U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

MSc

GENEANALYST – A web application for whole genome visualization and analysis of gene expresison data

Filipe Manuel Salvador Caramelo

Dissertação de Mestrado apresentada à
Faculdade de Ciências da Universidade do Porto em
Bioinformática e Biologia Computacional

2022

MSc

2.º
CICLO

FCUP
2022

U.PORTO

GENEANALYST – A web application for whole genome visualization and analysis of gene expression data

Filipe Manuel Salvador Caramelo

FC

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

MSc

GENEANALYST – A web application for whole genome visualization and analysis of gene expression data

Filipe Manuel Salvador Caramelo
MSc in Bioinformatics and Computational Biology
Computer Science Department
2022

Supervisor
Luís Filipe Costa de Castro, Auxiliary Professor, Faculty of Science of University of Porto

Co-supervisor
Inês Dutra, Auxiliary Professor, Computer Science Department of University of Porto

U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

” Dream. Not of what you are, but of what you want to be.” - Lotus

Sworn Statement

I, Filipe Manuel Salvador Caramelo, enrolled in the Master Degree of Bioinformatics and Computational Biology at the Faculty of Sciences of the University of Porto hereby declare, in accordance with the provisions of paragraph a) of Article 14 of the Code of Ethical Conduct of the University of Porto, that the content of this dissertation reflects perspectives, research work and my own interpretations at the time of its submission.

By submitting this dissertation, I also declare that it contains the results of my own research work and contributions that have not been previously submitted to this or any other institution.

I further declare that all references to other authors fully comply with the rules of attribution and are referenced in the text by citation and identified in the bibliographic references section. This dissertation does not include any content whose reproduction is protected by copyright laws.

I am aware that the practice of plagiarism and self-plagiarism constitute a form of academic offense.

Filipe Caramelo

28/09/2022

Acknowledgements

Firstly, I would like to express my gratitude and admiration to **Filipe Castro**, not only for taking me into his team and challenging me with this dissertation topic, but also for his commitment, his direction, and his readiness to assist at any moment.

I cannot thank Professor **Inês Dutra** enough for approving my dissertation. Thank you for your ceaseless support, for always being there for me when I needed guidance, for your round-the-clock assistance, and for the kind and heartfelt words you have always given me.

I would also want to express my appreciation for the opportunity and vote of confidence that has allowed me to collaborate with the **AGE** (Animal Genetics and Evolution) department at CIIMAR. I would like to give a special thanks to **André Machado** for his support and guidance throughout the entire dissertation, **Raquel Ruivo,** for her daily positive attitude, unwavering support, and willingness to assist, and **Marcos Domingues**, my buddy, who helped me through the dissertation and was always there to make me laugh and support me, thank you.

To **my mother**, who has known me longer than anybody else and has never stopped doubting me, whose will was strong enough to guide me and my brother into the men we are today: I hope you realize every day how much I admire and love you. A heartfelt thank you to my most stalwart accomplice since I could not accomplish any of these feats without you by my side. This is your dissertation as well.

To **Dudu**, my kryptonite, thank you so much for all you have done for me, not only for the support you provided throughout my dissertation, but for being such a pillar of strength in my life; I cannot express how much you mean to me and how much you motivate me to pursue my dreams. This dissertation is also yours. I am aware that you dislike clichéd things, however... I love you; many thanks for being YOU.

To **Luís**, my real-life Lotus, whom I shall forever cherish with all my heart. Over the last few years, I have grown with you, and I will never see our journey with lightness. Thank you for the times you listened to me whine and for always being the reassuring voice that led me through the darkness. Thank you, Tenno.

To **Juca**, the Pennywise to my Georgie, thank you for being there for me through the countless hours of smashing tomatoes and the words of consolation and occasionally counsel, when I fell off the rails. I will never be able to express how much I love you, yet I really hope that this will warm your kind heart. A particular thank you, my darling child.

To **Alice**, I wanted to say a special thank you for always being there for me, for all the love and friendship we shared over the years and for always being willing to listen to my drama. You are a one-of-a-kind soul, and I cannot express how much I genuinely love you and your "*crepiocas*".

To **my baby girls**. Since the beginning we bond instantly and for that I am always thankful, thank you for every "*apontamento*", for every Zoom session, for every toast we ever did and for every "*descasca*”. Thank you, babes! I will always be here for you. ♥

And finally, to **Professor Domitília** for her constant support and for believing in me throughout the years, I couldn’t do it without your fondness and support, and, to my **family**, for all they had to sacrifice for me to be here, I hope I never let you down.

Resumo

O genoma nuclear de um organismo contêm a informação essencial para identificar as características e propriedades de cada tipo celular em todos os seres vivos, fazendo com que a sua compreensão seja um objetivo crucial na Biologia, desde a descoberta do ADN.

Os recentes avanços no domínio das tecnologias de sequenciação de genomas foram também determinantes nas abordagens do estudo da expressão genética conhecidas como RNA-Seq. Neste contexto, têm emergido múltiplas técnicas informáticas e análises estatísticas, substituindo os estudos de expressão genética clássicos com *microarrays* ou baseadas em PCR. Significativamente, a procura por ferramentas de fácil acessibilidade, intuitivas e amigáveis ao utilizador, que permitam a biólogos e bioinformáticos acederem a estudos de expressão génica em espécies não-modelo representa um passo importante em estudos de Biologia comparativa.

