POLITECNICO DI TORINO

Master degree course in Computer Engineering

Master Degree Thesis

# A toolbox for the analysis and visualization of miRNA and isomiR expression levels

**Advisors**
Prof. Gianvito Urgese
Dr. Walter Gallego Gomez

**Candidate**
Ermes La Porta

December 2025

# Abstract

MicroRNAs (miRNAs) are a type of short non-coding RNA sequences involved in crucial biological processes. In fact, multiple studies show that they can be used as biomarkers for human diseases such as cancer and Parkinson's as well as phylogenetic markers for species evolution. Thanks to the huge amount of high-quality RNA reads that Next Generation Sequencing is able to produce, the analysis of miRNA has improved over the years, with computer science playing an increasingly important role in this bioinformatics field.

miRNAs are characterized by a mechanism that mostly relies on a specific nucleotide region to bind to messenger RNA, so when performing miRNA alignment it is crucial to prioritize the conservation of this so-called seed sequence. Among a multitude of available software for this task, isomiR-SEA stands out as an optimized tool that pays attention to the specific miRNA characteristics. It performs a seed-based alignment, leading to higher accuracies than those of general purpose aligners. Moreover, it accurately identifies the multiple isoforms that a miRNA family can have (isomiRs). For greater efficiency, isomiR-SEA is often paired with BioSeqZip, a collapsing tool to preprocess the input data.

The aim of this thesis is to build a modern toolbox for the analysis of miRNA (iSEA-TB) around a new unpublished version of isomiR-SEA, to prove its computational capabilities and its flexibility, while offering the user a reproducible analysis pipeline and an intuitive graphical user interface for results visualization and downstream analysis. The first component of iSEA-TB is a pipeline written in Nextflow, an open-source workflow management tool largely adopted in bioinformatics. The pipeline automatically runs all the steps required to perform a complete miRNA analysis: data download, quality check, trimming, collapsing, alignment, expression levels estimation, and results consolidation. To make the pipeline work, some modifications were coded into BioSeqZip and isomiR-SEA to allow them to correctly interface with the processed data. The second component of the iSEA-TB is a database-powered interactive analysis interface that allows users to interrogate the results obtained from the pipeline in the form of SQL queries, and build visualizations useful for downstream analysis, such as miRs and isomiRs expression levels, miR region conservation, and A2I substitutions. The interface was built using the open source software Grafana, which allows seamless interfacing with databases

and offers multiple data visualization and navigation modes.

The execution report by Nextflow was used to evaluate the pipeline performance. It includes runtime and memory usage that were compared against those of state-of-the-art tools. The pipeline is able to analyze considerable amounts of data in reasonable times, thanks to the modern C++ implementation of isomiR-SEA and BioSeqZip, proving that this combination of tools is ideal for large scale miRNA analysis. To demonstrate the usability of the GUI, two meaningful datasets were analyzed: raw RNA reads used by the MirGeneDB3.0 database, and a collection of human primary cell reads from the human microRNAome. The GUI produces interactive graphical representations that can dynamically show different sets of the obtained results, which in turn will facilitate the comparison between miRNA isoform expression levels of the requested selection. This flexible interface will certainly prove helpful for future pathological and philological studies.

# Contents

**List of Figures** 7

**List of Tables** 8

**1 Introduction** 11

**2 Background** 13

2.1 DNA and RNA sequencing . . . . . . . . . . . . . . . . . . . . . . . . 13

2.1.1 Next Generation Sequencing . . . . . . . . . . . . . . . . . . . 14

2.1.2 Adapters and trimming . . . . . . . . . . . . . . . . . . . . . . 14

2.2 MicroRNAs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

2.2.1 Biogeneseis . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

2.2.2 Biological functions . . . . . . . . . . . . . . . . . . . . . . . . 16

2.2.3 Philological functions . . . . . . . . . . . . . . . . . . . . . . . 17

2.2.4 MicroRNA isomorfs . . . . . . . . . . . . . . . . . . . . . . . . 18

2.3 Public miRNA databases . . . . . . . . . . . . . . . . . . . . . . . . . 19

2.3.1 miRBase v22 . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20

2.3.2 MirGeneDB 3.0 . . . . . . . . . . . . . . . . . . . . . . . . . . 20

2.4 Alignment algorithms and isomiR-SEA . . . . . . . . . . . . . . . . . 21

**3 Methods** 23

3.1 isomiR-SEA . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

3.1.1 A new and improved version . . . . . . . . . . . . . . . . . . . 23

3.1.2 isomir-SEA's workflow . . . . . . . . . . . . . . . . . . . . . . 24

3.1.3 Input files . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

3.1.4 Output Files . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

3.2 Chosen datasets . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

3.2.1 MirGeneDB 3.0 . . . . . . . . . . . . . . . . . . . . . . . . . . 27

3.2.2 Human miROme . . . . . . . . . . . . . . . . . . . . . . . . . 27

3.3 Workflow management: Nextflow and nf-core . . . . . . . . . . . . . 28

3.4 The iSEA-TB pipeline . . . . . . . . . . . . . . . . . . . . . . . . . . 29

3.4.1 SRA info . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

3.4.2 SRA Toolkit . . . . . 30
3.4.3 FastQC . . . . . 30
3.4.4 Infer PHRED . . . . . 31
3.4.5 fastp . . . . . 31
3.4.6 Cutadapt . . . . . 31
3.4.7 miRTrace . . . . . 31
3.4.8 BioSeqZip . . . . . 32
3.4.9 isomiR-SEA . . . . . 32
3.5 Pipeline manual . . . . . 34
3.5.1 Alternative workflows . . . . . 35
3.6 BioSeqZip and isomiR-SEA adaptation . . . . . 36
3.7 GUI and database analysis . . . . . 37
3.7.1 Database construction . . . . . 37
3.7.2 Interactive GUI in Grafana . . . . . 38
3.7.3 Database access and plots generation . . . . . 41

**4 Results** 45
4.1 Tools comparison . . . . . 45
4.1.1 miRge3.0 . . . . . 45
4.1.2 IsoMiRmap . . . . . 47
4.1.3 sRNAbench . . . . . 48
4.2 MirGeneDB dataset . . . . . 49
4.3 Human miROme dataset . . . . . 51
4.3.1 Preprocessing . . . . . 51
4.3.2 Alignment . . . . . 53

**5 Conclusion** 57

**Bibliography** 59

# List of Figures

2.1 Sequencing process with adapter ligation and post sequencing trimming 15
2.2 Biogenesis of microRNAs and its distinctive hairpin shape . . . . . 17
2.3 Representation of bioregulation executed by miRNA via mRNA binding . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
2.4 Different classes of isomiRs . . . . . . . . . . . . . . . . . . . . . . . 19

3.1 Comparison of isomiR-SEA execution time across versions . . . . . 24
3.2 Comparison of isomiR-SEA RAM usage across versions . . . . . . . 24
3.3 The iSEA-TB pipeline . . . . . . . . . . . . . . . . . . . . . . . . . 29
3.4 Example of the .tab output format . . . . . . . . . . . . . . . . . . 37
3.5 Example of the .gff output format . . . . . . . . . . . . . . . . . . . 37
3.6 Example of the structure of one of the database's tables . . . . . . . 38
3.7 The expression level of human (hsa) isomiR for all available tissues 39
3.8 Expression level for Hsa-Let-7-P1b_3p* and Hsa-Mir-10-P2c_3p* . 40

4.1 Comparison between MirGeneDB and Grafana visualization for Hsa-mir-1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
4.2 Visualization of isomiR expression levels for mmu . . . . . . . . . . 50
4.3 Nextflow execution report's CPU Resource graph . . . . . . . . . . 51
4.4 Nextflow execution report's RAM Resource graph . . . . . . . . . . 52
4.5 Nextflow execution report's Job Duration Resource graph . . . . . . 53
4.6 Nextflow execution report's CPU Resource graph . . . . . . . . . . 54
4.7 Nextflow execution report's RAM Resource graph . . . . . . . . . . 55
4.8 Nextflow execution report's Job Duration Resource graph . . . . . . 55

# List of Tables

3.1 General information about the analyzed datasets . . . . . . . . . . . 27
3.2 General information on the modules used in the pipeline . . . . . . 29

4.1 Comparison of alignment tools . . . . . . . . . . . . . . . . . . . . . . . 46

# List of Acronyms

| | |
|---|---|
| **DNA** | Deoxyribonucleic acid |
| **RNA** | Ribonucleic acid |
| **miRNA** | mircroRNA |
| **mRNA** | messengerRNA |
| **isomiR** | mircroRNA isoform |
| **NGS** | Next Generation Sequencing |
| **RPM** | Reads per million |

# Chapter 1

# Introduction

In recent years, as newer technologies got developed, the amount of available biological data increased significantly. The need to analyze this ever growing abundance of information has lead researchers to rely on computer science to aid their studies. Therefore, the development and refinement of software tools for this purpose has led computer science to play an increasingly central role in this field. This combination of disciplines is what today is known as bioinformatics.

Of this interdisciplinary field, one that has received a great deal of attention revolves around the analysis of short RNA sequences and, more specifically, the analysis of microRNA [1]. MicroRNAs (or miRNA) are short non-coding RNA sequences of limited length mostly being around 21-23 nucleotides long, that assume a critical role in the bioregulation of organisms. They bind to messenger RNA (mRNA) in order to prevent the transcription of proteins, regulating the biological functions of the cell. Recent studies have discovered how their expression levels can be used as biomarkers for early identification of human diseases such as cancer and autoimmune disorders. Moreover, they have been proven to be reliable phylogenetic markers for studies about species evolution [2]. Thus, deepening our understanding of miRNA is essential to support the medical field, as well as to gain a deeper understanding of gene expression in different species.

MicroRNAs present further challenges when it comes to their isoforms, commonly known as isomiRs [3]. In fact, with a deeper focus towards their analysis, researchers have found out that miRNAs do not always appear in their canonical form: some are missing some nucleotides at their ends, while others have slightly more (3' isomir and 5' isomir), and others present one or more base changes (single nucleotide polymorphism) [4]. Unfortunately, although a lot of information has been found about canonical miRNAs and their functions, the amount of information revolving around isomiRs is more limited. Although it has been proven that isomiRs perform biological regulatory functions as well by binding to different target mRNA with respect to their canonical counterpart, information about the subject remains sparse. Although many publicly available databases have published information about annotated canonical miRNA, the vast majority provides little to no information about miRNA isoforms.

The high amount of genomic data that Next Generation Sequencing (NGS) is capable of producing from biological samples, as well as the biological importance of short non-coding RNA sequences have, therefore, led to the development of many algorithms and tools throughout the last couple of decades, aiming to ease the process of studying and understanding miRNA's biological role. The process of analyzing raw data obtained from NGS is not trivial. The obtained reads contain errors, low quality sequences and short artificial sequences used during the sequencing process known as adapters. Thus, before trying to identify microRNAs and their isoforms in a sample, it is crucial to perform a precise preprocessing of the raw data.

The final step of the analysis is what is referred to as alignment. During the last few decades, a multitude of software has been developed to tackle this challenge. However, much like before, this procedure presents its own challenges. Different tools base their analysis on different libraries, some of which are too general purpose to be considered accurate. Some tools have underwhelming performances in terms of speed or require too much RAM for large inputs. Finally, the level of detail in the reported output varies greatly depending on the chosen tool.

Among the multitude of published tools, one that stands out for its efficiency and level of detail on output is isomiR-SEA [5, 6]. This highly competitive tool bases its analysis around a specific region of the miRNA sequence called "seed", and exploits it to identify not only annotated microRNA, but also isomiR sequences, as well as provide a variety of information about each aligned read like, among others, the interaction site with mRNA.

As mentioned, the preprocessing step represents a solid challenge for users who want to perform accurate miRNA and isomiR analysis, both in choice of the correct tool for the task and the correct arguments to provide (ie. adapter sequences). This task can be tackled in such a variety and the information provided with the raw sequences is so limited that it is easy for a user to make mistakes.

This thesis proposes a comprehensive pipeline centered around the use of isomiR-SEA, developed in Nextflow [7] for the analysis of miRNA and isomiR. This pipeline was named isomiR-SEA-ToolBox, or iSEA-TB for short, and is meant to automate the delicate process of analyzing raw reads obtained via NGS. Furthermore, this work aims to prove the computational capability, accuracy, flexibility and general usability of isomiR-SEA itself, as well as promote its use to fill the current gap in knowledge around isomiRs which, as previously stated, is sparse even among the most recognized public miRNA databases. For this last purpose, Grafana [8] has been exploited in order to develop an interactive graphical representation of the isomiR expression levels obtained via iSEA-TB, easing the analysis of the obtained data for potential future studies.

# Chapter 2

# Background

This chapter provides an overview of microRNAs, a type of short non-coding RNAs involved in several biological processes, and their isoforms known as isomiRs, of the public databases available for the studies of such micoRNAs and of the alignment tools specifically designed for the task, with a particular focus on isomiR-SEA.

## 2.1 DNA and RNA sequencing

**Deoxyribonucleic acid**, more commonly known as **DNA**, is a molecule composed of two polynucleotide chains that coil around each other to form a double helix. Said chains, as the name suggests, are a combination of nucleotides which are composed of one base of either cytosine, guanine, adenine or thymine. The two chains connect with each other by pairing bases: cytosine with thymine and guanine with adenine. The specific combination of those bases carries the genetic code of an individual organism [9].

Its single stranded counterpart is **ribonucleic acid**, more commonly known as **RNA**. Apart from usually being single stranded, there is also a difference in its nucleotide composition, where the thymine is absent and replaced with uracil. Although most commonly known for transcription of DNA sequences for the synthesis of proteins, depending on the specific type of RNA it is actually responsible for more biological functions that will be discussed more in depth later in this chapter.

The sequencing procedure is the process that allows the extraction of DNA and RNA sequences by identifying the nitrogenous bases (adenine, cytosine, guanine and thymine/uracil) that compose it. Over the years, different sequencing procedures have been proposed and refined. For the purposes of this thesis we will focus on the most commonly used in the current day: **Next Generation Sequencing**, whose advent represented a turning point with respect to the amount of data that can be generated in relatively short amount of time. Moreover, we will explain some concepts related to **NGS** required to understand this work.

### 2.1.1 Next Generation Sequencing

DNA and RNA sequencing has historically been a long, expensive process: Sanger-sequencing was the first widely accepted procedure for this task, to the point of being known as "first generation sequencing" however, only one sequencing reaction could be analyzed, resulting in a very limited throughput. Since then, a great deal of technological advancements and refining have been made in the field, leading to the introduction of **Next Generation Sequencing (NGS)** technologies. Due to nanotechnology principles and innovations that allowed massive parallel sequencing of single DNA molecules, NGS is nowdays capable of outputting massive amounts of data on a daily basis. When talking about these kinds of processes, it is important to note that DNA is way more adept to being treated than RNA therefore, to sequence RNAs, it is common practice to first convert them into complementary DNA (cDNA). Because of this property, when talking about sequencing, only the DNA will be mentioned. NGS approaches, such as the one on the Illumina platform, usually start from DNA fragmentation and DNA end-repair, then proceed with adapter sequence ligation, surface attachment, and in-situ amplification [9]. Of these steps, the adapter ligation part is of particular interest for the analysis of miRNA since, as we will shortly see, its use influences the end result of the sequencing step. A visualization of the sequencing process that shows adapters can is provided in Figure 2.1.

### 2.1.2 Adapters and trimming

Adapter sequences are short oligonucleotides of variable length that ligate to the ends of DNA fragments of interest. This process allows for successive operations that require the use of primer sequences during the amplification step. However, at the end of the sequencing process, these adapters remain attached to the extracted sequences, impeding alignment tools from recognizing potential miRNAs. This means that these artificial sequences need to be identified and removed (trimmed) from short-RNA sequences before being fed to alignment tools. Unfortunately, over the years a multitude of NGS technologies have been developed: each one of them uses a variety of their own preparation libraries and the adapters used kept changing as newer versions came out. This means that a different adapter sequence can be found within extracted reads, depending on the sequencing technology and version used, making the identification and trimming procedure non-trivial [11, 12, 13].

