX, and the Single Ease Question After Tasks and Usability Tests, 2 2018. URL: <https://www.nngroup.com/articles/measuring-perceived-usability/>.
- [36] William D. Lees and Adrian J. Shepherd. Studying antibody repertoires with next-generation sequencing. In *Methods in Molecular Biology*, volume 1526, pages 257–270. Humana Press Inc., 2017.
- [37] Ivica Letunic and Peer Bork. Interactive tree of life (iTOL) v5: An online tool for phylogenetic tree display and annotation. *Nucleic Acids Research*, 49(W1):W293–W296, 7 2021.
- [38] Shixia Liu, Weiwei Cui, Yingcai Wu, and Mengchen Liu. A survey on information visualization: recent advances and challenges. *Visual Computer*, 30(12):1373–1393, 12 2014.
- [39] A.M. MacEachren. *How Maps Work*. The Guilford Press, New York, USA, 1995.
- [40] Wolfgang Müller and Heidrun Schumann. Visualization methods for time-dependent data - An overview. In *Winter Simulation Conference Proceedings*, volume 1, pages 737–745, 2003.
- [41] Quang Vinh Nguyen and Mao Lin Huang. A space-optimized tree visualization. In *Proceedings - IEEE Symposium on Information Visualization, INFO VIS*, volume 2002-January, pages 85–92. Institute of Electrical and Electronics Engineers Inc., 2002.
- [42] Jakob Nielsen and Thomas K Landauer. A Mathematical Model of the Finding of Usability Problems. 4 1993.
- [43] Gary Olsen. The” Newick’s 8: 45” tree format standard, 1990. URL: <http://evolution.genetics.washington.edu/phylib/newick/doc.html>.
- [44] Kathrin Pieper, Bodo Grimbacher, and Hermann Eibel. B-cell biology and development. *Journal of Allergy and Clinical Immunology*, 131(4):959–971, 4 2013.
- [45] Aaron M. Rosenfeld, Wenzhao Meng, Eline T. Luning Prak, and Uri Hershberg. ImmuneDB, a novel tool for the analysis, storage, and dissemination of immune repertoire sequencing data. *Frontiers in Immunology*, 9(SEP), 9 2018.

- [46] Robert E. Roth. Visual Variables. In *International Encyclopedia of Geography: People, the Earth, Environment and Technology*, pages 1–11. John Wiley & Sons, Ltd, 3 2017.
- [47] Adrian Rusu. 5 Tree Drawing Algorithms.
- [48] Anna. Sachinopoulou and (Otamedia). *Multidimensional visualization*. Technical Research Centre of Finland, 2001.
- [49] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE Symposium on Visual Languages*, pages 336–343. IEEE Comput. Soc. Press.
- [50] Ben Shneiderman, Cody Dunne, Puneet Sharma, and Ping Wang. Innovation trajectories for information visualizations: Comparing treemaps, cone trees, and hyperbolic trees. *Information Visualization*, 11(2):87–105, 4 2012.
- [51] Alice Thudt, Jagoda Walny, Charles Perin, Fateme Rajabiyazdi, Lindsay Macdonald, Riane Vardeleon, Saul Greenberg, and Sheelagh Carpendale. Assessing the Readability of Stacked Graphs.
- [52] Jordan R. Willis, Bryan S. Briney, Samuel L. DeLuca, James E. Crowe, and Jens Meiler. Human Germline Antibody Gene Segments Encode Polyspecific Antibodies. *PLoS Computational Biology*, 9(4), 2013.
- [53] Gur Yaari, Martin Flajnik, and Uri Hershberg. Questions of Stochasticity and Control in Immune Repertoires, 11 2018.
- [54] Ji Soo Yi, Youn Ah Kang, John T. Stasko, and Julie A. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 11 2007.
- [55] Guangchuang Yu, David K. Smith, Huachen Zhu, Yi Guan, and Tommy Tsan Yuk Lam. ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. *Methods in Ecology and Evolution*, 8(1):28–36, 1 2017.