O objetivo central deste trabalho foi desenvolver uma ferramenta que ultrapassasse as limitações impostas por aplicações pré-existentes, através do desenvolvimento de um portal de genomas e que igualmente incorporasse informações de RNA-Seq, mantendo a possibilidade de uso em espécies modelo e não-modelo. Assim, foi desenvolvido o GeneAnalyst que permite a exploração e fácil visualização de vários conjuntos de dados *ómicos*, tal como a rápida e intuitiva interrogação e visualização de padrões de expressão génica gerados a partir de dados coletados de RNA-Seq com requisitos mínimos de programação. O comportamento do GeneAnalyst foi testado e comparado com dados pré-existentes, assim como com dados gerados *de novo*. Os resultados sugerem que esta aplicação apresenta uma robustez e rapidez em linha com o objetivo inicial.

Keywords: RNA-Seq, Visualização Genómica, Visualização de Anotações, Padrões de expressão génica, Bioinformática

Abstract

The genome sequence contains the information essential to identify the characteristics and activities of each cell type in every living organism, making its decipherment a crucial objective in Biology.

As a consequence of the advances in sequencing technology, new approaches based on next-generation sequencing such as RNA-Seq have emerged in conjunction with informatic and statistical techniques to replace microarrays in the analysis of gene expression. Demand for easy, intuitive, and user-friendly processes that enable biologists and bioinformaticians to access gene expression research has been sparked by the revelation of information supplied by RNA-Seq data.

Here, we develop GeneAnalyst, an application that aims to overcome limitations imposed by existing applications. In the context, er developed a hub that can be used on genome model and non-model species, enabling the exploration and visualization of various omic datasets, as well as the quick, intuitive interrogation and visualization of RNA-Seq generated gene expression profiles with minimal or no coding requirements. GeneAnalyst's performance was assessed and compared with new and pre-existing data. Underlying this evaluation, it presented sturdiness and efficient results.

Keywords: RNA-Seq, Genome Visualization, Annotation Visualization, Gene Profiling, Bioinformatics

Additional Information

GeneAnalyst is protected and requires user authentication, it can be accessed with the following credentials:

Link: http://portugalfishomics.ciimar.up.pt/app/geneanalyst
User: geneanalyst
Password: AnAlYsT!AGE2022

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AWS	AMAZON WEB SERVICE
BAM	BINARY ALIGNMENT AND MAP
BPKM	BASES PER KILOBASE OF GENE MODEL PER MILLION MAPPED BASES
BWT	BURROWS-WHEELER TRANSFORM
CDNA	COMPLEMENTARY DNA SEQUENCE
CDS	CODING REGIONS SEQUENCES
DNA	DEOXYRIBONUCLEIC ACID
FM-INDEX	FULL-TEXT INDEX IN MINUTE SPACE
FPKM	FRAGMENTS PER KILOBASE OF EXON PER MILLION MAPPED FRAGMENTS
GFF	GENERAL FEATURE FORMAT
GTF	GENE TRANSFER FORMAT
HPC	HIGH-PERFORMANCE COMPUTING
HTS	HIGH-THROUGHPUT SEQUENCING
ISX	INTESTINE SPECIFIC HOMEOBOX
IUPAC-IUB	INTERNATIONAL UNION OF PURE AND APPLIED CHEMISTRY-INTERNATIONAL UNION OF BIOCHEMISTRY
KCNE2	POTASSIUM VOLTAGE-GATED CHANNEL SUBFAMILY E REGULATORY SUBUNIT 2
MRNA	RIBONUCLEIC ACID MESSENGER
NGS	NEXT-GENERATION SEQUENCING
OGS	OFFICIAL GENE SET
PASA	PROGRAM TO ASSEMBLE SPLICED ALIGNMENTS
PDX1	PANCREATIC AND DUODENAL HOMEOBOX 1
PXR	PREGNANE X RECEPTOR
RMC	RNA MOLAR CONCENTRATION
RMSD	ROOT-MEAN-SQUARE DEVIATION OF ATOMIC POSITIONS
RNA	RIBONUCLEIC ACID
RNA-SEQ	RNA SEQUENCING
RPKM	READS PER KILOBASE OF EXON PER MILLION READS MAPPED
RRNA	RIBOSSOMAL RNA
RT-PCR	REAL TIME POLYMERASE CHAIN REACTION
SAM	SEQUENCE ALIGNMENT AND MAP
SRA	SEQUENCE READ ARCHIVES
TPM	TRANSCRIPTS PER MILLION

Chapter 1

Introduction

1. From “Letters” to “Information”: Welcome to the Age of the Genomes

1.1. Genome Assembly

The genome sequence represents the entirety of an organism or cell genetic information. Nucleic acid sequences are the central keepers of this information, and thus deciphering the genome sequence of an organism has been a critical quest in Biology.[1]

Actually, Watson and Crick's 1953 description and proposal of the double helix structure of deoxyribonucleic acid (DNA)[2], with the key contribution of Rosalind Franklin, was a fundamental event in the history of Biology. Importantly, it coincided temporally with the first protein sequence of the first two chains of the insulin protein, produced by Frederick Sanger.[3,4] This marked the beginning of the attempts to decipher these genetic instructions. In Sanger’s approach, proteins were cleaved and individually analysed; being then overlapped on their matched sequences from each fragment to produce a consensus sequence.[5] These initial attempts were slow and faced multiple technical hurdles. Later developments would entail vast modifications in pace and sequencing strategies[6–26] (Figure 1). Since those initial steps, genome assembly progressed through critical technologically and conceptually changes, leading to the transition to next-generation sequencing techniques (Figure 1).