The trimming process is a computational step which is carried out after the whole sequencing procedure, during the preprocessing phase of miRNA alignment, and involves the use of specialized algorithms. The presence of adapter sequences ligated to raw reads hinders the ability of alignment tools to recognize miRNA sequences, so their removal is a delicate and critical step to carry out. Although there are some algorithms like *fastp* [14] that have some ability to automatically

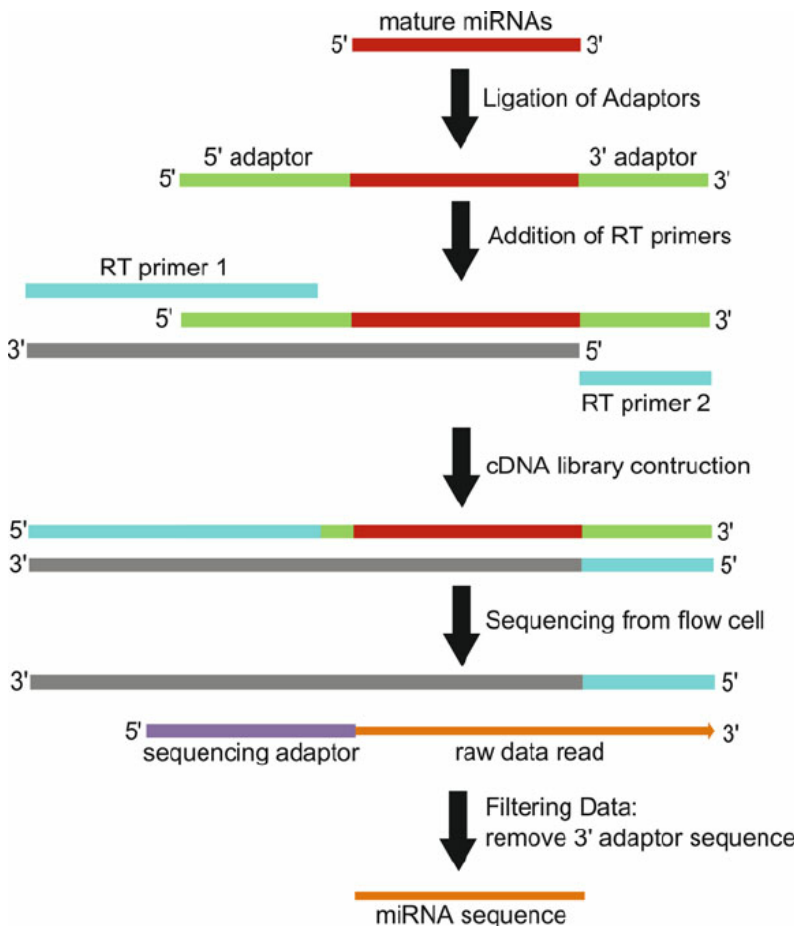


Figure 2.1: Sequencing process with adapter ligation and post-sequencing trimming. Image from Majer et al. [10].

detect adapter sequences, their reliability varies depending on the provided input. Other algorithms like *cutadapt* [15] need the user to explicitly state the adapter sequence that needs to be searched and used, either via input arguments or via a list inside an input file. On paper, this second approach would be more surefire however, the publicly available sequenced reads datasets usually do not explicitly state the adapter used by the preparation library, meaning that it's difficult to know which one has been used a priori. Another problem may arise from typical adapter trimming applications because usually only a short prefix of the full sequence tends to be scanned, which sometimes leads to the detection (and trimming) of false positives. Furthermore, during sequencing some noise in the data tends to be produced and the resulting sequences (both natural and artificial) can contain some errors, meaning that some flexibility towards errors needs to be employed when looking for adapters. However, depending on the amount of tolerance given, this may result in the application detecting a sequence homology instead of an adapter.

Because of all these reasons, there is no perfect way of executing this task over a large quantity of data and, if a bioinformatician aims to automate it, this process needs to be carefully tailored for it to be as accurate as possible.

## 2.2 MicroRNAs

Ribonucleic acid (RNA) is a polymeric molecule responsible for most of the biological functions of living organisms. RNA is very similar to DNA in terms of composition, with the main difference being a base change from thymine to uracil. Its structure also differs from DNA, because RNA tends to be single stranded. This difference allows it to assume much more complex structures, allowing it to fold and form short helices with itself. Depending on the biological function that the RNA undertakes, it gets referred to as a specific class [16]. For example, the messenger RNA (mRNA), whose job is to copy and carry information from the DNA, which then gets used to synthesize proteins.

Non-coding RNAs are a different class of ribonucleic acid that are involved in many cellular processes. Some partake in regulation of the information flow from DNA to protein, others perform RNA splicing, DNA replication, gene regulation and more. Different types of non-coding RNAs include: transfer RNAs (tRNAs) and ribosomal RNAs (rRNAs), as well as small RNAs such as microRNAs, siRNAs and piRNAs.

**MicroRNAs** (miRNA) are short, single-stranded non-coding RNA sequences that make use of diverse mechanisms to regulate gene expressions. They are usually between 21 and 23 nucleotides long sequences that are evolutionary conserved and perform gene expression regulation. They perform their biological task by binding to the target mRNA, in order to silence it and prevent it from synthesizing new proteins.

### 2.2.1 Biogeneseis

miRNAs, as shown in Figure 2.2, are formed as a result of RNA polymerase II, where a strand folds on itself generating a peculiar hairpin shape, commonly referred to as pri-miRNA. Those hairpins then get cleaved into partially double-stranded RNAs called pre-miRNA, before going through another cleaving process that separates the strands generating the sequences that are commonly known as mature-miRNA [18] .

### 2.2.2 Biological functions

miRNAs are known primarily for their regulation of gene expression. By binding their seed region (sequence composed of bases 2–8) to the 3' untranslated region of

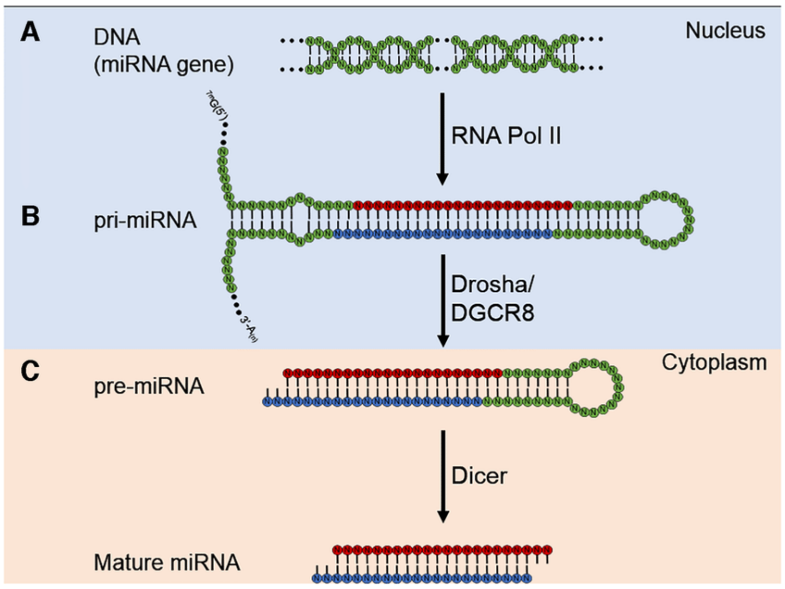


Figure 2.2: Biogenesis of microRNAs and its distinctive hairpin shape.
Image from Chandradoss et al. [17]

complementary messenger-RNA (mRNA), they suppress mRNA's transcription via post-translation repression or by initiating their degradation process. This process can be visualized in Figure 2.3 They are involved in most physiological processes, like cell survival, apoptosis, proliferation and metastasis, and even cellular activities like immune response, insulin secretion and neurotransmitter synthesis [19]. Medical studies have thus proven that, by detecting their deregulation, they can be used as biomarkers for pathologies such as cancer [20], autoimmune disorders [21], and cardiovascular diseases [22]. Furthermore, in recent years, a great amount of effort has been put into researching their use as medical treatment for the mentioned diseases [23]. In fact, in July of this year, an article about Novartis [24], a Swiss pharmaceutical company, has been published, informing of their $800 million cost acquisition of Regulus Therapeutics, whose research into the microRNA field is developing a miRNA-17 inhibiting drug against heart kidney failure. They thus seem to have huge, but yet undiscovered potential in the medical field for both disease recognition and treatment.

### 2.2.3 Philological functions

MicroRNAs have also been proven to be evolutionary conserved in both animals and plants. MirGeneDB [26] has put a great amount of effort on the manual curation and annotation of high-confidence miRNA across the metazoa. They developed a phylogenetic tree, comprehensive of 114 species, that has been carefully developed over multiple versions of the database [27]. All the provided data can be

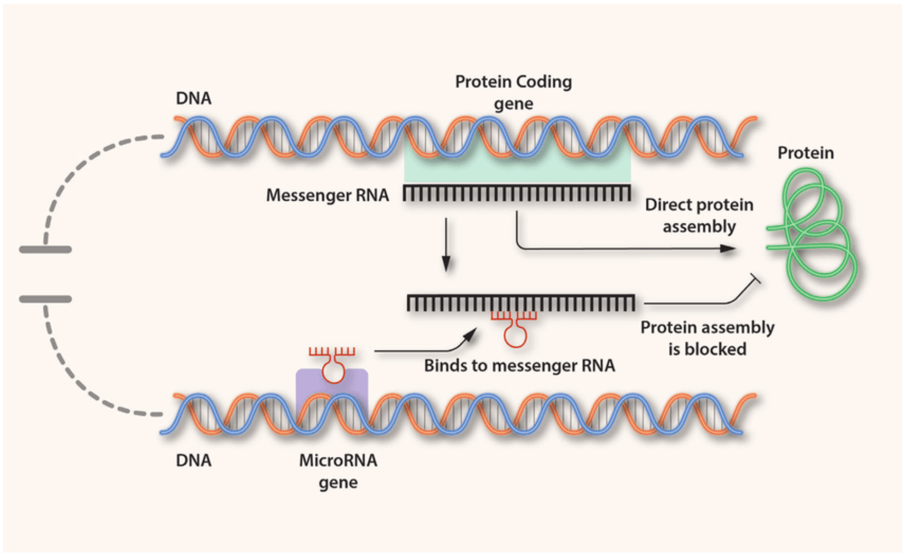


Figure 2.3: Representation of bioregulation executed by miRNA via mRNA binding Image from Yavropoulou et al. [25]

freely browsed, searched, compared and downloaded from the MirGeneDB website, allowing further studies in the philological field. Further studies of gene expressions over different species could provide new insights on miRNAs leading to new advancements in the field.

### 2.2.4 MicroRNA isomorfs

Some initial studies hypothesized that only 2 kinds of mature microRNA could generate from a single hairpin, specifically, one from the 3' arm (3p strand) and one from the 5' arm (5p strand). However, more recent studies that exploited the amount of data coming from NGS, have discovered that multiple variations of a canonical form can be generated, either by insertion, deletion or nucleotide polymorphism. Those miRNA isoforms are commonly referred to as **isomiRs** [28].

isomiRs originate from miRNAs loci as consequence of specific processes as exoribonucleases or nucleotidyl transferase activity, RNA editing, SNPs or imprecise cleavage by the ribonucleases Drosha and Dicer [4]. They differ from miRNA in terms of either composition or length (number of nucleotides) or both. In fact, the miRNA sequence can be segmented in different nucleotide (nt) regions based on interaction sites [19, 29]: the seed region from nt-2 to nt-7, the offset at nt-8, the supplementary region from nt-13 to nt-16 and the overall central site from nt-3 to nt-16 . The leftmost nucleotede of the sequence is called the 5' end, while the rightmost nucleotide is referred to as the 3'end. Depending on the affected region or end, isomiRs can thus be distinguished into three main classes: 3' isomiRs, 5' isomiRs and polymorphic (SNP) isomiRs, which can be seen in Figure 2.4.

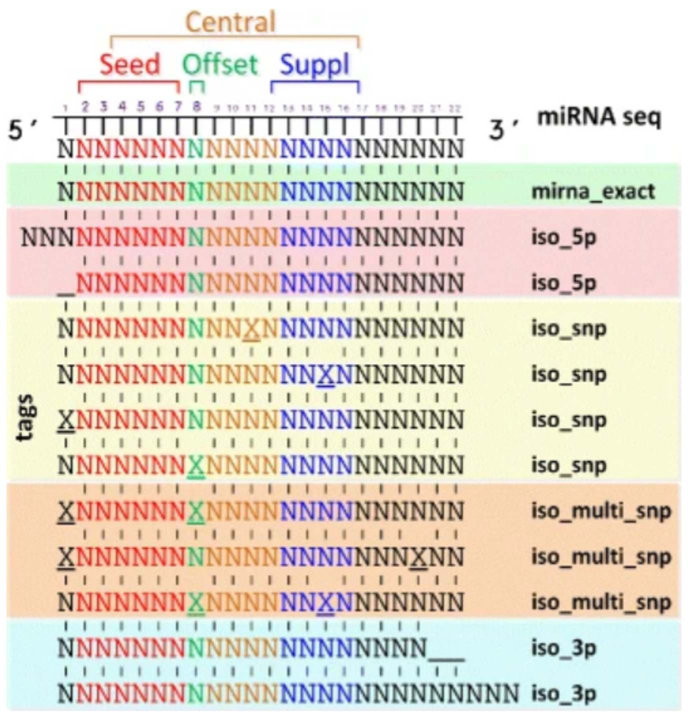


Figure 2.4: Different classes of isomiRs.
Image from Urgese et al. [5]

The knowledge around isomiRs is instead less refined and their functionality is, to this day, subject of study. The primary challenge for researchers seems to derive from their variability, which often leads to misclassifications of other small RNA classes as miRNA isoforms. However, established knowledge on the field is not completely absent. Much like their canonical counterparts, they act at the post transcriptional level via base-pairing. Of the previously mentioned classes, 3' isomiRs are by far the most represented, therefore most research has found success in their analysis in particular [30, 31]. Although they share the same seed region with annotated miRNAs, their ability to differ at the 3' end allows them to bind to different mRNA sequences with respect to their canonical counterpart. Their presence in different organisms and different stages has also been studied, suggesting variations in their biological functions [32]. However, knowledge on the subject is still far from comprehensive and further studies need to be aided in order to increase our understanding on the subject.

## 2.3 Public miRNA databases

An important step in the miRNA alignment procedure, that can have a significant impact in the overall quality of the results, is the selection of the reference genome(s) that will be used to compare and align the samples obtained from the NGS process. These reference genomes are usually obtained from bona-fide public databases such as *miRBase* [33] and *MirGeneDB* [26].

### 2.3.1 miRBase v22

Since its release in 2002, miRBase has become the most recognized public microRNA repository. It is responsible for establishing a gene naming scheme for novel miRNA discoveries, which has been employed to steadily increase its database size across numerous updates. Having reached the 22nd version release, it has collected 38 589 precursor entries, spanning across a total of 271 organisms. Its objective is to not only publish the canonical sequences of high confidence miRNAs, but to also provide a wide range of information regarding them, such as precursors, genomic coordinates, literature reference and more. Reportedly, distinguishing between bona fide miRNAs and mis-annotated RNA sequences has not been an easy task, since miRBase's primary source of novel sequences comes from author submissions and, as has been previously said, the amount of available data and research grows by the day. To address this issue, the sequences reported on the website have been assigned with a confidence value, depending on the amount of data addressing the sequence. To this day, only about 26% of the reported data has been classified as high confidence.

Despite its public recognition, miRBase does not provide a comprehensive list of isomiRs, as its main focus is the annotation of canonical miRNA, leaving a significant knowledge gap in this field. Other public databases like isomiRdb [34] have put some efforts in filling this gap based on miRBase's human data, but none seem to be comprehensive enough in both annotation and species abundance.

### 2.3.2 MirGeneDB 3.0

Previous studies have reported that the main issue with public databases is their lack of curation, which lead to an over-abundance of false positives in reported miRNA sequences (over two thirds) [35, 36, 37, 38]. Another problem that has arisen is the inconsistency or incompleteness of the nomenclature employed to classify the different miRNA families, which often hinders comparative studies. To tackle these problems, MirGeneDB has put great effort into manually curating data revolving around miRNAs, instead of simply accepting sequences from public literature [39]. Released in 2015, MirGeneDB has thus been the first manually curated, publicly available miRNA database. Spanning over 3 released versions, it has grown from just 4, to a total of 114 metazoan species, consistently employing a rigorous method for naming and classifying only high-confidence reads. One of its major objectives was to not only eliminate most, if not all, false positives from its repertoire, but to also aid comparative phylogenetic works. To this end, it has constructed and developed a comprehensive metazoan tree that graphically reports evolutionary closeness between species, and allows browsing for all the annotated miRNAs related to them. This feature emphasizes on the premise that multiple studies have reported that miRNAs are great phylogenetic markers, encouraging further research on the topic. Their efforts have been amply recognized during the years, leading to a multitude of studies and publications [40, 41, 42] relying to MirGeneDB for metazoan miRNA annotation.

In terms of isomiR, MirGeneDB has introduced a field expressing their existence in relation to each miRNA, but the amount of information regarding it is sparse and not nearly enough to fuel significant future studies in the field.

## 2.4 Alignment algorithms and isomiR-SEA

With the advent of Next Generation Sequencing, arose the opportunity to characterize miRNAs on a significantly larger scale, thanks to their ability to generate large amounts of sequence data in a short amount of time. However, to fully exploit the possibilities offered by this technology, it quickly became apparent that appropriate tools specialized for the job needed to be developed. Thus, over the years, an increasing amount of bioinformatic tools for the analysis of miRNAs have been published [43, 44]. Although the analysis of the sequence data obtained via NGS needs to undergo a multitude of steps, what this section will focus on is the alignment procedure.

Alignment algorithms typically make use of reference miRNAs to recognize the sequences that they receive as input, compering each one of them with the reference mature sequences. This process allows for the identification of canonical miRNAs and therefore their expression levels in a tissue or species. Some algorithms limit their capabilities to this job, others instead aim to identify novel unannotated miRNAs, while others expand their research scope to the recognition of isoforms. However, the different alignment logic, different analytical approaches, varied sensitivity and specificity that the various programs are characterized by, lead to a noticeable discrepancy in the obtained results. Moreover, there is a great variety in computational capabilities, both in terms of speed and memory, meaning that hardware capabilities need to be taken into consideration. Finally, there is the problem of the scope of analysis and quantity of information extrapolated from the alignment procedure, which varies greatly depending on the algorithm.