Appendix A

Questionnaire

A.1 Questionnaire

The form used to conduct the evaluation experiment can be consulted at: <https://forms.gle/yFQ4en7UChC2ZDkV8>.

The screenshot shows a Google Form titled "Usability Test B-cell Lineage Tree Visualization". The form is set by the user "up201207603@g.uporto.pt" and is marked as "Not shared". A red asterisk indicates a required question. The form includes an "Informed Consent Form" section with the following text:

You are invited to participate in a research study on the user experience of the prototype of a lineage tree visualization tool for ImmuneDB databases. This is the concluding section of my Master's Thesis, whose topic is on visualization of B-cell Lineage Trees. With this study, our goal is to assess the usability and effectiveness of a human-computer interface process composed of filtering tasks and visual analysis tasks towards the analysis of B-cell lineage trees, whose data sources is sampled from more than one time point.

We would like you to attempt to access the prototype and use it according to the provided instructions. Throughout this session, several questions will be asked about the tasks you'll perform and your opinions and experience while using the tool. The answers to these questions will be collected and used to perform a statistical analysis that will help us get a better understanding of the usability of the prototype.

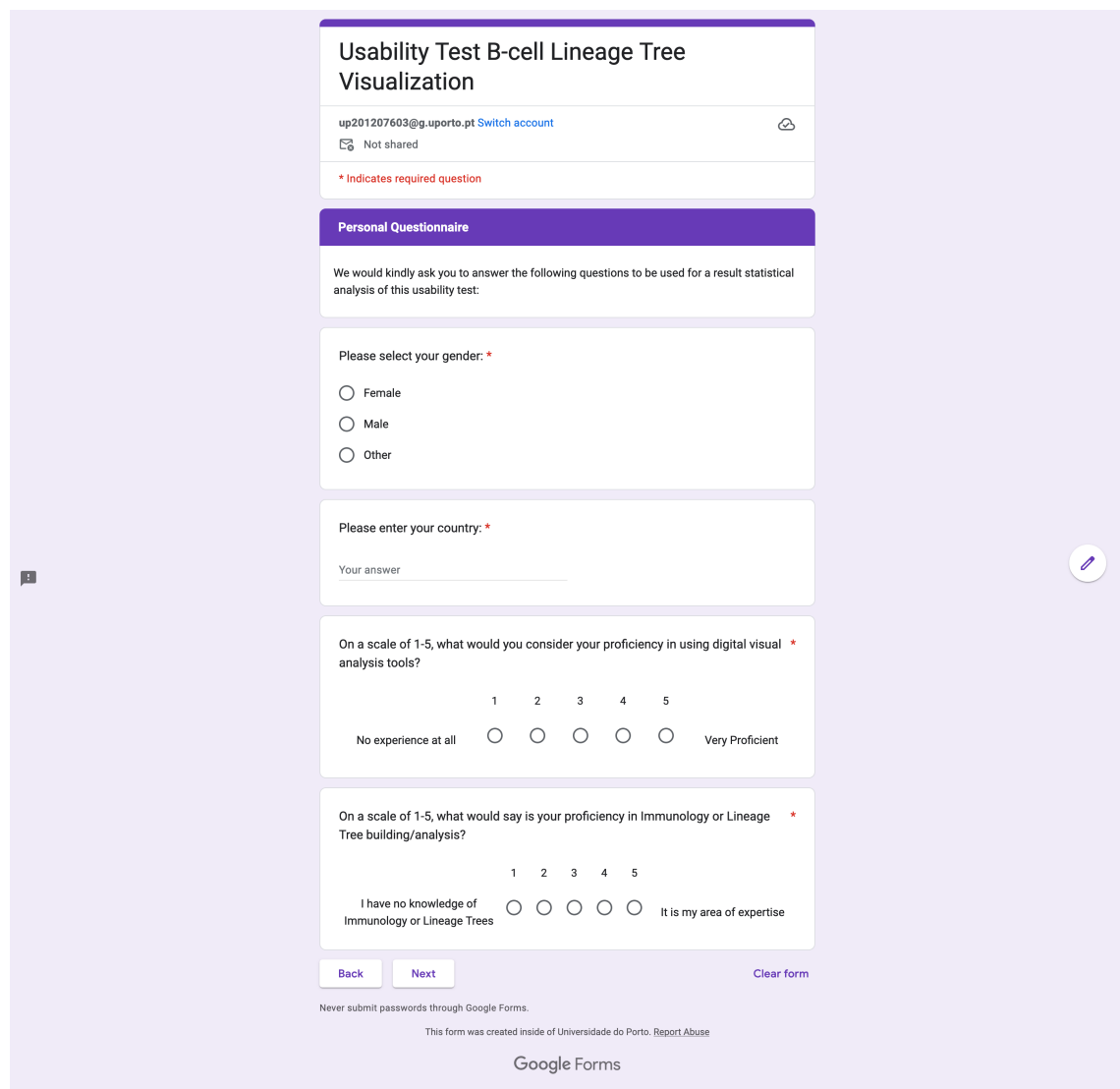
There will also be some personal data collected at the beginning of this experiment, namely your gender, your country and your level of knowledge in immunology and level of proficiency with digital visual analysis tools. Your anonymity will be kept throughout this study.