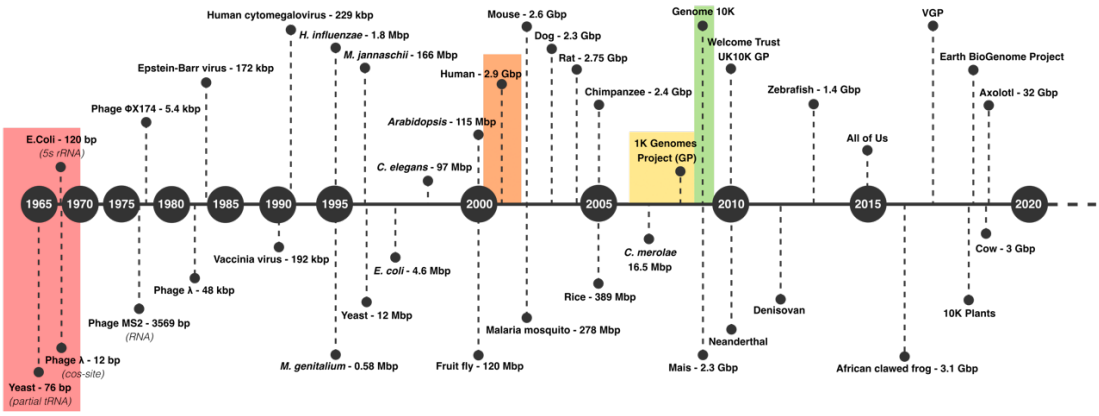


Figure 1 – Timeline representation key milestones in genome sequencing. Red-highlighted regions show the initial attempts at genome assembly and the important steps of the Human Genome Project. Adapted from *Giani et al.*[5]

Currently, high-throughput sequencing (HTS) represents a collection of techniques that empower large-scale genomic and transcriptomic sequencing. They differ from traditional Sanger sequencing by providing vast parallel analysis through the sequencing of billions of DNA nucleotides simultaneously, resulting in the production of immense quantities of data. Critically, it reduces the need for fragment-cloning techniques[6], as these methodologies can be accomplished without the need for constructing amplifications, while achieving a much significant reduction in cost.[7] This advantage — as well as the amount of data generated by these new methodologies — allows for a better understanding of various concepts in the broad field of "omics", from genomics to transcriptomics.[8] However, the massive amounts of output data represent a critical challenge for the handling and extraction of meaningful biological information.

Most initial high-throughput sequencing methods utilize a shotgun sequencing strategy (Figure 2). A simplified description of the assembly process includes the DNA being cleaved into relatively small arbitrary fragments that are sequenced by effective sequencing equipment. A fundamental step is the use of powerful computer algorithms that will piece together these reads in order to form a continuous sequence called a contig; this is known as *de novo* assembly[9] (Figure 2).

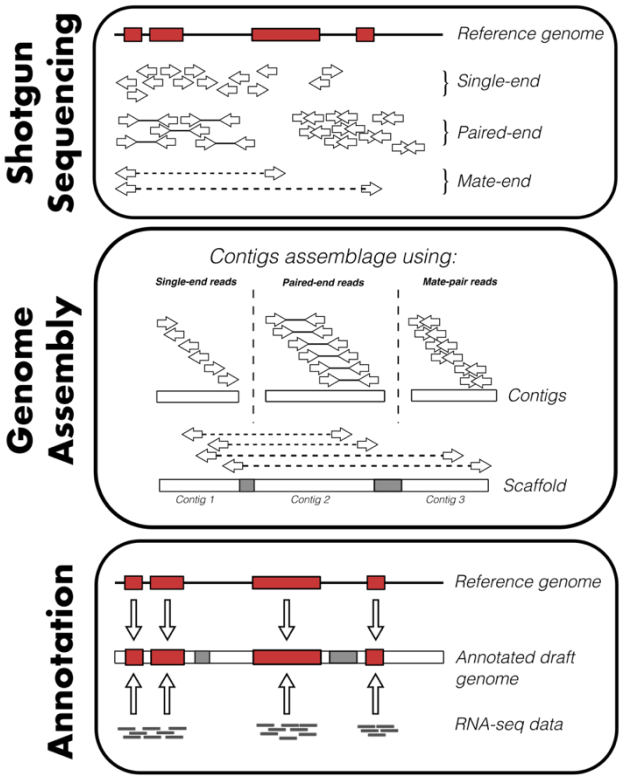


Figure 2 – Simplified illustration of the assembly and annotation procedure used in NGS sequencing. Adapted from *Ekblom et al.*[9]

After the initial assembly, paired-end libraries, comprised of long DNA fragments, are prepared by sequencing the ends of the fragments (e.g., mate-pair sequencing). This data will then be used as a foundation to combine several contigs into a scaffold if numerous independent fragments overlap.[9,10] The predicted fragment length of the library indicates the physical distance between two contigs, and the space between them is filled with the non-informative base-pair character "N". Later techniques of filling in the missing base-pair information, notably using reads that traverse repeated sequences, aid to fill in the missing information.[9]

When attempting an assembly at the chromosomal level, the resulting scaffolds can be joined into clusters or placed on genetic maps constructed from pedigree data for ordering and orienting.[11] However, detailed genetic maps require substantial genotyping efforts and profound pedigrees with enough meiosis, which are difficult to obtain in most biological systems. Given these challenges, chromosome-level assembly is not necessary for many conservation biology applications.[9,12] Recent developments in this field have led to the establishment of long read sequencing technologies that avoid the need to shot gun sequence (e.g., PacBio and Nanopore).