A key aspect that differentiates the various miRNA aligners is how the treat the miRNA sequence. Some tools use general purpose aligners. Others implement alignment algorithms specifically tailored to miRNA. Among this ever growing variety, a tool that stands out in terms of performance, accuracy and completeness in provided information is **isomiR Seed Extension Aligner**, more commonly referred to as **isomiR-SEA** [5]. isomiR-SEA, written in C++ using the SeqAn library [45], focuses its analysis on the "seed" region of the potential miRNA sequence, which is of outmost importance for a stable bind to thee target mRNA, and expands its matching algorithm around it. This approach, specific for miRNA, provides greater accuracy and allows isomiR-SEA to identify both miRNA and isomiR sequences. Moreover it allows the algorithm to provide a variety of information about the aligned sequences, which are extremely useful for downstream analysis.

# Chapter 3

# Methods

The following is a comprehensive description of which tools have been used, their general functionality and their use in this work, their interactions with each other and the adaptations that they received in order to perform the full analysis over the selected datasets.

## 3.1 isomiR-SEA

isomiR-SEA [5] is a miRNA alignment tool that can accurately identify canonical microRNA sequences and respective isoforms, called isomiRs. It's computational efficiency and alignment logic are what set it apart from other alignment algorithms. In fact, isomiR-SEA performs alignment based on the seed region of the full sequence and can also accurately identify miRNA-mRNA interaction site classification, giving it the edge in overall accuracy.

### 3.1.1 A new and improved version

In its first release state, the tool was not fully tested and, therefore, presented some bugs and some computational issues regarding both speed and RAM. However, five years prior to this thesis, a new and improved version was developed, which tackled the aforementioned issues [6].

For this new and unpublished version, the language was updated to a more recent C++17 programming standard. The core library used for the alignment procedure, SeqAn [46], was also manually ported (since no automation is available for the process) from SeqAn2 [45] to the more recent and optimized SeqAn3 [47]. It also implemented the ability to make use of multiple threads, exploiting concurrency and allowing for parallel alignment operations which significantly sped up execution time, as can be seen from Figure 3.1. Lastly, it implemented an on-the-fly output mechanism that significantly decreased the amount of RAM that the tool makes use of, shown in Figure 3.2.

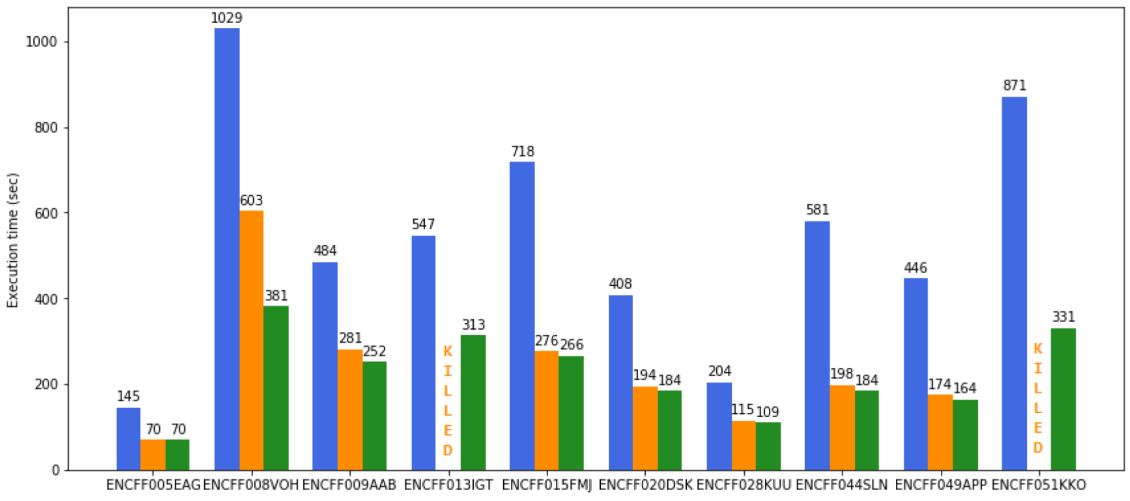


Figure 3.1: Comparison of isomiR-SEA execution time across versions. In blue v1.6, the published version, in orange v1.7 a first unpublished attempt at concurrency implementation, and in green v2.0 [6]. Here can be seen how the implementation of concurrency manages to halve execution time with the single-thread version.

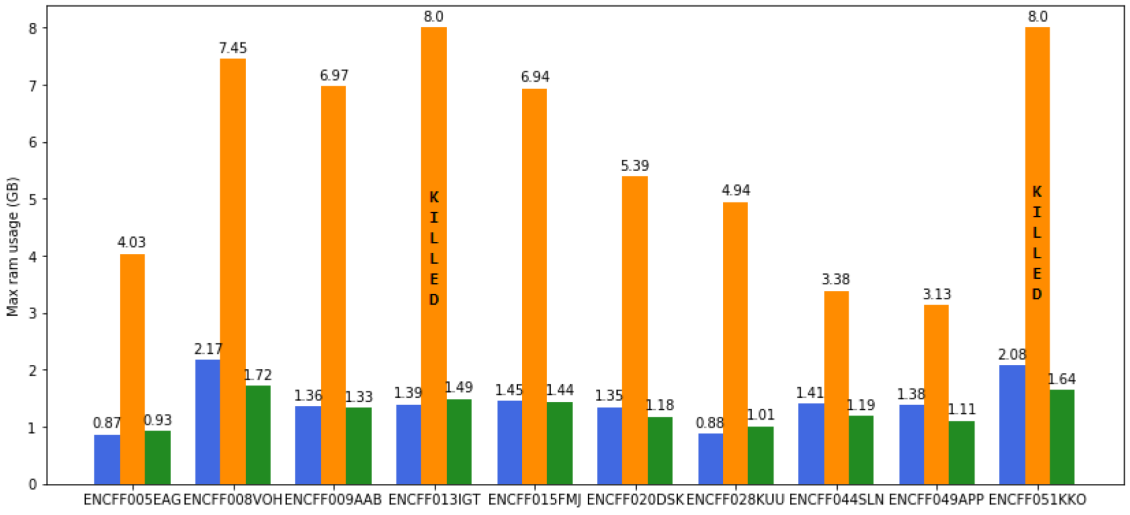


Figure 3.2: Comparison of isomiR-SEA execution time across versions. In blue v1.6, the published version, in orange v1.7 a first unpublished attempt at concurrency implementation, and in green v2.0 [6]. Here is evident how much v1.7 struggled with RAM usage without proper optimization, to the point of being forcefully stopped (KILLED) by the sistem for saturating it.

### 3.1.2 isomir-SEA's workflow

isomiR-SEA's main workflow can be divided into 3 main steps:

- **Preprocessing:** where, after the input parameters are set to produce the desired configuration, the input files are loaded in an appropriate data structure. There is a variety of arguments that can be explicitly changed when given as input parameters, that allow for different behaviors of the execution logic, like number of allowed mismatches in the sequence or specifically in the seed.

- **Alignment:** here the seed region of the miRNA gets compared to the input reads. When a match is found the rest of the read gets compared via both 3' end and 5' end extension. During this process, an arbitrary number of mismatches can be allowed via input argument. Since the seed region is the core part that defines the stability of the interaction between the miRNA sequence and the target mRNA, its exploitation allows for a more accurate miRNA identification and allows for the recognition of target mRNA. The read then receives an alignment score, gets its interaction site evaluated and gets paired with its precursor.

- **Output:** finally, all the recorded information get outputted in both .tab format and .gff format. A third .log file also gets produced with summarized information about the execution.

### 3.1.3 Input files

The alignment happens on .fasta input files, which contain textual representations of potential miRNA sequences, defined by the letters 'A', 'T', 'C' and 'G', each representing a different nucleotide. Those *reads* need to be preprocessed by trimming and quality control tools. Other than the potential sequences, isomiR-SEA needs to receive one or more input files representing the canonical forms of miRNAs, so that the alignment can be matched against them. There are a variety of options to choose from, all of which are available in public miRNA database websites like MirGeneDB:

- It can be provided with both a genomic coordinates file and a precursors file.
- It can alternatively receive a file of mature miRNA sequences and/or a star file.
- Lastly, it can process a combination of both of the previous options.

Furthermore, if paired with BioSeqZip [48] (a collapsing tool from the same authors), it can receive as input multiple potential read files collapsed into a single one, paired with a .tab file that explicitly states how many reads belong to which original file. With just those 2 files, isomiR-SEA is capable of outputting multiple aligned-reads files, one for each original input. This new feature is extremely useful when the input files come from sequencing of the same species, perhaps from different tissues or different stages, or from evolutionary close ones. In those cases, since multiple reads can be repeated across a variety of input files, a normal serialized execution with single input files would need to re-align the same sequence multiple times, resulting in a waste of time and resources. By making use of the tab-file to trace back each count to the original input file, this multi-sample analysis allows the tool to perform alignment on those sequences only once, significantly speeding up computation time.

### 3.1.4 Output Files

As previously mentioned, the output files come in two different formats: .gff and .tab. The .gff (General Feature Format) is a widely accepted standard output format for aligned miRNA reads. It allows for a decent amount of information to be compressed into a single line for the different reads, such as genomic coordinates, score, and an arbitrary list of attributes. isomir-SEA provides more detailed information in a custom format file of extension .tab, where the header describes the different fields represented by each column and the subsequent rows carry all the information that the tool was able to extract from the input sequence, such as count, genomic coordinates, isoform specific information, precursor information, reference miRNA sequence and much more. This more complete file is what allows for a more comprehensive downstream analysis of the obtained results. Finally the last output file is a simple .log of condensed information about the tool's execution, such as parameters set, features detected and execution time.

## 3.2 Chosen datasets

When presenting bioinformatics tools, it is important to chose appropriate datasets on which to test either their performance or their utility. In fact, one of the main concerns when analyzing large amounts of data, is how fast and efficiently said data can be elaborated, so when choosing a dataset to test a tool's performance, it is crucial to choose one of great size, in order to push its computational limits. This work needed to show the computational capabilities of both the single tool and the pipeline, as well as the applicability of isomiR-SEA, therefore, two meaningful yet sizable datasets were chosen: the raw reads used for the development of 7 species of MirGeneDB [26] and the astonishing 2398 raw reads files used for the creation of of the Human miROme from Patil et al. [49]. The table 3.1 compresses some information about the chosen datasts.

| Species (code) | Dataset origin | # Samples | Compressed Size |
|---|---|---:|---:|
| Worm (cel) | MirGeneDB | 8 | 6.2 GiB |
| Dog (cfa) | MirGeneDB | 20 | 12 GiB |
| Fruit fly (dme) | MirGeneDB | 25 | 17 GiB |
| Horse (eca) | MirGeneDB | 33 | 9.3 GiB |
| Human (hsa) | MirGeneDB | 81 | 20 GiB |
| Mouse (mmu) | MirGeneDB | 42 | 8,6 GiB |
| Rat (rno) | MirGeneDB | 18 | 1,6 GiB |
| Human (hsa) | miROme | 2398 | 757 GiB |

Table 3.1: General information about the analyzed datasets. It is important to note that the file size has been reported in its compressed format (.fastq.gz), so the actual size of the data is substantially bigger. It is easily noticeable how much bigger the miROme dataset (last row) is in comparison with the rest

### 3.2.1 MirGeneDB 3.0

Since miRNA and isomiR's role as phylogenetic markes is an important field that has a lot of potential for future insights and discoveries, choosing a variety of some of the most represented species was a natural choice for showcasing the applicability of isomiR-SEA. In its most recent publication [26], MirGeneDB has greatly amplified its metazoan repository, reaching a total of 114 species, and expanding upon the already present one.

From the paper, they provided a list of the NCBI Sequence Read Archive accessions in the form of supplementary table. These raw, unprocessed accession were thus used as input for the developed pipeline (which will be described shortly) to perform our analysis over a selection of 7 different species: *Homo Sapiens* (human), *Mus Musculus* (mouse), *Rattus Norvegicus* (rat), *Equus Caballus* (Horse), *Canis Familiaris* (dog), *Drosophila Melanogaster* (fruit fly) and *Caenorhabditis Elegans* (roundworm). This wide variety is meant to showcase the flexibility of isomiR-SEA and encourage its use for comparative isomiR expression studies of different species and tissues.

### 3.2.2 Human miROme

In 2022 Patil et al. [49] published as study revolving around the analysis of over 6000 human primary cell datasets downloaded from the *NCBI Sequence Read Archive*. The goal of this project was to provide the most complete reference of miRNA expression patterns by primary cell type, which was referred to as a human microRNAome (miROme). After an attentive selection, the total curated reads were restricted to 2077 samples, defining a total of 196 unique cell types. Given the importance, comprehensiveness and sheer size of this project, the human miROme was an outstanding candidate to test the performance of isomiR-SEA. By using it for this type of analysis, this work aims to showcase how, despite their size, meaningful datasets can be processed in acceptable amounts of time with reasonable resource usage.

## 3.3 Workflow management: Nextflow and nf-core

Nextflow [7] is a renowned open-source workflow management system for bioinformatics. It offers a multitude of automations to allow the user to run specialized tools for the analysis of biological data. In fact, it can automatically download work environments like conda and docker, with the required open-source tools, without explicitly needing to locally download them.

A standard Nextflow pipeline, at the highest level, is made of a main workflow that is tasked with calling the different components. Those can be either single modules or (optionally) a collection of them, elaborated by a subworkflow. Its main objective is therefore to ensure that the different parts communicate seamlessly with each other, making use of a data stream mechanism called channel that handles the input and output data of the different processes. Modules are the lowest level component of a Nextflow process chain. Those are the main work-force of the system, with each one defining a single process that executes a specialized task, built around either the selected tool or a custom script. Since many well known algorithms are usually executed together, it is common practice to group the modules that handle them into a unique subworkflow.

In terms of computational resources, Nextflow allows for a great control over both RAM and CPU utilization. A developer can specify the amount of memory or threads that a specific module can make use of, so that the developed pipeline does not over-allocate said resources. Moreover, Nextflow allows for implicit operation parallelism. In fact, when there are enough resources available and processes can be executed independently, Nextflow will try to run multiple tasks at the same time, resulting in greater computational efficiency and time save.

nf-core [50] is a community driven project aimed at providing it users with reliable open-source Nextflow pipelines, subworkflows and modules, in order to allow its users to either make use of a full pipeline, or to provide some already built components that can ease the process of creating a new pipeline. When building this pipeline, I adhered to the standard nf-core regulations in order to eventually be able to publish this work as an official nf-core pipeline.

## 3.4 The iSEA-TB pipeline

iSEA-TB has been developed with the purpose of seamlessly preprocess and align miRNA and isomiR starting from just a list of NCBI Sequence Read Archive accessions in .csv format. From that starting point, the workflow will generate the relevant metadata and download the raw compressed files. At this point, the pipeline will use a variety of tools to infer the relevant information about each individual file, in order to perform accurate adapter trimming and quality control. Finally the curated reads will be collapsed and sent to the alignment tool to produce the desired data. As an added feature, the pipeline workflow can be divided into three main sections and the user can decide to skip the undesired ones: download, preprocess and alignment, as shown in Figure 3.3.

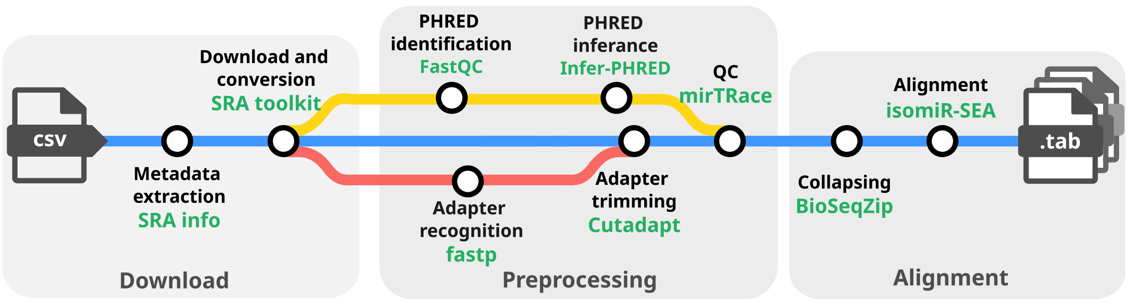


Figure 3.3: The iSEA-TB pipeline. Each step is part of a subsection of the entire workflow: Download, Preprocess and Alignment. The different operations in the Preprocess region are executed in parallel in order to provide both Cutadapt and miRTrace with the relevant details. The specifics of each tool is reported in Table 3.2

| Name | Version | Task | Source |
|---|---|---|---|
| SRA info | 1.0 | Metadata extraction | standalone script |
| SRA toolkit | 3.1.0 | Download and .fastq.gz conversion | nf-core subworkflow |
| FastQC | 0.12.1 | PHRED identification | nf-core module |
| Infer PHRED | 1.0 | PHRED inference | standalone script |
| fastp | 1.0.1 | Adapter recognition | nf-core module |
| Cutadapt | 5.0 | Adapter Trimming | nf-core module |
| miRTrace | 1.0.1 | Quality Control | nf-core module |
| BioSeqZip | 1.1 | Collapsing | local nf-core module |
| isomiR-SEA | 2.1 | Alignment | local nf-core module |

Table 3.2: General information on the modules used in the pipeline

### 3.4.1 SRA info

The first process that takes action is a self developed module that extracts the metadata required to perform the subsequent steps from the European Nucleotide Archive (ENA) website. By receiving as input a .csv file containing the desired NCBI SRA accessions (one per line), it calls a custom Phython script that fetches metadata by generating the appropriate web URL. Of the obtainable information, only the few needed by future operations are requested and saved. The script also generates a unique ID for the to-be-downloaded file combining both the run accession and the experiment accession. Finally, all the information regarding a single run are saved inside a .tsv file.