By clicking that you accept the conditions stated here, you are giving permission for your provided personal data and all data collected during to this study to be used for this research's purpose.

If you have read this form and have decided to participate in this project, please understand your participation is voluntary, and you have the right to withdraw your consent or discontinue participation at any time. Your identity will not be disclosed in any published and written material resulting from the study.

If you choose to withdraw your participation, please get in touch with Leonor Freitas at up201207603@up.pt.

Below the consent text is a question: "Do you agree with the conditions stated above? *". There is an unchecked checkbox labeled "I accept". At the bottom of the form, there are "Back" and "Next" buttons, and a "Clear form" link. The footer includes a disclaimer: "Never submit passwords through Google Forms." and "This form was created inside of Universidade do Porto. Report Abuse". The Google Forms logo is at the bottom.



The screenshot shows a Google Form titled "Usability Test B-cell Lineage Tree Visualization". The form is for a user named "up201207603@g.uporto.pt" with a "Switch account" link. It indicates the form is "Not shared". A red asterisk indicates a required question.

Personal Questionnaire

We would kindly ask you to answer the following questions to be used for a result statistical analysis of this usability test:

Please select your gender: *

☐ Female
☐ Male
☐ Other

Please enter your country: *

Your answer

On a scale of 1-5, what would you consider your proficiency in using digital visual analysis tools? *

1 2 3 4 5
No experience at all ☐ ☐ ☐ ☐ ☐ Very Proficient

On a scale of 1-5, what would say is your proficiency in Immunology or Lineage Tree building/analysis? *

1 2 3 4 5
I have no knowledge of Immunology or Lineage Trees ☐ ☐ ☐ ☐ ☐ It is my area of expertise

[Back](#) [Next](#) [Clear form](#)

Never submit passwords through Google Forms.
This form was created inside of Universidade do Porto. [Report Abuse](#)

Google Forms

A.2 Results

The raw answers to the questionnaire can be found at: https://docs.google.com/spreadsheets/d/1ACNRW2_hBVU2VCvqty4ElCEXN7XxNMwJ9FtM-QCPkSg/edit?usp=sharing.

Usability Test B-cell Lineage Tree Visualization

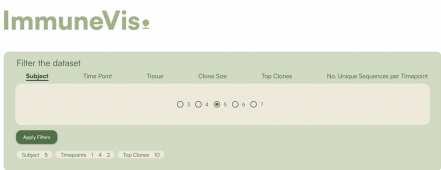
up201207603@g.uporto.pt [Switch account](#)

🔒 Not shared

Prototype Presentation

This lineage tree visualization process is composed of a sequence of three stages: the filter of clones, the stacked area chart visualization and clone selection and the lineage tree visualization and analysis.