1.2. Annotation

For a genome sequence to be comprehended biologically, it must be annotated to its fundamental units, the genes.[9] The annotation success is contingent on the quality of the genome assembly, which can be assessed using a variety of statistical approaches that, when applied, define how full and contiguous an assembly is[13]; yet it requires considerable effort and bioinformatics expertise.[9]

As a result of advancements in sequencing techniques, high-quality genome assemblies are now accessible at more affordable rates, offering substantial sources for phylogenetic data, which in turn have improved whole-genome alignments and annotation.[14]

The annotation process may be theoretically separated into two steps: annotation at a structural level and annotation at a functional level[15] (Figure 3). (1) Structural annotation involves: masking repeated sequences regions; identification of known genes, genetic markers, and other landmarks based on homology; prediction of gene structures; and construction of a comprehensive catalog of the organism's proteins, identifying them and assigning them potential functions.[16] (2) Functional annotation associates genes available in the Official Gene Set (OGS) with genes and other genomic components predicted in the structural annotation step[16].

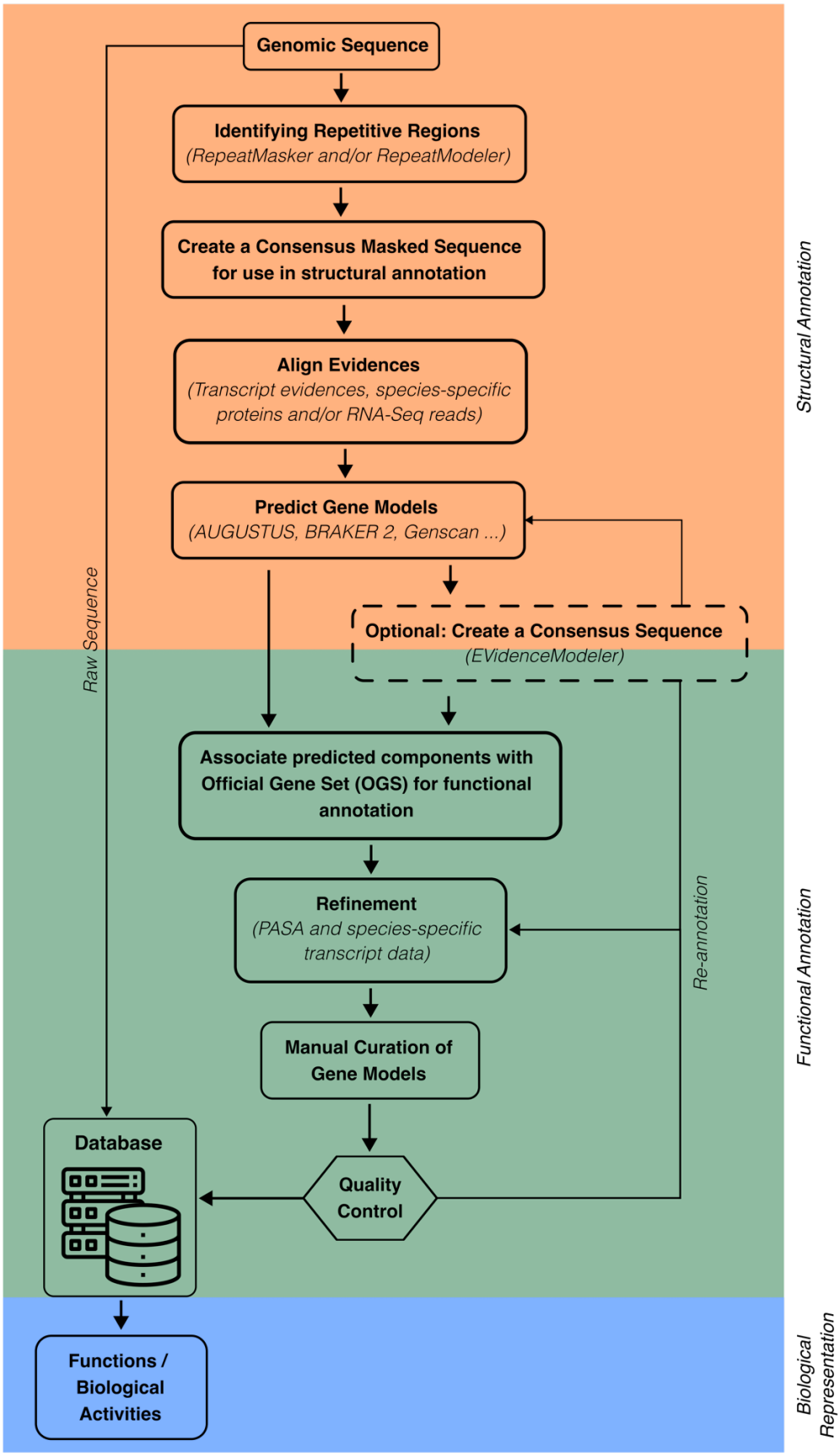


Figure 3 – Graphical flow of an annotation pipeline illustrating the two annotation steps. Adapted from: Ejigu and Jung[13] and *Humann et al.*[16].