Those information will then be retrieved via the path provided as the module output, and processed inside the main workflow to generate the initial metadata channel. This step is important not only for the flawless execution of the successive modules, but also because a given accession can refer to more than a single file. For example, usually an accession that starts with `"PRJ"` refers to a project containing multiple experiments and runs.

### 3.4.2 SRA Toolkit

The publicly available nf-core subworkflow SRA Toolkit [51, 52] was used to download the requested accession from the NCBI database. By feeding it the appropriate information via the metadata channel, this subworkflow is capable of outputting all the related fastq files in a compressed format. This section is divided into two different sub-steps. The first one performs the prefetch command, which downloads the requested files into .sra format. The second part makes use of both the fasterq-dump command, which converts the obtained data into .fastq format, and the *pigz* command that compresses it into a .fastq.gz file

### 3.4.3 FastQC

The next process makes use of the nf-core module FastQC [53, 54]. The real purpose of this tool in my workflow is to utilize the provided output to automatically detect the Phred offset used to express the quality of the reads. This step was necessary since the tool tasked to perform quality control later in the pipeline can sometimes fail to automatically detect the PHRED offset, leading the pipeline to abruptly stop. However, the mentioned tool can accept this parameter as an argument, but since older reads can make use of a different format than the more recent ones, automatically identifying them with FastQC is a crucial step to ensure reliability.

### 3.4.4 Infer PHRED

The output from FastQC then gets sent to a small self developed module built around another python script. This process takes the textual output files provided as input and searches for the line that mentions which format has been detected for the PHRED offset. Depending on the detected name, a new file containing the sample ID of the accession and the offset in numerical format (either 33 or 64).

### 3.4.5 fastp

fastp [14, 55] is a commonly used tool for automatic adapter detection, consequential trimming and quality check. Although that's its primary use, much like FastQC, its role in the pipeline is slightly different: the workflow makes use of this tool's ability to detect an adapter and, if found, adds it at the top of an already provided file containing a multitude of well known adapters. This pre-existing .fasta list of adapters is a manually curated default list that covers a variety of adapter sequences, ensuring a reliable trimming procedure. However, since it is difficult to provide a list that covers every possible adapter, this specific step was added to cover the possibility of the adapter missing from the provided list.

The potential adapter is extracted by the workflow from the fastp report file and compared against the existing list to check if it is already present. If it is revealed to be missing, the newly found sequence is added to the top of the list.

### 3.4.6 Cutadapt

The actual tool used for adapter trimming is Cutadapt [15, 56], another popular tool specialized for the task. One of its main selling points for this project is its ability to check for the presence of a multitude of adapters and perform trimming via best match. This ability allows us to give its module the previously mentioned list in order for it to properly process the provided reads file. This step allows us to obtain the most reliable set of curated reads that can be processed by the alignment tool at the end of the workflow.

At this point, the workflow extracts the PHRED offset from the output files generated by the Infer PHRED module, and joins them with the Cutadapt output files via sample ID. Lastly, a new channel of temporary .tsv files gets generated, one for each sample, that are used as the input for the next module. Each file contains: the name of the input file, the sample ID, an empty string representing the adapter sequence (since we don't want to perform trimming again) and the PHRED offset.

### 3.4.7 miRTrace

During the research phase of this thesis, a tool that stood out was miRTrace [57, 58], a tool that specializes in quality control and taxonomic origin inference. Being an algorithm specific to sRNA, its performance tends to be more accurate for quality control than general-purpose tools, making it an especially compatible tool for the pipeline. The original algorithm can be executed in two different modes: quality control and taxonomic origin tracing. However, only the former was needed for this pipeline's workflow and, luckily, was already a public nf-core module. Its QC process starts by removing low-quality reads, checking their Phred score. After that, it can perform 3' adapter trimming via matching to a provided sequence. Then it checks for the presence of low complexity reads that either have highly repeating nucleotides or contain ambiguous ones like 'N'. Finally, by checking the remaining reads' length, short ones (with length lower than 18) are removed [59]. An optional step allows it to collapse the obtained reads when writing the output file, making it more space efficient. However, the header format used is unique to the tool, making it difficult for downstream tools to process. The input arguments allow for either single specifics regarding adapter and PHRED offset, or to provide an input .tsv file with the relevant information. This is where the files created at the end of the previous step get used. Since the pipeline parallelizes execution, each .tsv file containes information about a single input file, although the tool itself allows for the presence of one line per input file.

### 3.4.8 BioSeqZip

The next process that takes action revolves around a self-built BioSeqZip module, an optimized C++ tool for read collapsing. The employed version of this tool has been updated during this work to be able to correctly process the output files from miRTrace. Specifics about the changes will be discussed in Section 3.6. Since this is an algorithm especially compatible with our alignment tool of choice, to the point of being recommended for it, its integration in the pipeline was a natural choice. This algorithm can perform collapsing over a single file or a multitude of those, and it is optimized for minimum RAM usage. The collapsing happens after the input reads get sorted in alphabetical order, significantly speeding up research for identical reads. In case of exceptionally large files or a large number of files, it generates temporary output files in order to occupy the least possible amount of RAM. In case of multi-file input, an extra output .tab file gets generated in order to keep track of how many reads come from a specific input file. This kind of behavior allows for the alignment to correctly assign each potential output read to a different original file.

### 3.4.9 isomiR-SEA

Lastly, the developed isomiR-SEA module comes into play. By taking in input the processed and collapsed reads, it can efficiently and accurately align every read. To perform an accurate alignment procedure, it needs to also receive an appropriate set of input reference files. Although the tool can work with just a subset of input files, for a more comprehensive analysis, the pipeline exposes the full set of inputs: precursors, genomic coordinates, mature sequences and star sequences. Thanks to the most recent version of the tool, the pipeline is capable of efficiently analyzing the reads that reach this final step, allowing the user to generate a fully analyzable output. In fact, isomiR-SEA generates 2 different kinds of output: a simpler .gff file and a much more detailed .tab file that allows for a more comprehensive downstream analysis.

## 3.5 Pipeline manual

The pipeline is fairly easy to use, needing only a few input arguments and files, depending on the analysis that a user wishes to perform. As previously stated, its workflow can be divided into three main sections: download, preprocess and alignment. Although the pipeline can seamlessly perform every step from the beginning to the end, some users may want to perform only a subset of operations that suit their needs. The full analysis launch command looks something like this:

```
nextflow run path/to/pipeline/main -profile profile_1,profile_2
--input path/to/input.csv --outdir path/to/output_directory
--species hsa
```

The `-profile <profile_list>` argument is a standard argument for Nextflow pipelines. In this case it is advised to at least declare either `conda` or `docker` as a profile option for the pipeline to be able to automatically setup an environment/-container which will automatically retrieve the required tools without forcing the user to explicitly download each one of them. By official Nextflow design, more than one profile can be declared via comma separation.

The `--input <path/to/csv>` argument expects a .csv file containing NCBI Sequence Read Archive accessions, with one accession per line. Those are not restricted to runs, they can also be experiments or projects. The pipeline will take care of correctly retrieving the single runs related to provided input. The argument is used by the *SRA info* module to retrieve the relevant metadata for each sample file. There is a use case where this parameter becomes optional, which will be discussed below.

The `--outidr <path/to/out_dir>` argument is another standard Nextflow argument. As the name suggests, it requires the user to specify the directory in which they wish to save the outputs from the individual modules, including the final result of the workflow and the Nextflow execution report.

The `--species <ID>` argument is required for both miRTrace and isomiR-SEA. Those tools need to know the species they are dealing with for them to correctly execute their respective tasks. For this reason, the pipeline requires the input accessions to be related to a single species. The parameter should be provided in its canonical short form, like `hsa` for *Homo Sapiens* or `mmu` for *Mus Musculus.* There is a use case where this parameter becomes optional, which will be discussed below.

### 3.5.1 Alternative workflows

Depending on the user's needs, some more arguments become available and are used to specify to the pipeline which subsection of the workflow to perform. Some are flag arguments, meaning they will automatically be set to `true` when declared. Other arguments become mandatory depending on the provided flag.

The `--skip_download` flag allows the user to skip the SRA Tools related modules and allows the pipeline to begin its workflow at the preprocessing step, starting from the *SRA info* module and immediately jumping to the *FastQC* module execution. This parameter is very convenient for those users who are already in possession of the raw reads files from the NCBI Sequence Read Archive, since it allows them to skip a potentially time and space consuming section of the process.

The `--skip_preprocess` flag allows the user to skip the subsection of the workflow that starts from *FastQC* and ends with *miRTrace* (included). It is recommended to pair this flag argument with `--skip_download` as, in most cases, the download step will not be required either. This kind of behavior is convenient for users that only want to perform the alignment procedure on multiple already (perhaps manually) curated files.

The `--skip_alignment` flag allows the user to stop the workflow right before the execution of the *BioSeqZip* module, allowing users to obtain the fully curated unaligned reads, so that they can eventually be analyzed independently. This behavior is useful for users who want to manually check the curated reads before performing the alignment procedure, perhaps to check the correctness of the operations or to perform alignment against different databases. It can be paired with the `--skip_preprocess` flag to force the pipeline to only perform the download and conversion in .fastq.gz format of the raw reads. This behavior helps with storage management and allows users that are already in possession of a portion of the reads that will be processed.

The `--input_files <path/to/dir>` argument becomes mandatory when either (or both) `--skip_download` and `--skip_preprocess` are provided as input. It tells the pipeline where to find the initial files that will be used for the analysis. Depending on the starting point of the pipeline, this argument is used in case the user is already in possession of raw read files downloaded from the NCBI Sequence Read Archive in fastq.gz format (`--skip_download`), or is in possession of already curated reads (`--skip_preprocess`).

## 3.6 BioSeqZip and isomiR-SEA adaptation

Besides the development of custom Nextflow modules for BioSeqZip and isomiR-SEA, a few adaptations to their source code was required, as in their initial state neither of them was capable of interfacing with the peculiar output header format of the collapsed fastq files provided by miRTrace.

Usually, the .fasta or .fastq files that get used as input for those tools, simply contain a header and sequence pair for each read. This means that the number of input reads can be inferred by simply counting the number of times that the same sequence appears, so whatever info is usually saved in the header can be ignored. Although isomiR-SEA is able to read the output header format of BioSeqZip in order to extract the number of collapsed reads, since there is no standard convention, said format is different for all collapsing tools, meaning that this feature is only specific to this pair.

In fact, the BioSeqZip header looks like this:

```
>BIOSEQZIP|ID:0|CN:123456
```

where each field is separated by a `|` and the count of reads is `CN:123456`.

Instead, the miRTrace header has the following format:

```
>seq_1_x123456 rnatype:mirna
```

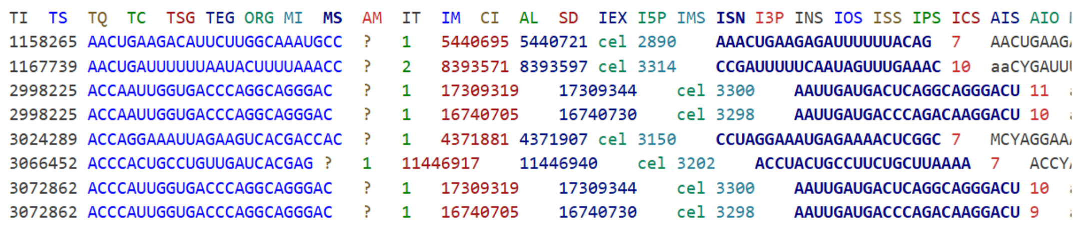
where the read counter `x123456` is embedded inside the first field, with a prepended `x`.

To tackle this challenge, the source code of both tools had to be adapted, so that either of them could individually interface with the mentioned data. By far, the hardest algorithm to adapt was BioSeqZip's. Its structure is very convoluted, many different part of the code communicate with each other in a very strict way, so both the extraction of header information and the optimal behavior to employ were challenging tasks. In terms of optimization, since the tool executes a sorting operation before collapsing for efficiency reasons, if left untouched, a lot of computational time would have been wasted in case the input was a single collapsed file from miRTrace. In fact, in this case, the tool only needed to convert the header format so that any downstream alignment tool that already uses BioSeqZip can extract the reads count. The use case of multiple input files needed to instead perform the operations regardless. In any case, an input flag was also added, since it is recommended that a user knows what type of input is being fed, however, BioSeqZip can automatically detect the miRTrace .fasta header format, in which case it will simply output a warning to inform the user, before proceeding with the correct operation regardless.

Since, as previously stated, isomiR-SEA already had a mechanism to extract information from the header, its implementation was a bit more straightforward, although it presented a couple of challenges of its own. The modifications of BioSeqZip allowed it to either reformat the individual input files, or collapse them into a single one that can then be fed directly to isomiR-SEA. By allowing such behavior, any workflow that either already makes use of BioSeqZip, or wants to do so in the future, can seamlessly interface with the data processed by miRTrace. If instead a user wants to directly feed miRTrace's output to isomiR-SEA, this recent adaptation of the program allows them to do so.

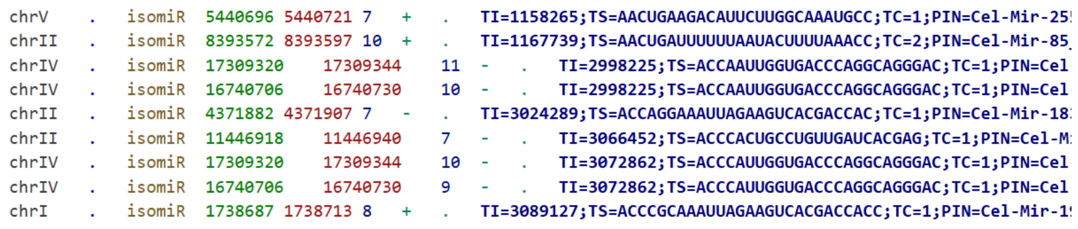
## 3.7 GUI and database analysis

The aligned data acquired at the end of the entire pipelined process can be difficult to analyzed if left in the simple textual format of the .tab , in Figure 3.4, or .gff file, in Figure 3.5. To make it human-readable and, therefore, usable for any sort of biological analysis, it was essential to convert it into a more user-friendly format.



```
TI   TS   TQ   TC   TSG TEG ORG MI   MS   AM   IT   IM   CI   AL   SD   IEX I5P IMS ISN I3P INS IOS ISS IPS ICS AIS AIO
1158265 AACUGAAGACAUUCUUGGCAAAUGCC  ?   1   5440695 5440721 cel 2890    AAACUGAAGAGAUUUUUUACAG  7   AACUGAAG
1167739 AACUGAUUUUUUAAUACUUUUAAACC  ?   2   8393571 8393597 cel 3314    CCGAUUUUUCAAUAGUUUGAAAC 10  aaCYGAUU
2998225 ACCAAUUGGUGACCCAGGCAGGGAC   ?   1   17309319    17309344    cel 3300    AAUUGAUGACUCAGGCAGGGACU 11
2998225 ACCAAUUGGUGACCCAGGCAGGGAC   ?   1   16740705    16740730    cel 3298    AAUUGAUGACCCAGACAAGGACU 10
3024289 ACCAGGAAAUUAGAAGUCACGACCAC  ?   1   4371881 4371907 cel 3150    CCUAGGAAAUGAGAAAACUCGGC 7   MCYAGGAA
3066452 ACCCACUGCCUGUUGAUCACGAG ?   1   11446917    11446940    cel 3202    ACCUACUGCCUUCUGCUUAAAA  7   ACCY
3072862 ACCCAUUGGUGACCCAGGCAGGGAC   ?   1   17309319    17309344    cel 3300    AAUUGAUGACUCAGGCAGGGACU 10
3072862 ACCCAUUGGUGACCCAGGCAGGGAC   ?   1   16740705    16740730    cel 3298    AAUUGAUGACCCAGACAAGGACU 9
```

Figure 3.4: Example of the .tab output format, seen via a VSCode extension that colors each column. Only a small portion of the total columns can fit in a single image.