Filtering Stage



Stages of ImmuneVis

The process starts by filtering clones, according to several criteria. Clone filter (illustrated in the previous figure) is used to get the sample to be used for visualization.

The first filter is by subject. Mandatory. Only a single subject can be selected since clone evolution is only meaningful per individual.

The second filter is about time points at which each sample is collected in the experiment and that clones appear on. One or more time points can be selected.

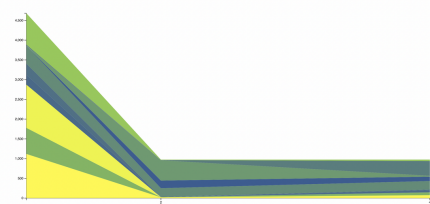
The third filter addresses the tissue organ. All of the possible sample origin tissues are listed, the user can select one or more (if available in the dataset).

The last filter defines the clone sizes in one of two ways:
(1) min and max clone size - the min /max range of the total count of sequences in each clone.
(2) the top n clones by size in total sequence counts.

Once the subject filter is selected the clone size filter and the top clone filter are updated so that only values that are possible for the selected subject are available to be chosen.

These filters are meant to be used in an integrated fashion.

Stacked Area Chart



Clone stacked visualization

The stacked area chart (illustrated in previous figure) is displayed right after the user finishes filtering clones.

The horizontal axis represents time, with the time points of sample collection marked on it while the vertical axis represents the size of each unique clone (using the total count of clones). Each stack represents the evolution of a single clone over the selected time periods.

The process continues by selecting one of the unique clones from the stack, in order to proceed to the last stage of the process: lineage tree visualization. This display of the next stage may take up to a few minutes to complete.

Lineage Tree Visualization

By selecting a clone in the previous chart, the metadata and related information about all the sequences found for that clone's lineage trees are fetched from the database. Once this data is available, the process can continue by clicking on the **New tree view**. This interaction shows a dialogue (illustrated in the next figure) allowing the user to define the configuration for a new tree view.

Later the user can add, hide or delete already created tree views.

Add a new tree view

X

View Name *

Time Point(s)
☐ 1 ☐ 2 ☐ 4

Mode
☐ Side-by-Side ☐ Integrated

Tree Layout
☐ Regular ☐ Radial

Metadata

Add View

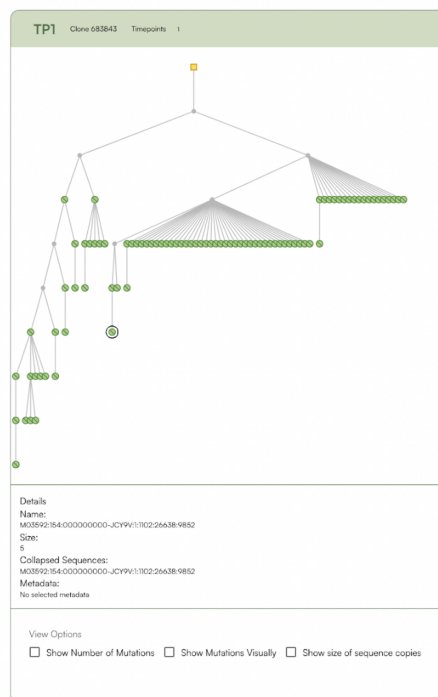
Configure Tree Visualization Options

The participant needs to

1. **name** the view
2. select which **time points** (one or more) to visualize
3. The type of lineage tree integration:
 - a) **side-by-side**: the lineage trees for each time point are shown, as the name indicates, next to each other. In the case that only one of the time points is selected, a single lineage tree for that time point will be shown.
 - b) **integrated view**: a representation of the evolution of the tree across all time points. If all the available time points are selected, the tree will appear with all nodes colored. Otherwise, the ones that were not selected will appear greyed out on the all-time point tree.
4. select the layout type: a **regular** layout or **radial** layout.
5. the clone's sequences **metadata** field to share. The visual representation of the selected metadata can also be **displayed** in the tree.