Prior to gene prediction, it is crucial to mask repetitive sequences, such as low-complexity regions and transposable elements, since repeated regions are often poorly conserved across species.[9] For this reason, it is recommended to generate species-specific repeat libraries using tools such as RepeatMasker[13] so that *ab initio* algorithms, such as AUGUSTUS, Genie EST, and GenomeScan, trained on gene models from closely related species, may be used to predict coding sequences.[17] Protein alignments are important additions to these prediction models to supplement these methods. In conclusion, all the evidence predicted by these *ab initio* predictions and protein alignments must be gathered for a final set of gene annotations, which frequently benefit from manual curation. Several automated annotation tools, such as MAKER and BRAKER, assist in this curation, but qualitative validation is of utmost importance.[9]

Finally, the functional annotation step starts by associating the predicted gene models with the Official Gene Set (OGS). The OGS is a set of gene models, selected by the user, in which functional annotation software are based on to relate predicted gene models to their validated equivalents.[16] This step requires manual curation and the selection of the appropriate OGS by the user. A supplementary, yet optional step, is to combine the OGS association through PASA (Program to Assemble Spliced Alignments) with transcript evidence to further refine gene structure junctions and start and stop positions.[16]

2. From DNA to function: the history of RNA-Seq methods

In living organisms, DNA encodes the information required to define the traits and activities of each cell type. From this blueprint, cells dynamically access and translate specific instructions via "gene expression" to generate functional gene products by reading the genetic information stored in DNA. This process yields ribonucleic acid messenger (mRNA) molecules which code for proteins or non-coding RNAs. Thus, at its most fundamental level, synthesis is what connects information contained in the DNA to an observable feature: the initial trigger transcribes DNA into mRNA molecules, which can then be translated into proteins, or be used directly to control gene expression. Consequently, the set of transcribed RNA can reveal not only cell/tissue/organisms' functional states, but also pathological mechanisms underlying diseases conditions.[18]

RNA molecules are fundamental components of all living organisms. Thus, the ultimate objective an RNA study is to determine their identity, "which gene?", and abundance, "how much?"[19] and has driven scientists towards the development of numerous sequencing technologies which altered our understanding of biology.[20–33] Several revolutionary HTS methods for DNA sequencing were developed in the mid to late 1990s

and used in commercial DNA sequencers by the year 2000, enabling the large-scale analysis of RNA sequences.

Prior to these high-throughput approaches, DNA arrays, in the form of dot plots and slot bots, were available in the 1970s[34], and are considered to have ushered in the age of bulk expression profiling. Using radioactive labelling, these arrays permitted the identification of homology or the expression study of a series of samples.[34] DNA arrays and other post-development methods, such as northern blotting, could only study one or a few genes at a time, yielding extremely restricted insights into the complexity of cellular processes.[35] Consequently, oligonucleotide chips and DNA microarrays were created to meet the demand for high-throughput procedures coupled with statistical and computer analysis, changing fields as varied as biology and medical diagnostics (Figure 4). Similarly, new methodologies, paired with informatic methods for evaluating the massive, outputted data, have been used to attempt to address several dogmas, resulting in significant leaps and breakthroughs in our knowledge of some biological processes.