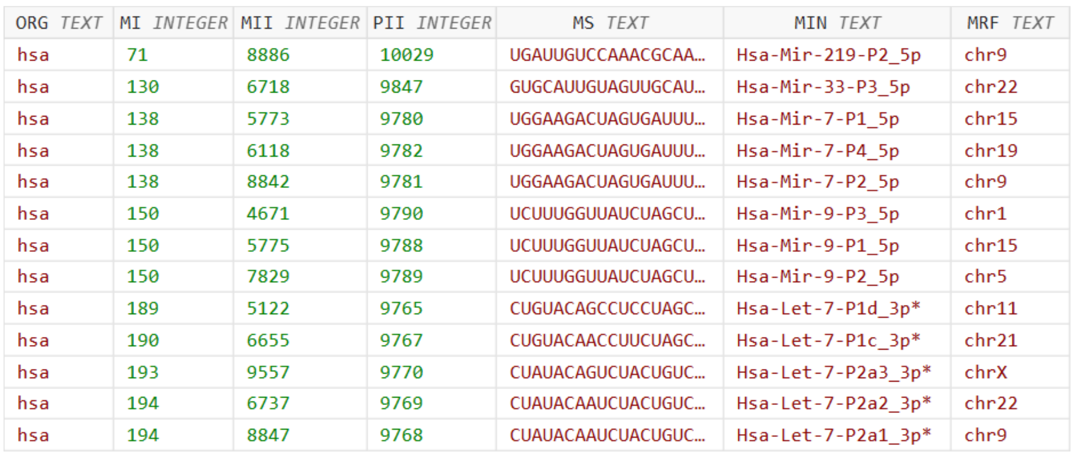


```
chrV    .   isomiR  5440696 5440721 7   +   .   TI=1158265;TS=AACUGAAGACAUUCUUGGCAAAUGCC;TC=1;PIN=Cel-Mir-25
chrII   .   isomiR  8393572 8393597 10  +   .   TI=1167739;TS=AACUGAUUUUUUAAUACUUUUAAACC;TC=2;PIN=Cel-Mir-85
chrIV   .   isomiR  17309320    17309344    11  -   .   TI=2998225;TS=ACCAAUUGGUGACCCAGGCAGGGAC;TC=1;PIN=Cel
chrIV   .   isomiR  16740706    16740730    10  -   .   TI=2998225;TS=ACCAAUUGGUGACCCAGGCAGGGAC;TC=1;PIN=Cel
chrII   .   isomiR  4371882 4371907 7   -   .   TI=3024289;TS=ACCAGGAAAUUAGAAGUCACGACCAC;TC=1;PIN=Cel-Mir-18
chrII   .   isomiR  11446918    11446940    7   -   .   TI=3066452;TS=ACCCACUGCCUGUUGAUCACGAG;TC=1;PIN=Cel-M
chrIV   .   isomiR  17309320    17309344    10  -   .   TI=3072862;TS=ACCCAUUGGUGACCCAGGCAGGGAC;TC=1;PIN=Cel
chrIV   .   isomiR  16740706    16740730    9   -   .   TI=3072862;TS=ACCCAUUGGUGACCCAGGCAGGGAC;TC=1;PIN=Cel
chrI    .   isomiR  1738687 1738713 8   +   .   TI=3089127;TS=ACCCGCAAAUUAGAAGUCACGACCACC;TC=1;PIN=Cel-Mir-1
```

Figure 3.5: Example of the .gff output format, seen via a VSCode extension that colors each column. A majority of information is compressed in the last column

### 3.7.1 Database construction

The first operation to perform was the creation of a comprehensive database that envelops all the information needed for downstream analysis. This was achieved with the use of two python scripts (in the form of Jupyter notebooks) that make use of the pandas dataframe library to extract each field from the .tab files generated as output by isomiR-SEA. Those scripts were previously developed when isomiR-SEA was updated to its newer version [6]. The first script is used to divide the reference miRNAs into three categories: unique mapped, multiple mapped and discarded, depending on the alignment score obtained by a read. The second one computes, for the non-discarded miRNAs, several statistics useful for the isomiR expression. A snippet of one of the resulting tables is shown in Figure 3.6. In its different tables, the dataset contains a variety of information about both the original samples and the analyzed sequences. Information about the miRs include, but are not limited to: the species, the miR name, the chromosome of origin and relative genomic coordinates, the precursor's name and sequence and the mir family. The specifics about the isomiRs contain fields relevant for the identification of the isoform, such as the number of added or removed nucleotides at each end, the number of mismatches, the number of mismatches in the seed specifically and more.



| ORG *TEXT* | MI *INTEGER* | MII *INTEGER* | PII *INTEGER* | MS *TEXT* | MIN *TEXT* | MRF *TEXT* |
|---|---|---|---|---|---|---|
| hsa | 71 | 8886 | 10029 | UGAUUGUCCAAACGCAA... | Hsa-Mir-219-P2_5p | chr9 |
| hsa | 130 | 6718 | 9847 | GUGCAUUGUAGUUGCAU... | Hsa-Mir-33-P3_5p | chr22 |
| hsa | 138 | 5773 | 9780 | UGGAAGACUAGUGAUUU... | Hsa-Mir-7-P1_5p | chr15 |
| hsa | 138 | 6118 | 9782 | UGGAAGACUAGUGAUUU... | Hsa-Mir-7-P4_5p | chr19 |
| hsa | 138 | 8842 | 9781 | UGGAAGACUAGUGAUUU... | Hsa-Mir-7-P2_5p | chr9 |
| hsa | 150 | 4671 | 9790 | UCUUUGGUUAUCUAGCU... | Hsa-Mir-9-P3_5p | chr1 |
| hsa | 150 | 5775 | 9788 | UCUUUGGUUAUCUAGCU... | Hsa-Mir-9-P1_5p | chr15 |
| hsa | 150 | 7829 | 9789 | UCUUUGGUUAUCUAGCU... | Hsa-Mir-9-P2_5p | chr5 |
| hsa | 189 | 5122 | 9765 | CUGUACAGCCUCCUAGC... | Hsa-Let-7-P1d_3p* | chr11 |
| hsa | 190 | 6655 | 9767 | CUGUACAACCUUCUAGC... | Hsa-Let-7-P1c_3p* | chr21 |
| hsa | 193 | 9557 | 9770 | CUAUACAGUCUACUGUC... | Hsa-Let-7-P2a3_3p* | chrX |
| hsa | 194 | 6737 | 9769 | CUAUACAAUCUACUGUC... | Hsa-Let-7-P2a2_3p* | chr22 |
| hsa | 194 | 8847 | 9768 | CUAUACAAUCUACUGUC... | Hsa-Let-7-P2a1_3p* | chr9 |

Figure 3.6: Example of the structure of one of the database's tables. Only a portion of the many available fields (columns) is visible

### 3.7.2 Interactive GUI in Grafana

The database was only the first step of the process. Although well re-organized, the data could not yet be defined as easily readable: it needed some form of digestible visualization for it to be able to provide some immediate information. Although a simple static bar chart could have been the simplest and more straightforward option, the actual use that could have come out of it would have been very limited, therefore a tool that could interactively re-elaborate a graphical representation of the data on-demand was needed. Thus, Grafana undoubtedly represented an excellent yet simple enough choice as a tool for the downstream analysis of the data. Grafana [8] is an open source multi platform web application for analytics and interactive visualization. Although often used for time series databases, it is a general tool for the construction of flexible graphs and charts from data sources. It has been used in combination with PostgreSQL [60] in order to generate the needed graphs from the developed database.

It is common practice to run PostgreSQL and Grafana in a shared Docker network. Although Docker isolates each service, they are both still able to communicate through a virtual network. In this environment, Postgres acts as the database management system, allowing Grafana to store both the database itself and the dashboards obtained via SQL query.

By using a python script that extracts the tables from the database of choice, saves them in a pandas dataframe, and uploads them to PostgreSQL via an established connection, it was possible to make full use of the Grafana web interface to analyze the provided data. The information could, at this point, be requested and re-elaborated via SQL queries and applied to any of the available Grafana visualizations. To check the signature of isomiRs and their expression levels, the stacked bar chart, seen in Figure 3.7, was the obvious choice. The read counts were first

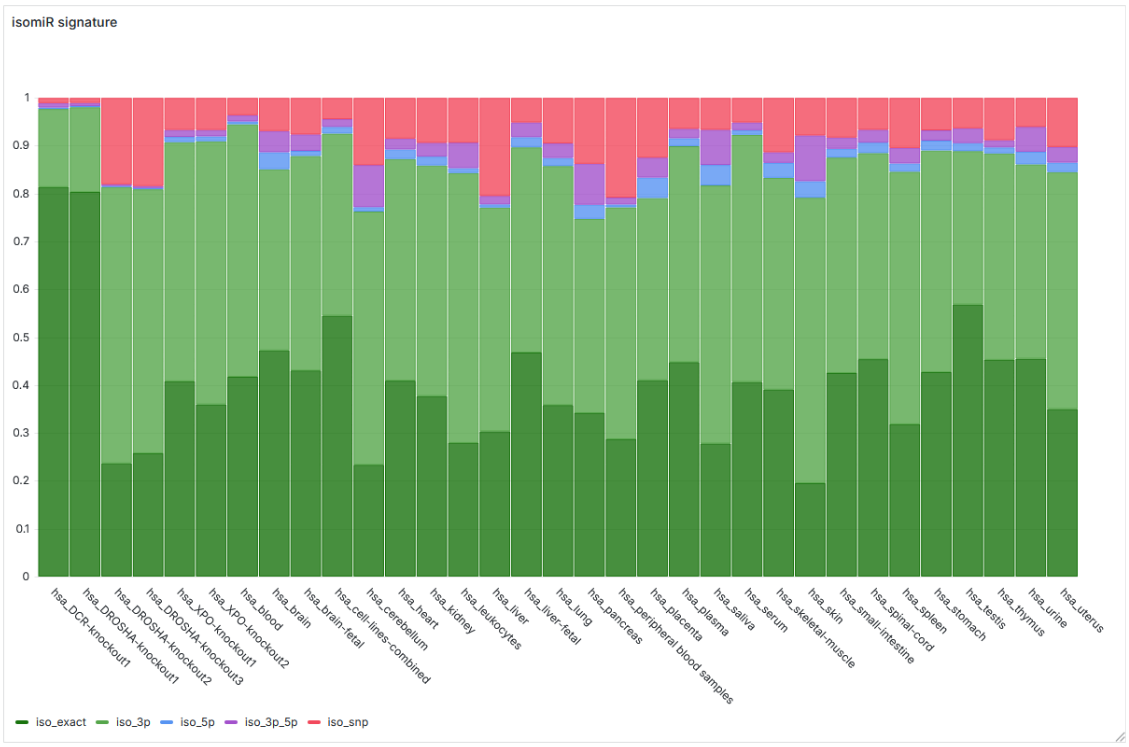


Figure 3.7: The expression level of human (hsa) isomiR for all available tissues. The isoform count was normalized by Read Per Million. Hovering over a bar or clicking it allows the user to see the exact RPM total for the selected isoform and tissue.

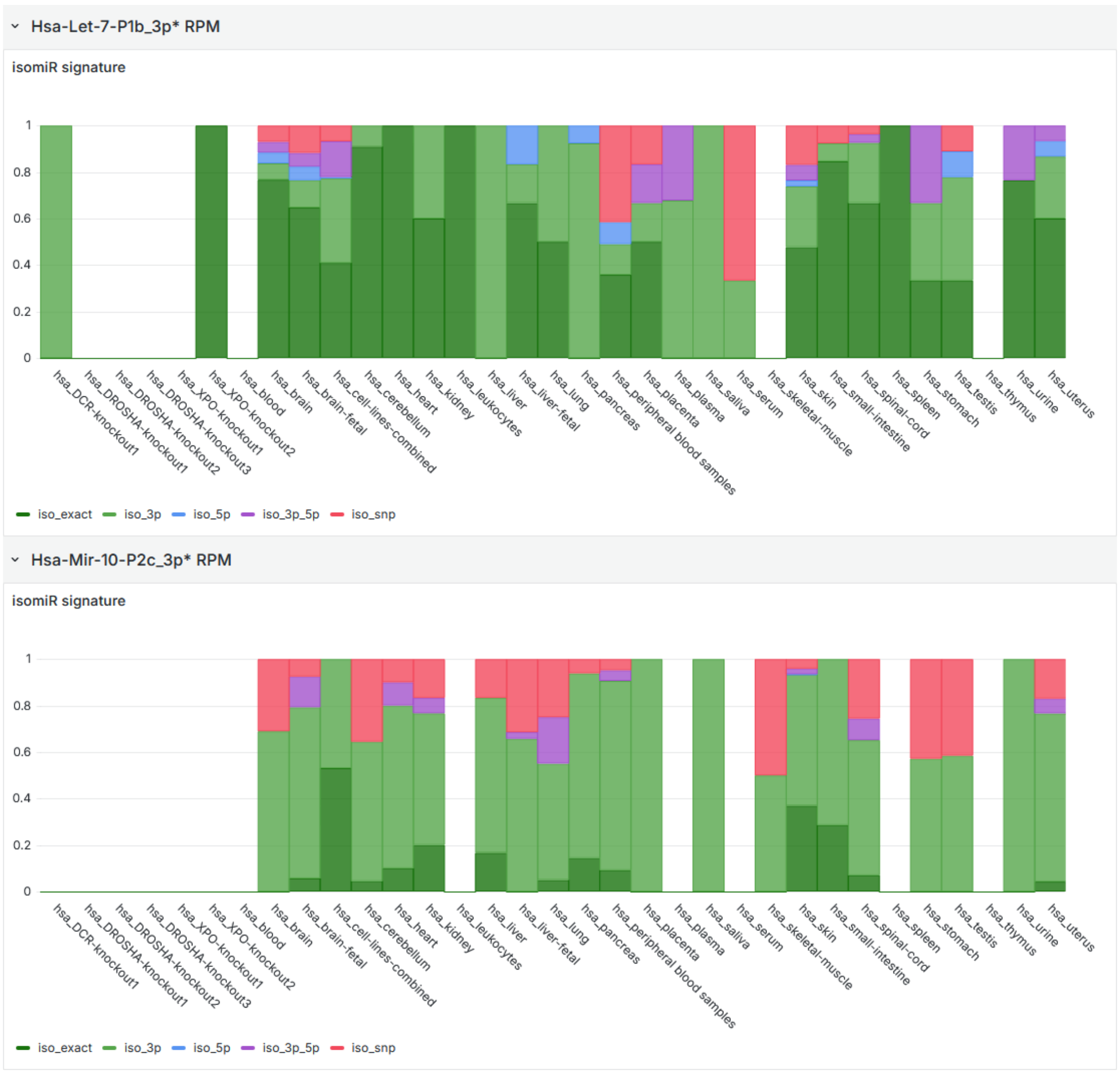


Figure 3.8: Expression level for Hsa-Let-7-P1b_3p* and Hsa-Mir-10-P2c_3p*. When multiple miRs are selected, Grafana allows a visual comparison between the individual plots.

normalized as reads per million for each sample (SRA run), then divided between isoforms and grouped by Tissue. This grouping was executed to mirror the covariance models available on the MirGeneDB website. The Dashboard has been set up to allow the user to select one or more species of their choice and do the same for the available miRNAs. This allows for both inter-species isoform expression-level comparison and intra-species, miRNA-specific, isoform expression-level comparison as shown in Figure 3.8. In fact, when selecting different miRNAs, the first panel shows the isomiR signatures for all the selected microRNAs in stacked bar chart form, while successive rows show their expression levels for single miRNAs. Of course, this interface is flexible, and any user who wishes to analyze their own data can do so by uploading their own database

### 3.7.3 Database access and plots generation

The following paragraphs give a brief description of the steps required to access the database and build the table required for the bar chart visualizations presented in section 3.7.2. These steps can be used as reference for users that whish to exploit the flexibility offered by Grafana's visualizations and powerful SQL access engine, allowing the creation of customized queries and views, for specific data sources and analysis goal.

#### Identification of all tissues

All tissues for the selected organism are recovered, and are used to generate bar placeholders for the bar chart plot. This ensures the bar chart shows has all tissues entries, even if no isomiR counts exist for a particular combination of tissue and miR. The query is shown in code 3.1.

Listing 3.1: Collect all tissues for the selected Organisms

```sql
WITH all_tissues AS (
   SELECT DISTINCT
      mir."ORG" AS org,
      sample."tissue"
   FROM "Sample" sample
   JOIN "Sample_MII_iso_inter" isomir
      ON sample."sample_isea_db_id" = isomir."sample_isea_db_id"
   JOIN "Sample_MII" mir
      ON isomir."MII" = mir."MII"
   WHERE mir."ORG" = ANY(ARRAY[$org]::varchar[])
),
```

#### Total counts per sample

For each sample, the total count of isomiRs is computed. This value is used as denominator when calculating the reads per million values. The query is shown in code 3.2.

Listing 3.2: Calculate total counts per sample

```sql
per_sample AS (
   SELECT
      sample."sample_isea_db_id",
      sample."tissue",
      mir."ORG" AS org,

        SUM(isomir."TC (Sum)") AS sample_total

    FROM "Sample_MII_iso_inter" AS isomir
    JOIN "Sample" AS sample ON isomir."sample_isea_db_id" =
        sample."sample_isea_db_id"
    JOIN "Sample_MII" AS mir ON isomir."MII" = mir."MII"
    GROUP BY sample."sample_isea_db_id", sample."tissue", mir."ORG"
),
```

### isomiR counts per sample

For each sample, the counts for each type of isomiR (exact, 3p, 5p, 3p-5p, snp) are calculated. The query is shown in code 3.3

Listing 3.3: Counts for each isomiR type

```
per_sample_iso AS (
    SELECT
        sample."sample_isea_db_id",
        sample."tissue",
        mir."ORG" AS org,

        SUM(CASE WHEN isomir."IEX" = 'T' THEN isomir."TC (Sum)" ELSE 0 END) AS
            iso_exact,
        SUM(CASE WHEN isomir."I3P" != 0 AND isomir."I5P" = 0 THEN isomir."TC
            (Sum)" ELSE 0 END) AS iso_3p,
        SUM(CASE WHEN isomir."I3P" = 0 AND isomir."I5P" != 0 THEN isomir."TC
            (Sum)" ELSE 0 END) AS iso_5p,
        SUM(CASE WHEN isomir."I3P" != 0 AND isomir."I5P" != 0 THEN isomir."TC
            (Sum)" ELSE 0 END) AS iso_3p_5p,
        SUM(CASE WHEN isomir."IMS" = 'T' OR isomir."ISN" = 'T' THEN isomir."TC
            (Sum)" ELSE 0 END) AS iso_snp

    FROM "Sample_MII_iso_inter" AS isomir
    JOIN "Sample" AS sample ON isomir."sample_isea_db_id" =
        sample."sample_isea_db_id"
    JOIN "Sample_MII" AS mir ON isomir."MII" = mir."MII"

    WHERE mir."MIN" = ANY(ARRAY[$mir]::varchar[])
      AND mir."ORG" = ANY(ARRAY[$org]::varchar[])

    GROUP BY sample."sample_isea_db_id", sample."tissue", mir."ORG"
),
```

### Reads per million

The isomiR counts per sample are converted into reads per million, using the previously calculated values. The query is shown in code 3.4

Listing 3.4: Calculate Reads per million

```sql
per_sample_rpm AS (
    SELECT
        i.org,
        i.tissue,
        i."sample_isea_db_id",

        (i.iso_exact / NULLIF(t.sample_total,0)) * 1e6 AS iso_exact,
        (i.iso_3p    / NULLIF(t.sample_total,0)) * 1e6 AS iso_3p,
        (i.iso_5p    / NULLIF(t.sample_total,0)) * 1e6 AS iso_5p,
        (i.iso_3p_5p / NULLIF(t.sample_total,0)) * 1e6 AS iso_3p_5p,
        (i.iso_snp   / NULLIF(t.sample_total,0)) * 1e6 AS iso_snp

    FROM per_sample_iso i
    JOIN per_sample t ON i."sample_isea_db_id" = t."sample_isea_db_id"
),
```

### isomiR levels at each tissue

The reads per million values calculated for each sample belonging to the same tissue are aggregated, to obtain the isomiR expression levels, divided by type, at tissue level. The query is shown in code 3.5

Listing 3.5: Calculate isomiR levels at tissue level

```sql
tissue_rpm AS (
    SELECT
        org,
        tissue,
        SUM(iso_exact) AS iso_exact,
        SUM(iso_3p)    AS iso_3p,
        SUM(iso_5p)    AS iso_5p,
        SUM(iso_3p_5p) AS iso_3p_5p,
        SUM(iso_snp)   AS iso_snp
    FROM per_sample_rpm
    GROUP BY org, tissue
)
```

### Data preparation for Grafana

Finally, all the identified tissues and the reads per million values calculated for each isomiR type at tissue level are merged into a single table to be consumed by Grafana and produce the bar chart plot.