Finally, by clicking the **Add View** button the lineage tree is rendered (illustrated in the next figure).

Tree View Example



Lineage tree visualization

In addition to the tree display, this view has some more visualization options:

- a) a list of details on the selected node or branch;
- b) the options to visualize the number of mutations on each node and the size of each node.

How trees are being calculated (under the hood)

The visualization of lineage trees requires the assembly of data supported on the sequences retrieved from ImmuneDB in AIRR format: clustered by clone, and organized by subject and time point, with all the corresponding germlines. The data is then used to build Newick trees with the help of the IgPhyML tool.

Once a Newick tree is generated through IgPhyML, all the mutations for each node are calculated and the tree is collapsed, both if to the distance to the parent of each node is smaller than the defined threshold or if, after the calculations, the number of mutations of a node in relation to its parent is zero.

Tree visual components

After each tree is built and the mutations are calculated for each branch, the tree is finally displayed. It is composed of nodes and branches.

Nodes

There are three types of nodes:

1. The germline, represented by a yellow square shape;
2. "Black" nodes, that are inferred when the tree is built, with no known sequence. Represented with a smaller dark node;
3. Nodes that have known sequences and all its related information, represented by a round shape and colored according to the time point they belong to. The fill pattern consists of a line in a specific direction that also represents the time point to reinforce the visual cue for time. Nodes that belong to more than one time point, (a sequence is found in more than one time point) are painted with a color that is the intersection of the colors of the time points it belongs to.

Furthermore:

- a) a selected node is also represented by a circle surrounding it. For this node it is possible to view its related information on the details window (name, metadata and collapsed sequences). b) the diameter of the node circles is also change proportionally to its size, when the corresponding option (visualize the size of the sequence) is selected.
- c) when metadata categories are selected on the tree view creation, it is possible to visualize them on the tree by selecting from a drop-down menu a shape to attribute to a value of a category of metadata. There is a maximum of five values that can be visualized (one value per shape). Once the shapes are attributed, they show up next to the nodes that possess the metadata value corresponding to each shape.

Branches

Branches can also be selected and are thus a) represented by a dashed stroke and b) the related information is presented in the details window (the supporting nodes and the mutations that occur between them).

Furthermore:

- a) the thickness of the branches changes proportionally to its number of mutations when the option to show mutations visually is enabled. b) the number of mutations appear close to the branch when the option to show the number of mutations is enabled. This is not available for radial tree layout.

Usability Test B-cell Lineage Tree Visualization

up201207603@g.uporto.pt [Switch account](#)

Not shared

* Indicates required question

Filter Tasks

In the prototype, start by using the filtering stage to perform the following tasks and answer the related questions.

F1 - Select subject 5.

F2 - Select clones from timepoints 1, 2 and 4.

F3 - Select top 10 clones.

F4 - Combine all these filters and obtain the stacked area chart.

Select the correct statements about the obtained stacked area chart. *

☐ The total sum of the size of the chart increases exponentially from time point 1 to 4.

☐ Clone 691147 increases its size significantly from time point 1 to time point 2.

☐ There is a clone that keeps a constant size throughout time (throughout the evolution of the chart).

☐ In this chart, yellow represents larger stacks and blue represents smaller stacks.

☐ None of the above is correct.

Count the number of stacks you see in the obtained stacked chart and provide it below. *

Your answer

In your opinion, the stacked area chart: *

☐ Helps understand the evolution of the clones' presence over time.

☐ Gives a good overview of the size of each clone, in order to choose one for analysis.

☐ None of the above

Back

Next

Clear form

Never submit passwords through Google Forms.