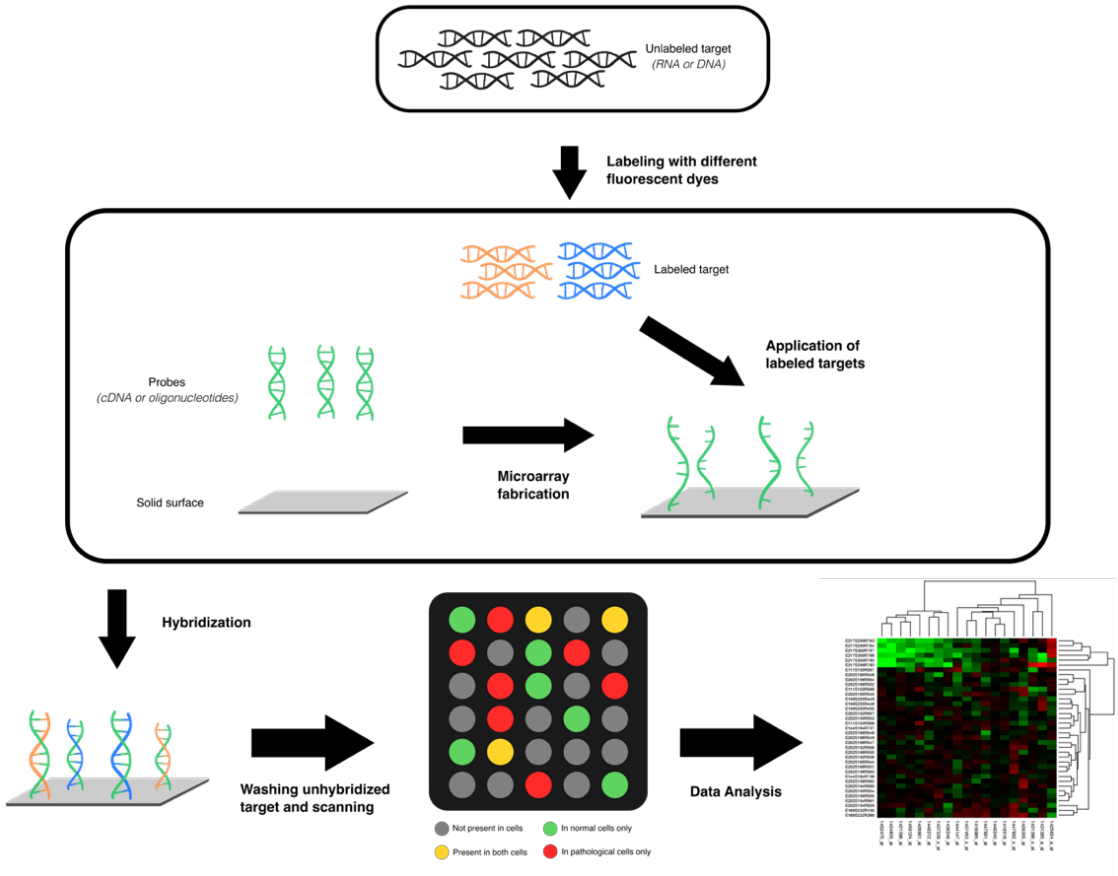


Figure 4 – Illustration of a DNA microarray experiment workflow. Adapted from Mlakar and Glavac[36].

New methodologies for RNA profiling, based on next-generation sequencing (NGS), called RNA Sequencing (RNA-Seq) when combined with informatic and statistical

methods are replacing microarrays in the study of gene expression.[18] These new methodologies allow to, simultaneously and at high resolution, study all RNAs present in a sample, characterize their sequences, and quantify their abundance.[18] RNA-Seq technologies have some advantages over prior microarrays approaches. The high level of reproducibility of RNA-Seq technologies reduces the number of technical replicates required for experiments. Also, it permits the identification and quantification of expressed isoforms and unknown transcripts. Importantly, the cost of next-generation sequencing experiments has significantly decreased [15,37] making it a valuable alternative. A quick search of the PubMed database for the terms "rnaseq gene expression" (accessed on May 31, 2022) gathered around 32,000 papers on this subject, a proxy for its relevance. However, a complete understanding of qualitative and quantitative analysis of RNA-Seq has not yet been obtained, and the scientific community continues to debate these topics regularly.[37,38]

The most common application of RNA-Seq involves the initial fragmentation of RNA samples into millions of short, tiny complementary DNA sequences (cDNA) fragments, referred to as "reads," and then sequenced from random locations on a high throughput platform. These reads are then computationally mapped to a reference genome or transcriptome to reveal a "transcriptional map", allowing for the estimation of expression levels for each gene or isoform. Finally, these estimations are transformed into standardized measurements using statistical and machine learning approaches, and their biological significance is then assessed (Figure 5).[18,37,39]