Listing 3.6: Grafana data preparation

```sql
SELECT
    at.org || '_' || at.tissue AS tissue,
    COALESCE(tr.iso_exact, 0) AS iso_exact,
    COALESCE(tr.iso_3p, 0)  AS iso_3p,
    COALESCE(tr.iso_5p, 0)  AS iso_5p,
    COALESCE(tr.iso_3p_5p, 0) AS iso_3p_5p,
    COALESCE(tr.iso_snp, 0) AS iso_snp
FROM all_tissues at
LEFT JOIN tissue_rpm tr
       ON tr.org = at.org
      AND tr.tissue = at.tissue
ORDER BY at.org, at.tissue;
```

# Chapter 4

# Results

In this chapter, the results obtained during this analysis, as well as some observations, get shown. The first section compares the speed performance, RAM usage and other aspects of isomiR-SEA against that of some state-of-the-art tools, **miRGe3.0**, **isomiRMap**, and **sRNABench** from **sRNAToolbox**, showing that isomiR-SEA is the preferred choice for overall computational efficiency.

The second section exploits the accuracy of iSEA-TB's analysis to develop a database of isomiR expression levels between seven different species, analyzing the same accessions that MirGeneDB used for the development of their database. These results are the ones used with Grafana to develop human-readable interactive bar charts. They can be used for future comparative analysis between different species and inter-specie tissues.

The last section tests the ability of the pipeline to process a large and meaningful dataset of human reads. The main focus of this last analysis is the overall performance of the iSEA-TB pipeline as a whole.

## 4.1 Tools comparison

The performance of isomiR-SEA was compared against state of the art tools such as *miRge3.0*, *isoMiRmap* and *SRNAtoolbox*'s *SRNAbench*. Table 4.1 shows a compact comparison between tools where the speed was calculated relatively to sRNAbench. The comparison was executed using one of MirGeneDB's species reads, specifically Human, since it was sizeable enough to make any difference between tools noticeable enough. The analysis was performed on an 8-core virtual machine, provided with 16GB of RAM.

### 4.1.1 miRge3.0

miRge [61] is a python-based alignment tool was chosen because of its recency, with its third version being published in 2021, and because of the emphasis put into its

| Name | Execution time | Relative speed | Collapsing |
|---|---|---|---|
| **iSEA** | **5m 53s** | 208% | collapsed input |
| **iSEA+BioSZ** | **9m 24s** | 130% | executes collapsing |
| miRge3.0 | 11m 5s | 111% | executes collapsing |
| IsoMiRmap | 11m 46s | 104% | uncollapsed input |
| sRNAbench | 12m 15s | 100% | uncollapsed input |

Table 4.1: Comparison of alignment tools. For each tool, it has been specified wether their input was uncollapsed, collapsed or if the collapsing was performed at execution time. All speed performance was evaluated using sRNAbench as the base. iSEA was the best performer in all categories.

performance inside the paper that presented it. When upgraded to this version, the tool got turned from a simple alignment algorithm, to a full pipeline, which made it a closer competitor for iSEA-TB.

**Caveats:**
The first noticeable difference between the miRge3.0 pipeline and iSEA-TB resides in its flexibility. For example, while the latter can run independent sections, the former needs to run its own preprocessing step at least once. Although it can later run the alignment step independently with different parameters, this can only happen on data generated (and explicitly saved) by miRge itself in binary format, meaning that any curated file separated from the pipeline cannot be aligned with this tool. In fact, miRge does not allow any input to be in simple .fasta format. For this comparative analysis, this caveat meant that the input data going into the alignment part of the two competing pipelines was bound to be different, since the quality control and trimming procedures were performed in a different manner. Another flexibility issue comes from the necessity of explicitly providing an adapter argument to the tool in order to perform trimming (ie. `-a illumina`). This means that, although the execution of this step is faster in miRge, because only one kind of adapter gets searched, the tool has no form automation for the detection of different adapters in different samples, making its procedure more rigid.

Another problem that this pipeline seems to present comes from its use of RAM when inputting multiple input files. Since it seems like any calculation performed during the preprocess step is kept in memory, the more input data that the tool receives, the more RAM will progressively be used. The memory usage peaks during the collapsing procedure and, although it drops back to a more reasonable amount at the end, in later stages when a good portion of it is already occupied by the analysis of previous input files, the tool will saturate the RAM, leading to the operating system to kill the running process. Because of this, the 81 input files from the chosen database had to be separated into 4 different batches, which seemed to be the allowed minimum of data.

**Speed:**
Since this pipeline performs, among the other steps, its own collapsing, the execution-time report is based on already collapsed data. Thus, to even out the ground, the computational comparison against isomiR-SEA will be made against files already collapsed by BioSeqZip. While miRge's alignment step was resolved after a total of 11 minutes and 5 seconds, isomiR-SEA managed to complete its full analysis in just 5 minutes 53 seconds, resulting in about a 88% increase in performance from isomiR-SEA for a 47% time-save.

### 4.1.2 IsoMiRmap

Published in 2021, IsoMiRmap [62] is another python-based alignment algorithm that was presented as a fast, reliable tool for isomiR detection and characterization. Its speed was reported to be of "10 million sequenced reads in under 55 seconds", with time scaling linearly with input size. Once again, both recency and statements about the speed were a deciding factor for this competitor.

**Caveats:**
The first inconvenience to take note of is that there seems to be no way to provide multiple input files for a single run. The program expects only a single input file, meaning that when working with bigger datasets, composed of multiple samples, the analysis can get tedious. During this study, this problem was solved via the use of a script that launched the execution command multiple times, once for each input file. However, this behavior means that the program has to re-read the reference files to perform alignment each time the execution starts, adding static processing time and hindering the execution performance for larger amounts of files.

Another problem comes from the need for the program to use a reference isomiR file. This means that the tool can only report isoforms that are present in the reference file. Furthermore, although the *GitHub* repository comes with a few pre-computed mapping bundles containing reference data for the homo sapiens species, the reference databases used only come from miRBase and miRCarta (another public database), with the implications about potential unreliability of non-manually curated databases that were discussed in previous sections.

Lastly, the naming scheme used for isomiRs is very simplistic. Although the output reports some information about the analyzed sequence, the name itself does not provide any info about the canonical counterpart, making it more difficult to identify.

**Speed:**
At the end of the execution loop, the script reported the total execution time:

11 min 46 seconds. This puts this tool at about the same level of performance with miRge3.0, meaning that, in terms of alignment, the difference between it and isomiR-SEA is about the same, with the latter, of course, being the top performer. This, however, does not account for the fact that the input provided to IsoMiRmap was not composed of collapsed reads. For the sake of fairness, the execution time of 3 minutes 31 seconds of BioSeqZip will also be added to isomiR-SEA's, reaching a total of 9 minutes 24 seconds, making isomiR-SEA about 25% faster, with a time save of around 20%. In this case, it is important to note that the execution of BioSeqZip brings more noticeable benefits with respect to the size of the input dataset, like in the case of the human miROme dataset.

### 4.1.3 sRNAbench

First Published in 2014, java developed sRNAbench is the longest living tool in this list. Despite its age, it received multiple updates over the course of the year, becoming the starting point for the development of the famous sRNAtoolbox in 2015 of which the latest update came in 2022 [63]. Over the course of the updates, the tool has been improved and refined, and the entirety of the toolbox that was built around it has become the go-to tool for miRNA analysis. Naturally, this makes sRNAbench a prime candidate for the comparison with isomiR-SEA.

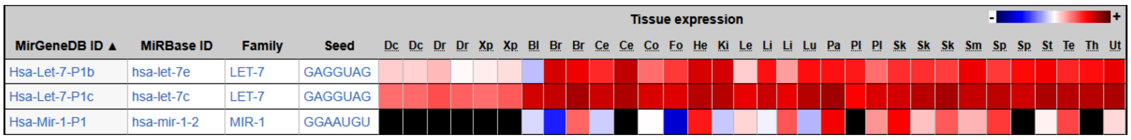
**Caveats:**
sRNAtoolbox and the tools provided within can be run either online, as a standalone version, or inside a pre-built Docker container. For the sake of a fair comparison, sRNAbench was run from inside this last option, in order to provide it with the same hardware capabilities. Once again, the tool by itself does not allow multiple input files, so it had to be run using a bash script that also calculated its execution time, which seems to not be given by default.

**Speed:**
The reported execution time at the end of the bash script was of 12 minutes 15 seconds, which is the slowest in this list of competitors by just a few seconds. This is to be expected, as over the years, sRNAbench has expanded its functionality to also perform trimming and quality control. Although no parameters have been provided, this part of the workflow has probably made a small impact on its performance. Once again, since uncollapsed files were given as input, its performance is evaluated against that of both BioSeqZip and isomiR-SEA, which amounts to 9 minutes 24 seconds. This means that this couple is faster by about 30.3%, bringing a total time save of 23.3%. The same observations about the dataset size in regards to the use of BioSeqZip apply here.

## 4.2 MirGeneDB dataset

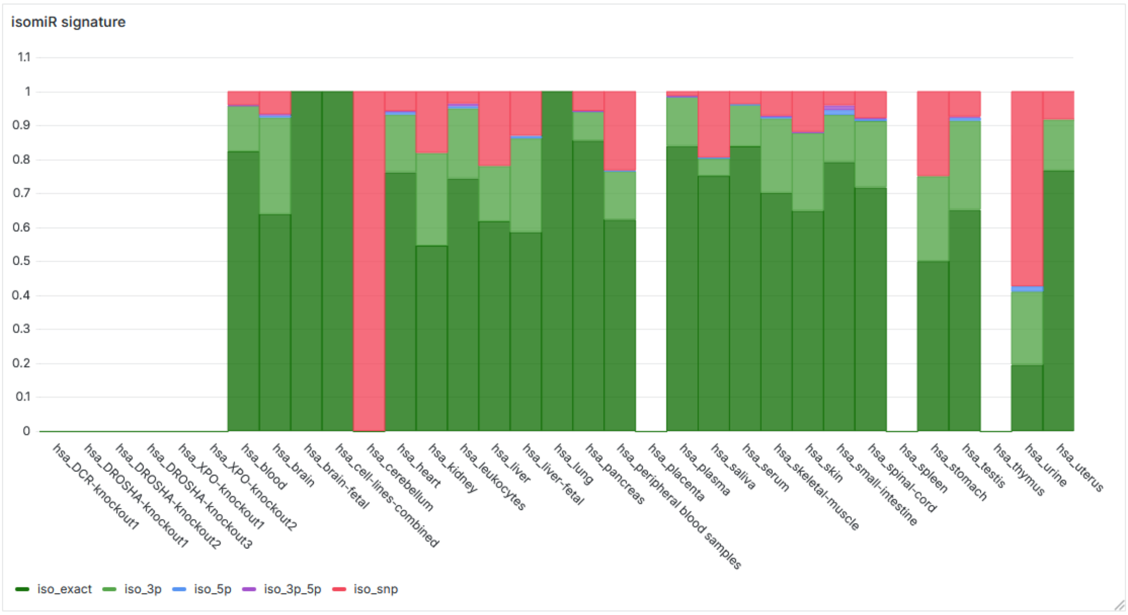
The dataset of raw reads used for the development of MirGeneDB3.0 represented a meaningful dataset to perform an isomiR analysis. In fact, the obtained data can be used to deepen our knowledge regarding isomiRs, their expression levels in different species and tissues, and potentially inspire future uses based on them.

Wherever a miRNA was absent for a tissue in MirGeneDB, it was also either absent or solely represented as an isomiR in this analysis, meaning that the results tend to be mostly free of false positives, as can be seen in Figure 4.1. However, comparing some of the obtained results against the reports found on the MirGeneDB website, lowly expressed canonical miRNAs sometimes did not get detected by this analysis. This result may depend on the quality control being too strict or perhaps on the trimming performed with too much flexibility, both resulting in the deletion of potential sequences.



| | | | | Tissue expression | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MirGeneDB ID ▲ | MiRBase ID | Family | Seed | Dc | Dc | Dr | Dr | Xp | Xp | Bl | Br | Br | Ce | Ce | Co | Fo | He | Ki | Le | Li | Li | Lu | Pa | Pl | Pl | Sk | Sk | Sk | Sm | Sp | Sp | St | Te | Th | Ut |
| Hsa-Let-7-P1b | hsa-let-7e | LET-7 | GAGGUAG | 🟥 | 🟥 | 🟥 | ⬜ | 🟥 | 🟥 | 🟦 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 |
| Hsa-Let-7-P1c | hsa-let-7c | LET-7 | GAGGUAG | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 |
| Hsa-Mir-1-P1 | hsa-mir-1-2 | MIR-1 | GGAAUGU | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | ⬛ | 🟦 | 🟦 | 🟥 | 🟦 | ⬛ | ⬜ | 🟦 | 🟥 | 🟦 | 🟥 | ⬜ | 🟥 | 🟦 | 🟥 | ⬛ | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | ⬛ | ⬜ | 🟥 | ⬛ | 🟥 |

(A) Compressed version of the MirGeneDB browsable table. The black squares represent absence of mir in the tissue, as can be see in the last row with Hsa-mir-1-P1.



(B) Visualization of the Hsa-mir-1-P1_3p in Grafana. Here the empty bars are on the same tissues with respect to the MirGeneDB website.

Figure 4.1: Comparison between MirGeneDB and Grafana visualization for Hsa-mir-1.

The isomiR expression levels visualized via the Grafana stacked bar chart are within expectations, with the majority of the reported forms either being the exact canonical microRNA or the 3p isoform, as can be seen in Figure 4.2. In fact, literature on the topic has always reported that the 3p isoform is the most common among the possible isomiRs, sometimes overgrowing the canonical counterpart's expression level. It is important to note that, by nature, these isoforms are also the most likely to be false positives and that a manual procedure must be performed by an expert in the field in order to prove which ones can be considered bona-fide.

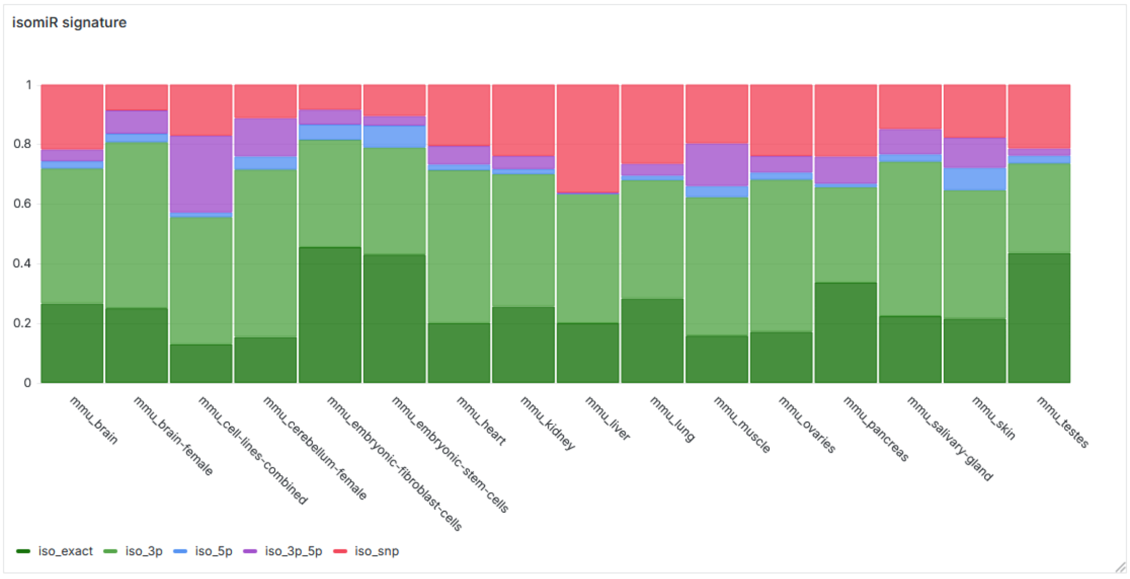


Figure 4.2: Visualization of isomiR expression levels for mmu.