This form was created inside of Universidade do Porto. [Report Abuse](#)

Google Forms

Usability Test B-cell Lineage Tree Visualization

up201207603@g.uporto.pt [Switch account](#)

Not shared

* Indicates required question

Stacked Area Chart Tasks

SAC1 - Hover over the stacked area chart for the corresponding clone id to appear over each stack. Click on the stack of the clone 695490. (Please note that after clicking on a stack, it may take some minutes to fetch all the information needed for the next stage)

After you clicked on clone 695490 in the stacked area chart, a waiting message * appears. Once this message is hidden:

- ☐ A button to add a new visualization appears, and when clicked a form is presented.
- ☐ A form to create a new tree view appears immediately.
- ☐ I obtained nothing after selecting the stack.

[Back](#) [Next](#) [Clear form](#)

Never submit passwords through Google Forms.

This form was created inside of Universidade do Porto. [Report Abuse](#)

Google Forms

Usability Test B-cell Lineage Tree Visualization

up201207603@g.u.upto.pt [Switch account](#)

Not shared

* Indicates required question

Lineage Tree Interaction Tasks

For the following tasks, please use the lineage tree creation and visualization section that appeared after clicking on the stack on the previous stage and after the loading of the sequences and metadata is finished.

LT11 - Create a view for the current clone (chosen in stacked area chart) with the name "Integrated TP 1 & 2", for time point 1 and 2, in integrated mode and with a regular tree layout. Select also the option to visualize age metadata. When the tree view appears, select the correct option: *

☐ There are two trees.

☐ Some of the nodes are green.

☐ I obtained nothing after creating the view.

LT12 - Create a view for the current clone (chosen in stacked area chart) with the name "TP 1", for time point 1, in integrated mode and with a radial tree layout. When the tree view appears, select the correct option: *

☐ There are only purple nodes on this tree.

☐ There are green nodes and greyed out nodes on this tree.

☐ I obtained nothing after creating the view.

☐ There are only green nodes on this tree.

LT13 - Create a view for the current clone (chosen in stacked area chart) with the name "TP 4", for time point 4, in side-by-side mode and with a regular tree layout. When the tree view appears, select the correct option: *

☐ The view has two trees side by side, one with different colored nodes and another with only one color nodes.

☐ There is a single tree with all the colored nodes purple.

☐ I obtained nothing after creating the view.

LT14 - Get the collapsed sequences for the node whose name ends in 8243 for the tree view "TP 4". Please write the last set of digits on the name (4 or 5 digits). *

Your answer

LT15 - Get the target node name for all branches with more than seven mutations on the view "TP4". Please write the last set of digits on the name (4 or 5 digits). In the case there is a node that does not have a name, add 'black node' to the list. *

Your answer

LT16 - Find the name of the node(s) with the largest number of sequence copies for the tree for time point 4 clone number 695490. Please write the last set of digits on the name (4 or 5 digits). *

Your answer

Post Task Questionnaire

Please answer the following questions about your experience while performing these tasks:

How mentally demanding were these tasks? *

1 2 3 4 5

Very Low ☐ ☐ ☐ ☐ ☐ Very High

How hurried or rushed was the pace of the tasks? *

1 2 3 4 5

Very Low ☐ ☐ ☐ ☐ ☐ Very High

How successful were you in performing the tasks according to what was asked? *

1 2 3 4 5

Failure ☐ ☐ ☐ ☐ ☐ Success

How hard did you have to work to accomplish your level of performance? *

1 2 3 4 5

Very Low ☐ ☐ ☐ ☐ ☐ Very Hard

How irritated, stressed, and annoyed versus content, relaxed, and complacent did you feel during the task? *

1 2 3 4 5

Very Low ☐ ☐ ☐ ☐ ☐ Very High

Back

Next

Clear form

Never submit passwords through Google Forms.

This form was created inside of Universidade do Porto. [Report Abuse](#)

Google Forms

Usability Test B-cell Lineage Tree Visualization

up201207603@g.uporto.pt [Switch account](#) Draft saved

* Indicates required question

Visual Analysis Tasks

For the following tasks, please answer freely after obtaining the following views for clone 695490:

- an integrated view of all the time points in a regular mode, called "All timepoints"
- a side-by-side view of each time point (if not created before), in regular mode, each called "TP1", "TP2" and "TP4" respectively.