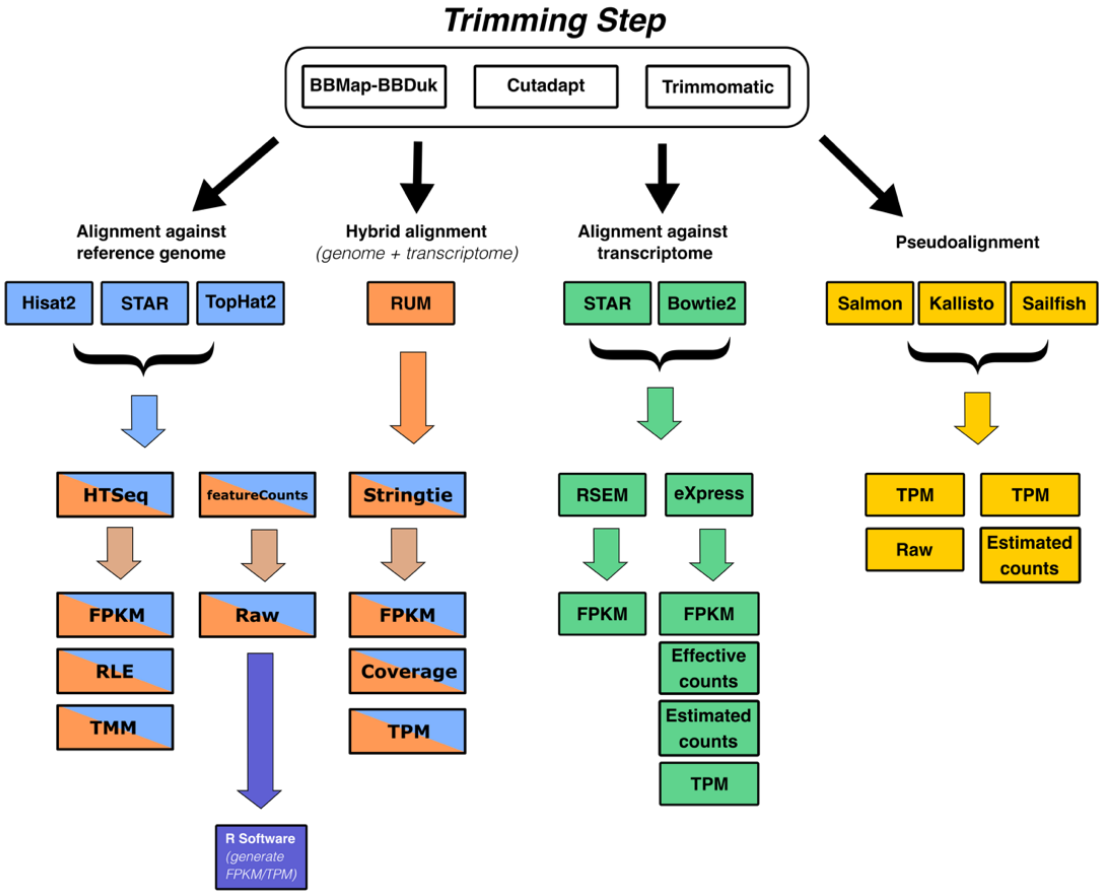


Figure 5 - RNA-Seq data analysis workflow represents the raw gene quantification workflow according to the various possible methodologies and their subsequent counting and normalization steps. Adapted from *Sánchez et al.*[40]

3. Key concepts in RNA-Seq

3.1. FASTA files

Bioinformatics and Molecular Biology adopt FASTA format, exemplified in Figure 6, to store and display nucleotide or amino acid sequences in a text-based format. A sequence encoded in FASTA format starts with a single-line identifying description (metadata, sequencer information, annotation, etc.) marked by the “>” character, followed by lines of the sequence itself.[41–43]

Figure 6 – Sample of a FASTA file containing various concatenated sequences illustrating the structure of the file. (*Source: Hosseini et al.*[44])

Nucleotides and amino acid codes are represented in the standard IUPAC-IUB (International Union of Pure and Applied Chemistry-International Union of Biochemistry) nucleic and amino acidic nomenclature[45], a commission on biochemical nomenclature. Figure 7 represents this nomenclature, with the following exceptions: (1) lower-case letters are accepted and mapped into upper-case, (2) a single hyphen can be used to represent a gap of indeterminate length, and "*" can be used in amino acid sequences, as it indicates translation stop.[41]

A.

Symbol	Meaning
G	*G*
A	*A*
T	*T*
C	*C*
R	*G / A*
Y	*T / C*
M	*A / C*
K	*G / T*
S	*G / C*
W	*A / T*
H	*A / C / T*
B	*G / T / C*
V	*G / C / A*
D	*G / A / T*
N	*G / A / T / C*

B.

Symbol	Meaning
G	*Glycine*
A	*Alanine*
V	*Valine*
L	*Leucine*
I	*Isoleucine*
P	*Proline*
F	*Phenylalanine*
Y	*Tyrosine*
C	*Cysteine*
M	*Methionine*
H	*Histidine*
K	*Lysine*
R	*Arginine*
W	*Tryptophan*
S	*Serine*
T	*Threonine*
D	*Aspartic acid*
E	*Glutamic acid*
N	*Asparagine*
Q	*Glutamine*
B	*Aspartic acid or asparagine*
Z	*Glutamic acid or glutamine*
.	*Terminator*
X	*Unknown*

Figure 7 – IUB-IUPAC code of symbols. A. Represents the nucleotide IUB-IUPAC nomenclature. B. Represents the amino acid IUB-IUPAC nomenclature.

There is no standard file extension for FASTA files containing formatted sequences; instead, a number of extensions are used to indicate the file's contents[46]: (1) `fasta` and `fa` are the general FASTA extension, (2) `fna` is used to define nucleic acids, (3) `ffn` is used to hold the coding portions of a genome, (4) `faa` stores amino acid sequences, while (5) `frn` stores non-coding RNA sections, such as transfer RNA (tRNA) and ribosomal RNA (rRNA), of a genome.

3.2. GTF/GFF files

Numerous bioinformatics programs depict structural and non-structural genomic annotation in Gene transfer format (GTF) and General feature format (GFF). These tab-delimited text file formats define the locations and properties of gene and transcript characteristics on the genome (chromosome or scaffolds/contigs).[47] GFF files are presently available in three distinct versions, the GTF file being a specific instance of the GFF version 2 format used by Ensembl containing special terminology such `transcript_id`, `protein_id`, and `gene_id`.

This file, with an example depicted in Figure 8, consists of nine tab-separated fields, apart from the last field, which should be signaled with a period when empty[48]:

(1) seqname - name of the chromosome or scaffold location of the supplied feature.
(2) source - name of the software that inferred the feature, or the data source (database or project name).
(3) feature - feature type, such as gene, exon, or transcript.
(4) start - start position of the feature in the genomic site.
(5) end - end position of the feature in the genomic location.
(6) score - a floating-point value representing the feature's score. It is highly advised to employ E-values for sequence similarity features and P-values for *ab initio* gene prediction features.
(7) strand - characterized as + (forward) or - (reverse).
(8) frame - for coding regions (CDS) features, the phase specifies when the next codon starts relative to the 5' end (where the 5' end of the CDS lies in relation to the strand of the CDS feature) of the current CDS feature.
(9) attribute - a semicolon-separated list of tag-value pairs that provide further information for each feature, such as, alias, parent of the feature and cross references.

```
scaffold1  MAKER  transcript  2622  3067  .  +  .  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  exon        2622  2667  .  +  .  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  CDS         2622  2667  .  +  0  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  start_codon 2622  2624  .  +  0  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  exon        2751  3067  .  +  .  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  CDS         2751  3064  .  +  2  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  stop_codon  3065  3067  .  +  0  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
scaffold1  MAKER  UTR         3065  3067  .  +  .  gene_id "ENSG00000XXX"; transcript_id "hypot_gene_01" ...
```

Figure 8 – Sample of a hypothetical GTF/GFF file demonstrating the file's structure.