Lastly, the performance of the iSEA-TB pipeline was more than satisfactory. The human dataset, being the bigger among the 7 species with 81 input files, took 1 hour 22 minutes 28 seconds for the entire preprocessing, while the collapsing and alignment procedure took 10 minutes and 27 seconds, accounting also for the pipeline initialization and conclusion steps, for a total of 1 hour 32 minutes and 49 seconds. We want to bring attention to the fact that the preprocessing procedure takes longer than a user might expect, because of the extra steps performed by different tools in order to ensure the correctness of this procedures. As explained in previous sections, those tools do not uniquely perform the steps that they are used for in iSEA-TB, but are accurate nonetheless, therefore their execution requires some extra time in exchange for reliability.

## 4.3 Human miROme dataset

The raw reads used to sequence the human miROme [49] are both abundant enough to form a dataset of impressive size and are meaningful enough to be studied. Even when compressed in .fastq.gz format, the raw reads occupy a massive amount of storage space, reaching about 757 GB, more than some current-day laptops are capable of storing.

The use of the obtained 2398 accessions allowed us to achieve an accurate analysis of the performance of both the entire pipeline and of the BioSeqZip and isomiR-SEA pair. For convenience reasons, the preprocessing step and the alignment step were executed separately. The preprocessing step was the most time-consuming, needing a total of 1 day 21 hours 57 minutes, and 7 seconds to be completed.

### 4.3.1 Preprocessing

The Nextflow execution reports helped with the analysis of the resources used, as see in Figures 4.3, 4.4, 4.5, showing that the tool requiring the most CPU power

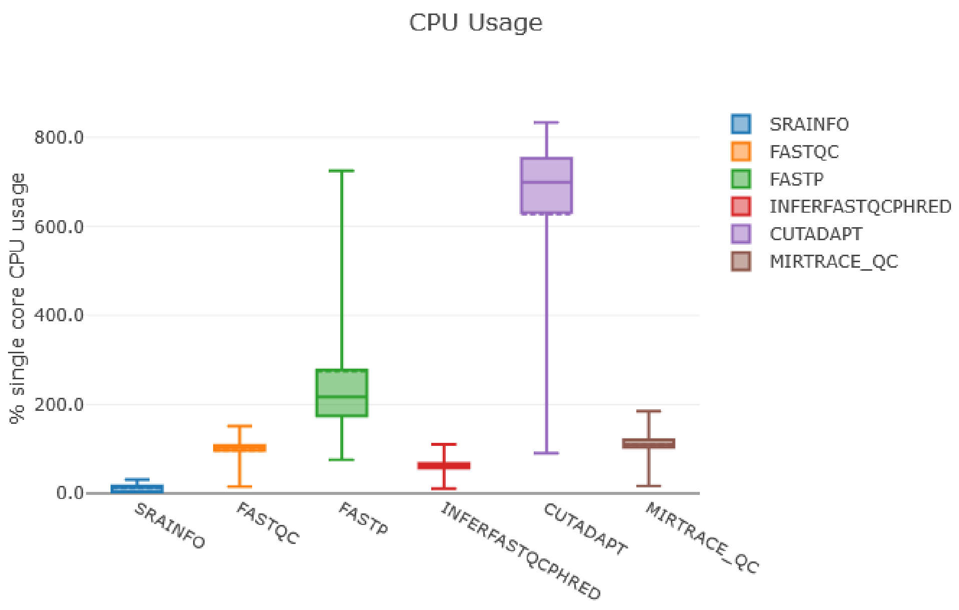


Figure 4.3: Nextflow execution report's CPU Resource graph. For each tool, the filled-in box represents the average CPU consumption, while the top and bottom lines represent the maximum and minimum CPU usage, which represent outliers or extreme cases, depending on the distance from the box. On average, Cutadapt requires the most amount of computational power among all the tools, with fastp coming to a close second place for larger files.

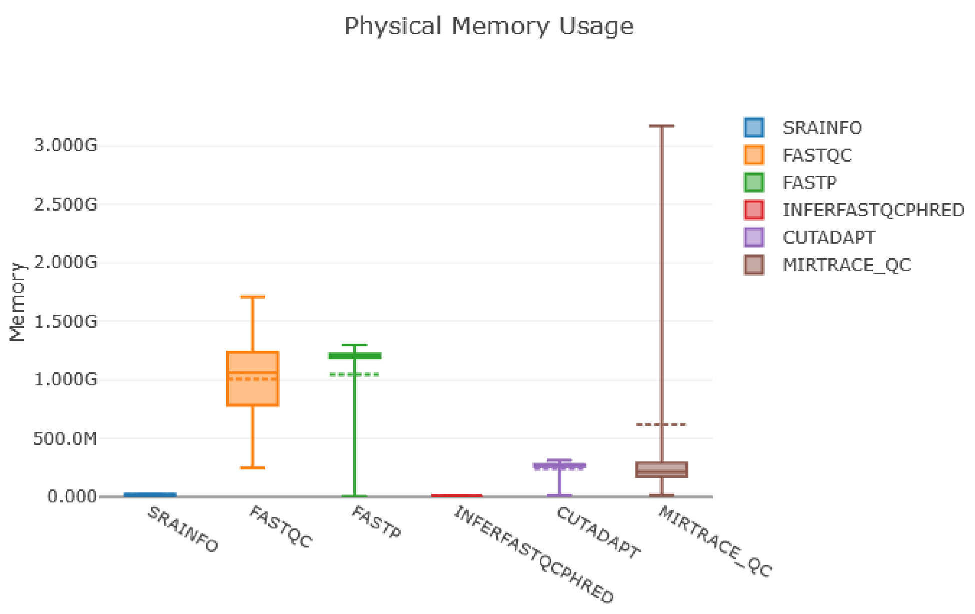


Figure 4.4: Nextflow execution report's RAM Resource graph. For each tool, the filled-in box represents the average RAM consumption, while the top and bottom lines represent the maximum and minimum RAM usage, which represent outliers or extreme cases, depending on the distance from the box. The RAM requirements tend to be very contained, where even bigger-sized outliers require less than 2 GB of RAM. Here is noticeable how, although miRTrace tends to be very efficient with average sized input, it can start requiring more RAM with larger input files (top bar). However, the total memory usage remains contained under 4GB.

was Cutadapt by a large margin. This is to be expected because the file containing the possible adapters provided to this module contains a multitude of sequences, a challenge that Cutadapt tackles by exploiting multi-threading. For larger files, Fastp can also require a greater amount of CPU to perform its analysis, while the other tools remain within acceptable ranges.

In terms of RAM, none of the tools have particularly high needs, with all of them requiring less than 2GB of memory. The only outlier is mirTRace that, when dealing with particularly sizeable files, tends to require more physical memory. However, even in those more extreme cases, the total need of RAM remains under 4GB, which means that, provided with enough storage space, even low-end devices can comfortably perform this kind of analysis.

Regarding execution time, all tools are efficient enough to require, on average, less than 2 minutes to analyze a single sample. Of course, as can be seen, when input files are of greater size, the execution may require longer than 6 minutes, with Cutadapt needing over 10 minutes for the largest file provided in this set. Since

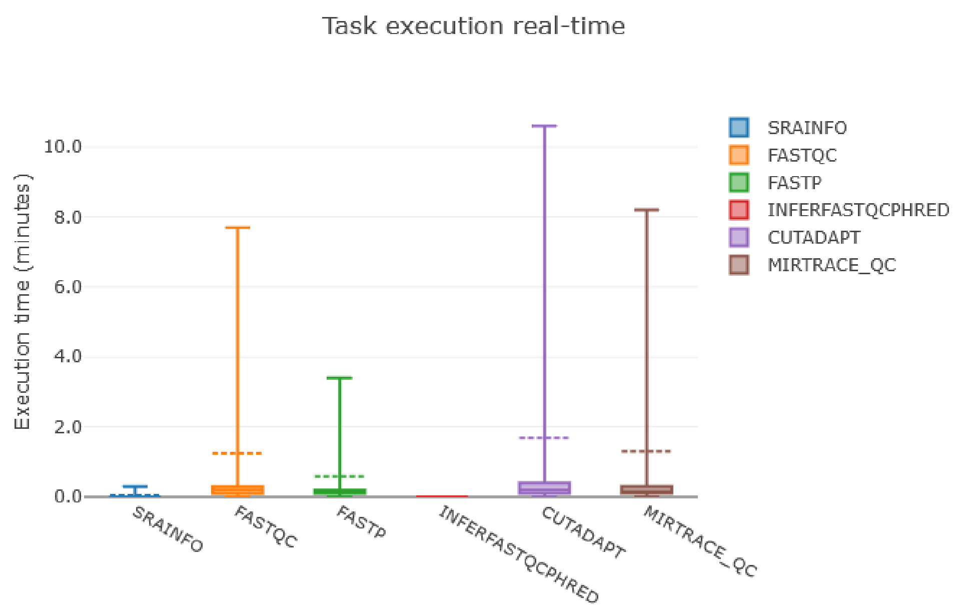


Figure 4.5: Nextflow execution report's Job Duration Resource graph. Although the execution has a low average and an even lower median, larger files may require a more noticeable amount of time to be analyzed.

both SRA-info and Infer-PHRED are, at their core, simple Python scripts that perform a straight forward job, they are the ones that require the least amount of time to be executed, independently of the input files.

### 4.3.2 Alignment

After the whole preprocessing was completed, we executed the alignment procedure. Unfortunately, at test time, a bug within the current version of BioSeqZip was discovered that made the program crash at the end of the collapsing procedure when too many files were given, independently of RAM saturation. To work around this current issue, the obtained curated reads were given to the program in batches of 49 files willingly unbalanced in size to retain variability. Once again, Nextflow provided the graphical representations of the tool's performances used for this analysis, as can be seen in Figures 4.6, 4.7, 4.8. As for CPU usage, it is within expectation that isomiR-SEA made use of a good portion of the available CPU, since it exploits concurrent execution. In terms of RAM, both isomiR-SEA and BioSeZip remained within low memory usage across the majority of the batches, with a few outliers exceeding 10GB whilst still remaining well below the provided amount. Finally in terms of time, only the biggest among batches took over 8 minutes for BioSeqZip and over 10 minutes for isomiR-Sea. The majority stayed within ranges of 2 to 4 minutes for BioSeqZip and 2 to 6 minutes for isomiR-SEA, which are incredibly good times given the sizes of the batches. The total execution time, including pipeline specific procedures, was of 5 hours 44 minutes 30 seconds. It is important to note that if BioSeqZip were able to collapse the entirety of the dataset, the entire process would have taken much less time as, with the used method, isomiR-SEA was forced to elaborate redundant reads multiple times.

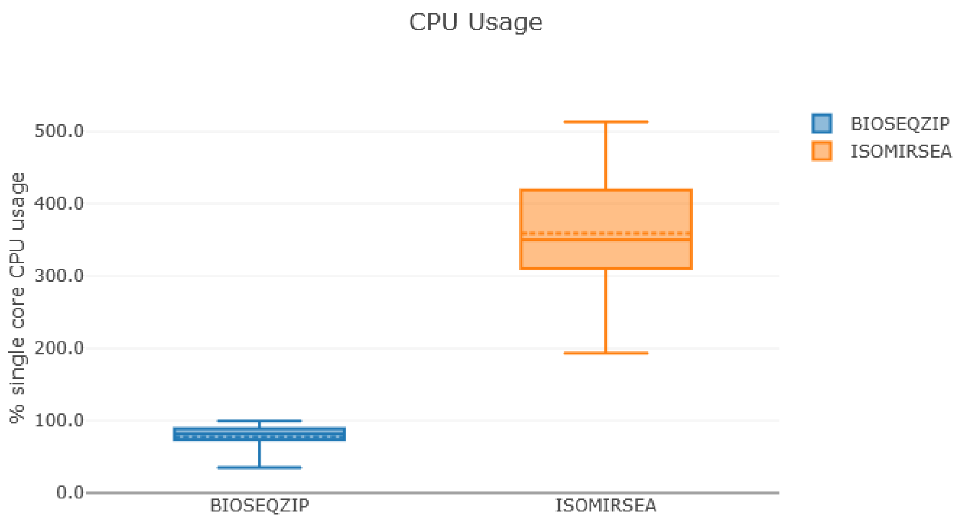


Figure 4.6: Nextflow execution report's CPU Resource graph. For each tool, the filled-in box represents the average CPU consumption, while the top and bottom lines represent the maximum and minimum CPU usage, which represent outliers or extreme cases, depending on the distance from the box.

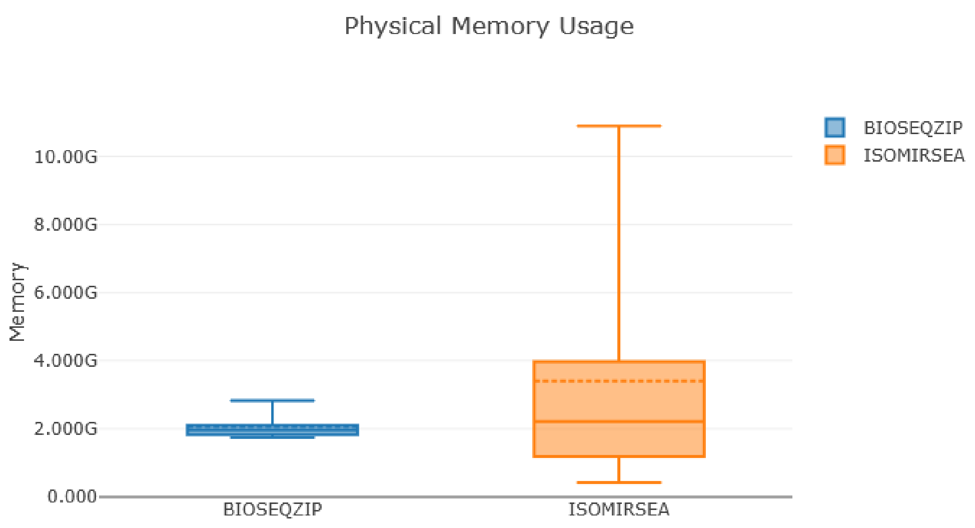


Figure 4.7: Nextflow execution report's RAM Resource graph. For each tool, the filled-in box represents the average RAM consumption, while the top and bottom lines represent the maximum and minimum RAM usage, which represent outliers or extreme cases, depending on the distance from the box.

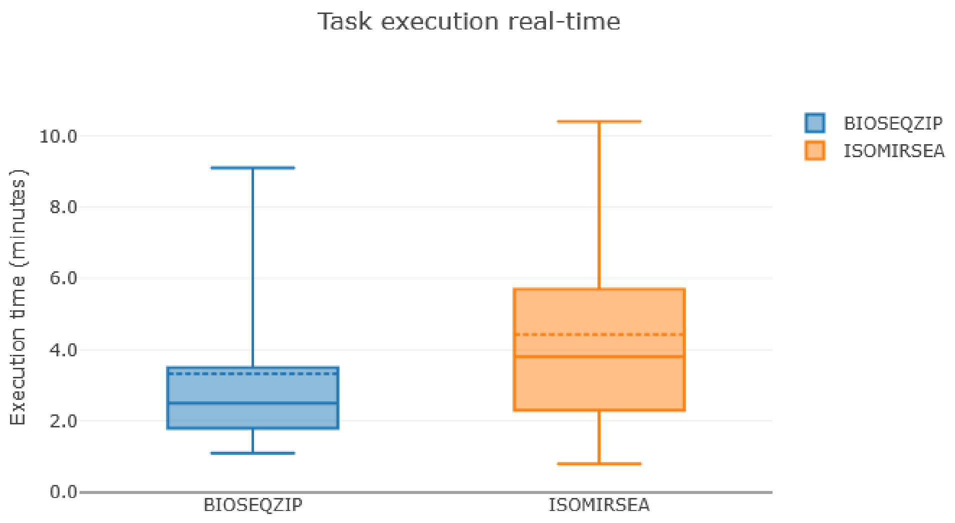


Figure 4.8: Nextflow execution report's Job Duration Resource graph.

# Chapter 5

# Conclusion

Although over the last decades much progress has been made, microRNAs and especially isomiRs still remain a complicated subject to study. There are many intricacies to consider, especially when dealing with the preprocessing of raw miRNA reads and adapter trimming, and different studies apply different methods to deal with them. When trying to analyze this kind of data, pipelines represent a solid solution in terms of consistency and reproducibility. With a properly refined workflow, a pipeline can precisely deal with a variety of data, ensuring consistency in the obtained results. Another problem that this type of research faces comes from computational resources. When faced with multiple large input files, some devices either struggle to keep low RAM usage or simply do not give the option for multiple inputs. Many tools also struggle with completing their entire execution within acceptable times. Finally, another challenge comes from the readability of obtained results. In fact, when analyzing large amounts of processed data, it is crucial to ease the process of making it as human-readable as possible. Visualization thus represents a crucial factor in downstream analysis, and when trying to promote further studies.

This thesis presented a solution to overcome the aforementioned challenges by implementing a Nextflow pipeline using isomiR-SEA for alignment, and by providing an user-friendly dashboard for downstream analysis. The work can be extended in several directions: more data can be analyzed, particularly the full metazoan tree from MirGeneDB; the obtained isomiR expression levels can be integrated directly into the MirGeneDB website to aid the relevant studies, after checking all the discrepancies between the obtained results and what is currently published; lastly the process of elaborating the database and generating the Grafana bar charts can be integrated within the developed pipeline.