VA1 - Consider the side-by-side views of each time point. Are the selection processes we see in the lineage from an earlier time point consistent with the past predicted by the later lineage? *

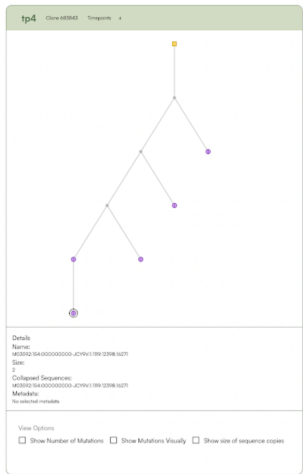
Your answer

This is a required question

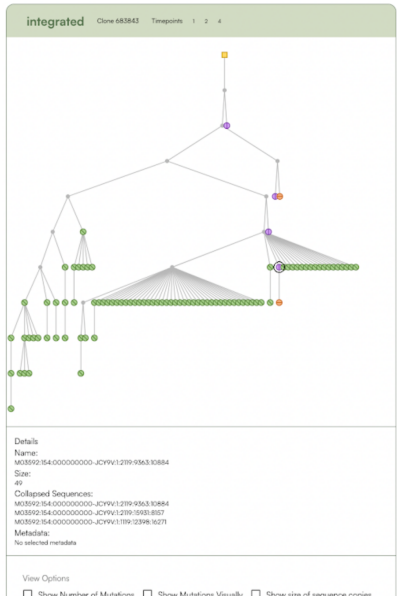
VA2 - After obtaining the views mentioned above, explain what might be happening to the structure of the tree in each of the single time point views when comparing to each corresponding subtree when integrated with the other time points, in the "All timepoints" view. *

Your answer

VA3 - Single Time Point Tree (See question below)



VA3 - Based on the image provided above and below of the trees of another clone, what could be happening to the structure of the single time point tree when integrated with the rest of the time points? *



Usability Test B-cell Lineage Tree Visualization

up201207603@g.uporto.pt [Switch account](#)

Not shared

* Indicates required question

Post-Test Questionnaire

Thank you for completing all the tasks! Please answer the following questions regarding your experience while using this prototype.

I think I would like to use this tool frequently *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I found the tool unnecessarily complex *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I thought the tool was easy to use *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I think I would need the support of a technical person to be able to use this tool *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I found the various functions on this tool to be well integrated *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I thought there was too much inconsistency in this tool *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I would imagine that most people would learn to use this tool very quickly *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I found the tool too cumbersome to use *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I felt very confident using the tool *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree

I needed to learn a lot of things before I could get going with this tool *

1 2 3 4 5

Strongly Disagree ☐ ☐ ☐ ☐ ☐ Strongly Agree


[Back](#) [Next](#) [Clear form](#)


Never submit passwords through Google Forms.

This form was created inside of Universidade do Porto. [Report Abuse](#)

Google Forms

Usability Test B-cell Lineage Tree Visualization

up201207603@g.uporto.pt [Switch account](#) 

 Not shared

* Indicates required question

Additional Questions

Please answer the following questions about the tool you just tested.

What other features would you like to see implemented in this prototype? Please specify in which stage/part of the prototype this feature(s) would exist and provide a small explanation to why you think it would improve the purpose and usability of this tool. *

Your answer

Do you feel like this tool allowed you to better analyze the evolution of B-cell Lineages across time? Please explain the reason for your answer. *

Your answer

[Back](#) [Submit](#) [Clear form](#)

Never submit passwords through Google Forms.

This form was created inside of Universidade do Porto. [Report Abuse](#)

Google Forms

Appendix B

Prototype

The source code of the prototype can be consulted at: https://github.com/leonormfreitas/MS_C_PROJECT.

The prototype is deployed and can be used at: <https://multitimevis.inesctec.pt/>