In its current state, the iSEA-TB pipeline is a fast, accurate and reliable pipeline, capable of both analyzing massive datasets and produce meaningful results. The pipeline is centered around isomiR-SEA, which is a particularly fast tool for the analysis of miRNA reads. isomiR-SEA relies on the identification and alignment of the seed region, which is especially important for the miRNA:mRNA interaction sites. Great care was taken to ensure that the preprocessing was robust, to the point of using complex tools to extract the relevant information for the downstream tools. Of particular interest were the use of FastQC for correct PHRED offset identification and fastp to ensure that the correct adapter was provided to Cutadapt to perform 3' adapter trimming. The database developed with the obtained results allowed the generation of very intuitive and interactive Grafana bar charts that are user-friendly and human-readable. The dashboard is flexible enough to allow different configurations and can be provided with a user's own database to perform their own analysis.

This thesis' work will help future research in microRNAs and isomiRs expression levels, which has great potential for growth and discovery. Both medical studies revolving around the use of microRNAs as biomarkers for medical purposes, as well as phylogenetic studies are sure to benefit from this work by broadening their research to include isomiRs expression levels.

# Bibliography

[1] Thomas X Lu and Marc E Rothenberg. «MicroRNA». In: *Journal of allergy and clinical immunology* 141.4 (2018), pp. 1202–1207.

[2] Bastian Fromm. «A renaissance of microRNAs as taxonomic and phylogenetic markers in animals». In: *Zoologica Scripta* 53.6 (2024), pp. 754–762.

[3] Corine T Neilsen, Gregory J Goodall, and Cameron P Bracken. «IsomiRs–the overlooked repertoire in the dynamic microRNAome». In: *Trends in genetics* 28.11 (2012), pp. 544–549.

[4] Li Guo and Feng Chen. «A challenge for miRNA: multiple isomiRs in miRNAomics». In: *Gene* 544.1 (2014), pp. 1–7.

[5] Gianvito Urgese et al. «isomiR-SEA: an RNA-Seq analysis tool for miRNAs/isomiRs expression level profiling and miRNA-mRNA interaction sites evaluation». In: *BMC Bioinformatics* 17.1 (2016), p. 148. DOI: 10.1186/s12859-016-0958-0. URL: https://doi.org/10.1186/s12859-016-0958-0.

[6] Marco Capettini. «A modern reimplementation of an alignment pipeline for the analysis and quantification of small non-coding RNA and isoforms using C++ and Python». Mar. 2020. URL: http://webthesis.biblio.polito.it/14493/.

[7] Paolo Di Tommaso et al. «Nextflow enables reproducible computational workflows». In: *Nature biotechnology* 35.4 (2017), pp. 316–319.

[8] Grafana Labs. *Grafana: Open-source analytics and monitoring platform.* https://grafana.com/. Accessed: 26 November 2025.

[9] David P Clark and Nanette J Pazdernik. *Molecular biology.* Elsevier, 2012.

[10] Anna Majer, Kyle Caligiuri, and Stephanie Booth. «A User-Friendly Computational Work fl ow for the Analysis of MicroRNA Deep Sequencing Data». In: vol. 936. Jan. 2013, pp. 35–45. ISBN: 978-1-62703-082-3. DOI: 10.1007/978-1-62703-083-0_3.

[11] Xiangfu Zhong et al. «Accurate Adapter Information Is Crucial for Reproducibility and Reusability in Small RNA Seq Studies». In: *Non-Coding RNA* 5.4 (2019). ISSN: 2311-553X. DOI: 10.3390/ncrna5040049. URL: https://www.mdpi.com/2311-553X/5/4/49.

[12] Hongshan Jiang et al. «Skewer: a fast and accurate adapter trimmer for next-generation sequencing paired-end reads». In: *BMC Bioinformatics* 15.1 (2014), p. 182. DOI: 10.1186/1471-2105-15-182. URL: https://link.springer.com/article/10.1186/1471-2105-15-182.

[13] Lodoe Lama et al. «Small RNA-seq: The RNA 5′-end adapter ligation problem and how to circumvent it». In: *Journal of Biological Methods* 6.1 (2019), e108. DOI: 10.14440/jbm.2019.269. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC6507418/.

[14] Shifu Chen et al. «fastp: an ultra-fast all-in-one FASTQ preprocessor». In: *Bioinformatics* 34.17 (Sept. 2018), pp. i884–i890. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty560. eprint: https://academic.oup.com/bioinformatics/article-pdf/34/17/i884/50582648/bioinformatics_34_17_i884.pdf. URL: https://doi.org/10.1093/bioinformatics/bty560.

[15] Marcel Martin. «Cutadapt removes adapter sequences from high-throughput sequencing reads». In: *EMBnet.journal* 17.1 (2011), pp. 10–12. ISSN: 2226-6089. DOI: 10.14806/ej.17.1.200. URL: https://journal.embnet.org/index.php/embnetjournal/article/view/200.

[16] Pouya Goleij et al. «Chapter Three - Types of RNA therapeutics». In: *RNA Therapeutics Part A*. Ed. by Dinh-Toi Chu and Van Thai Than. Vol. 203. Progress in Molecular Biology and Translational Science. Academic Press, 2024, pp. 41–63. DOI: https://doi.org/10.1016/bs.pmbts.2023.12.022. URL: https://www.sciencedirect.com/science/article/pii/S1877117323002041.

[17] S. D. Chandradoss et al. «Reexamining assumptions about miRNA-guided gene silencing». In: *Nucleic Acids Research* 50.2 (2022), pp. 617–634. DOI: 10.1093/nar/gkab1256. URL: https://doi.org/10.1093/nar/gkab1256.

[18] Jacob O'Brien et al. «Overview of MicroRNA Biogenesis, Mechanisms of Actions, and Circulation». In: *Frontiers in Endocrinology* Volume 9 - 2018 (2018). ISSN: 1664-2392. DOI: 10.3389/fendo.2018.00402. URL: https://www.frontiersin.org/journals/endocrinology/articles/10.3389/fendo.2018.00402.

[19] David P Bartel. «MicroRNAs: genomics, biogenesis, mechanism, and function». In: *cell* 116.2 (2004), pp. 281–297.

[20] Maria Teresa Di Martino, Pierosandro Tagliaferri, and Pierfrancesco Tassone. «MicroRNA in cancer therapy: breakthroughs and challenges in early clinical applications». In: *Journal of Experimental & Clinical Cancer Research* 44.1 (2025), p. 126.

[21] Ji-Qing Chen et al. «The role of microRNAs in the pathogenesis of autoimmune diseases». In: *Autoimmunity reviews* 15.12 (2016), pp. 1171–1180.

[22] Temo Barwari, Abhishek Joshi, and Manuel Mayr. «MicroRNAs in cardiovascular disease». In: *Journal of the American College of Cardiology* 68.23 (2016), pp. 2577–2584.

[23] Laura DeFrancesco. *microRNAs blaze into the clinic*. 2025.

[24] Laura DeFrancesco. «microRNAs blaze into the clinic». In: *Nature Biotechnology* 43 (2025), pp. 1583–1585. DOI: `10.1038/s41587-025-02870-y`. URL: `https://doi.org/10.1038/s41587-025-02870-y`.

[25] Maria Yavropoulou and John Yovos. «The "dark matter" of DNA and the regulation of bone metabolism: The role of non-coding RNAs». In: *Journal of musculoskeletal & neuronal interactions* 18 (Mar. 2018), pp. 18–31.

[26] Alexander W Clarke et al. «MirGeneDB 3.0: improved taxonomic sampling, uniform nomenclature of novel conserved microRNA families and updated covariance models». In: *Nucleic Acids Research* 53.D1 (Nov. 2024), pp. D116–D128. ISSN: 1362-4962. DOI: `10.1093/nar/gkae1094`. eprint: `https://academic.oup.com/nar/article-pdf/53/D1/D116/60925558/gkae1094.pdf`. URL: `https://doi.org/10.1093/nar/gkae1094`.

[27] Bastian Fromm et al. «MirGeneDB 2.1: toward a complete sampling of all major animal phyla». In: *Nucleic acids research* 50.D1 (2022), pp. D204–D210.

[28] Ilias Glogovitis et al. «Isomirs–hidden soldiers in the mirna regulatory army, and how to find them?» In: *Biomolecules* 11.1 (2020), p. 41.

[29] Thomas Desvignes et al. «Unification of miRNA and isomiR research: the mirGFF3 format and the mirtop API». In: *Bioinformatics* 36.3 (2020), pp. 698–703.

[30] Adriana Ferre et al. «3′ IsomiR species composition affects reliable quantification of MiRNA/IsomiR variants by Poly (A) RT-QPCR: impact on small RNA-seq profiling validation». In: *International journal of molecular sciences* 24.20 (2023), p. 15436.

[31] Chung Wah Wu et al. «A Comprehensive Approach to Sequence-oriented IsomiR annotation (CASMIR): demonstration with IsomiR profiling in colorectal neoplasia». In: *BMC genomics* 19.1 (2018), p. 401.

[32] Ganesh Panzade et al. «Global profiling and annotation of templated isomiRs dynamics across Caenorhabditis elegans development». In: *RNA biology* 19.1 (2022), pp. 928–942.

[33] Ana Kozomara, Maria Birgaoanu, and Sam Griffiths-Jones. «miRBase: from microRNA sequences to function». In: *Nucleic Acids Research* 47.D1 (Nov. 2018), pp. D155–D162. ISSN: 0305-1048. DOI: `10.1093/nar/gky1141`. eprint: `https://academic.oup.com/nar/article-pdf/47/D1/D155/27437612/gky1141.pdf`. URL: `https://doi.org/10.1093/nar/gky1141`.

[34] Ernesto Aparicio-Puerta et al. «isomiRdb: microRNA expression at isoform resolution». In: *Nucleic Acids Research* 51.D1 (Oct. 2022), pp. D179–D185. ISSN: 0305-1048. DOI: `10.1093/nar/gkac884`. eprint: `https://academic.oup.com/nar/article-pdf/51/D1/D179/48440646/gkac884.pdf`. URL: `https://doi.org/10.1093/nar/gkac884`.

[35] Leandro Castellano and Justin Stebbing. «Deep sequencing of small RNAs identifies canonical and non-canonical miRNA and endogenous siRNAs in mammalian somatic tissues». In: *Nucleic Acids Research* 41.5 (Jan. 2013), pp. 3339–3351. ISSN: 0305-1048. DOI: `10.1093/nar/gks1474`. eprint: `https://academic.oup.com/nar/article-pdf/41/5/3339/16945337/gks1474.pdf`. URL: `https://doi.org/10.1093/nar/gks1474`.

[36] H. Rosaria Chiang et al. «Mammalian microRNAs: Experimental evaluation of novel and previously annotated genes». In: *Genes & Development* 24.10 (2010), pp. 992–1009. DOI: `10.1101/gad.1884710`. URL: `https://genesdev.cshlp.org/content/24/10/992.short`.

[37] Nicole Ludwig et al. «Bias in recent miRBase annotations potentially associated with RNA quality issues». In: *Scientific Reports* 7.1 (2017), p. 5162. DOI: `10.1038/s41598-017-05070-0`. URL: `https://www.nature.com/articles/s41598-017-05070-0`.

[38] Zhonglong Guo et al. «PmiREN: a comprehensive encyclopedia of plant miRNAs». In: *Nucleic Acids Research* 48.D1 (Oct. 2019), pp. D1114–D1121. ISSN: 0305-1048. DOI: `10.1093/nar/gkz894`. eprint: `https://academic.oup.com/nar/article-pdf/48/D1/D1114/31697758/gkz894.pdf`. URL: `https://doi.org/10.1093/nar/gkz894`.

[39] Michael Hackenberg et al. «Knowing is not enough; we must apply: the case for rigorous microRNA annotation standards». In: *Nucleic Acids Research* 53.19 (2025), gkaf1049.

[40] David P Bartel. «Metazoan micrornas». In: *Cell* 173.1 (2018), pp. 20–51.

[41] Kijun Kim et al. «A quantitative map of human primary microRNA processing sites». In: *Molecular cell* 81.16 (2021), pp. 3422–3439.

[42] S Chan Baek et al. «Structural atlas of human primary microRNAs generated by SHAPE-MaP». In: *Molecular Cell* 84.6 (2024), pp. 1158–1172.

[43] Georges Pierre Schmartz et al. «Encyclopedia of tools for the analysis of miRNA isoforms». In: *Briefings in Bioinformatics* 22.4 (2021), bbaa346.

[44] Klaudia Pawlina-Tyszko and Tomasz Szmatoła. «Benchmarking of bioinformatics tools for NGS-based microRNA profiling with RT-qPCR method». In: *Functional & integrative genomics* 23.4 (2023), p. 347.

[45] Knut Reinert et al. «The SeqAn C++ template library for efficient sequence analysis: A resource for programmers». In: *Journal of biotechnology* 261 (2017), pp. 157–168.

[46] Andreas Döring et al. «SeqAn: An efficient, generic C++ library for sequence analysis». In: *BMC Bioinformatics* 9 (2008), p. 11. DOI: `10.1186/1471-2105-9-11`. URL: `https://doi.org/10.1186/1471-2105-9-11`.

[47] Hannes Peer Hauswedell. «SeqAn3–Sequence Analysis and Modern C++». PhD thesis. 2021.

[48] Gianvito Urgese et al. «BioSeqZip: a collapser of NGS redundant reads for the optimization of sequence analysis». In: *Bioinformatics* 36.9 (Jan. 2020), pp. 2705–2711. DOI: `10.1093/bioinformatics/btaa051`. eprint: `https://academic.oup.com/bioinformatics/article-pdf/36/9/2705/48985134/bioinformatics_36_9_2705.pdf`. URL: `https://doi.org/10.1093/bioinformatics/btaa051`.

[49] Arun H Patil et al. «A curated human cellular microRNAome based on 196 primary cell types». In: *GigaScience* 11 (Aug. 2022), giac083. ISSN: 2047-217X. DOI: `10.1093/gigascience/giac083`. eprint: `https://academic.oup.com/gigascience/article-pdf/doi/10.1093/gigascience/giac083/45809922/giac083.pdf`. URL: `https://doi.org/10.1093/gigascience/giac083`.

[50] Philip A Ewels et al. «The nf-core framework for community-curated bioinformatics pipelines». In: *Nature biotechnology* 38.3 (2020), pp. 276–278.

[51] National Center for Biotechnology Information. *NCBI*. `https://www.ncbi.nlm.nih.gov/`. Accessed: 26 November 2025.

[52] @Midnighter @drpatelh. *nf-core subworkflow:* $fastq_download_prefetch_fasterqdump_sratools$. `https://nf-co.re/subworkflows/fastq_download_prefetch_fasterqdump_sratools/`.

[53] Simon Andrews. *FastQC: A Quality Control Tool for High Throughput Sequence Data*. `https://www.bioinformatics.babraham.ac.uk/projects/fastqc/`. Online; accessed <date>. 2010.

[54] @drpatelh @grst @ewels @FelixKrueger. *nf-core module: fastqc*. `https://nf-co.re/modules/fastqc/`.

[55] @drpatelh @kevinmenden. *nf-core module: fastp*. `https://nf-co.re/modules/fastp/`.

[56] @drpatelh @kevinmenden. *nf-core module: cutadapt.* `https://nf-co.re/modules/cutadapt/`.

[57] Wenjing Kang et al. «miRTrace reveals the organismal origins of microRNA sequencing data». In: *Genome biology* 19.1 (2018), p. 213.

[58] @atrigila. *nf-core module: mirtrace$_q$c.* `https://nf-co.re/modules/mirtrace_qc/`.

[59] GeunYoung Sim et al. «Determining the defining lengths between mature microRNAs/small interfering RNAs and tinyRNAs». In: *Scientific Reports* 13.1 (2023), p. 19761.

[60] PostgreSQL Global Development Group. *PostgreSQL: The world's most advanced open source relational database.* `https://www.postgresql.org/`. Accessed: 26 November 2025.

[61] Arun H Patil and Marc K Halushka. «miRge3.0: a comprehensive microRNA and tRF sequencing analysis pipeline». In: *NAR Genomics and Bioinformatics* 3.3 (July 2021), lqab068. ISSN: 2631-9268. DOI: `10.1093/nargab/lqab068`. eprint: `https://academic.oup.com/nargab/article-pdf/3/3/lqab068/56833751/lqab068.pdf`. URL: `https://doi.org/10.1093/nargab/lqab068`.

[62] Phillipe Loher et al. «IsoMiRmap: fast, deterministic and exhaustive mining of isomiRs from short RNA-seq datasets». In: *Bioinformatics* 37.13 (Jan. 2021), pp. 1828–1838. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btab016`. eprint: `https://academic.oup.com/bioinformatics/article-pdf/37/13/1828/50340190/btab016.pdf`. URL: `https://doi.org/10.1093/bioinformatics/btab016`.

[63] Ernesto Aparicio-Puerta et al. «sRNAbench and sRNAtoolbox 2022 update: accurate miRNA and sncRNA profiling for model and non-model organisms». In: *Nucleic Acids Research* 50.W1 (May 2022), W710–W717. ISSN: 0305-1048. DOI: `10.1093/nar/gkac363`. eprint: `https://academic.oup.com/nar/article-pdf/50/W1/W710/44379497/gkac363.pdf`. URL: `https://doi.org/10.1093/nar/gkac363`.