3.3. SAM/BAM files

In bioinformatics, mapping software often generates sequence alignment and map (SAM) or binary alignment and map (BAM) files for the huge number of mapped reads. As noted earlier, alignment is a frequent step in many bioinformatics processes requiring nucleic acid sequencing using a variety of aligners. Most bioinformatics programs accept alignment results in the BAM format, which has several benefits over SAM, most notably its smaller file size. These files are used in conjunction with downstream analyses to give new inferences, such as comparing gene expression, surveying biodiversity, analyzing DNA methylation, and researching DNA-protein interaction.[49]

The alignment information of many sequences that were mapped to the reference genome is stored in SAM files, illustrated on Figure 9, which are human-readable text files including all information. Currently, SAM files are the most used output format for aligners that read FASTQ files and align sequences to a specified genome or transcriptome.[50]

```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:45
r001   99 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGAT *
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAGG   *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCT     * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCA     *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC         * SA:Z:ref,9,+,5S6M,30,1;
r001  147 ref 37 30 9M         =  7 -39 CAGCGGC       * NM:i:1
```

Figure 9 – Sample of the SAM file. (*Source: Hosseini et al.*[44])

Both SAM and BAM files contain the same information, but in different formats. BAM files contain the same data but in a compressed, indexed and in binary form, which is not readable by humans. This format reduce the size of the information, allowing algorithms to perform fast queries over the data, saving time, and reducing costs of computation and storage.[51]

3.4. Gene expression measurements

Unveiling valid information from RNA-Seq data requires careful validation of gene identity and the selection of quantification measurements. These steps are particularly critical for inter-sample comparisons and downstream analysis, such as differential expressions between conditions.[52] Several approaches have been presented, since the appearance of these methodologies, and are commonly used. However, the scientific community has

not established a complete consensus about the optimal gene expression quantification method for RNA-Seq data processing procedures.[53]

Due to the nature of these metrics, transcripts per million (TPM), reads per kilobase of exon per million reads mapped (RPKM) and fragments per kilobase of exon per million mapped fragments (FPKM) are suitable for comparing RNA transcript expression across various gene expression data analysis within a single sample. However, due to the shortfall of irrefutable experimental data, the scientific community has yet to achieve an agreement over which RNA-Seq quantification measure should be used for cross-sample comparisons.[53–58] Such non-agreement has led the science community to revise new measurement, e.g., TMM, SCnorm, and GeTMM, by performing transformations such as median centering and unit variance scaling (Z-score) or re-normalization, to increase the comparability of the expression measures.[53–58]

In this section the detail aspects of RPKM, FPKM, and TPM measurements will be discussed. The selection of these measurements in later discussions was based on the fact that for collected data analyses, cross-sample comparisons, and differential expression (DE) analysis, many current peer-reviewed publications, databases, bioinformatic applications and platforms continue to use TPM or RPKM/FPKM.[53]

3.4.1. RPKM/FPKM

RPKM is a gene expression unit that assesses the expression levels of genes or transcripts. It was proposed by *Mortazaci et al. in 2008*[59] to enable clear comparability of transcript levels within and across samples. This unit rescales gene counts to accommodate for variances in gene length and library size.[52]

Similar to RPKM, FPKM is used in paired-end RNA-Seq analyses to standardize counts for paired-end RNA-Seq wherein two (forward and reverse) reads are sequenced from the same DNA fragment.[60] Equation 1 is used to calculate RPKM or FPKM for gene i.

$$RPKM_i \; or \; FPKM_i = \frac{q_i}{\frac{l_i}{10^3} \cdot \frac{\sum j \cdot q_j}{10^6}} = \frac{q_i}{l_j \cdot \sum j \cdot q_j} \cdot 10^9 \qquad (1)$$

Where q^i is the count of raw reads or fragments, l_i is the size of the feature, e.g., length of the gene or transcript, and $\sum_j q_j$ is defined by the total number of mapped reads or fragments.[53]

3.4.2. TPM

According to some studies[52,61], RPKM and FPKM do not accurately depict the relative quantification of RNA molar concentration (rmc) and can be misleading towards identifying differentially expressed genes due to the fact that the total normalized counts for each sample will vary; therefore, TPM was proposed as an alternative. TPM is constant and proportionate to the relative RNA molar concentration, in contrast to RPKM (rmc).[62]

TPM is an abbreviation for transcripts per million, and the total of all TPM values is the same for all samples, allowing for comparisons across samples.[63] TPM is defined by Equation 2.

$$TPM_i = \frac{\frac{q_i}{l_i}}{\sum j\,(\frac{q_i}{l_i})} \cdot 10^6 \qquad (2)$$

Where q^i is the count of raw reads mapped to the transcript, l_i is the size of the transcript, and $\sum_j q_j$ is defined as the sum of mapped reads to transcript normalized by transcript length.[53]

TPM is a more desirable unit for RNA abundance because it satisfies the invariance property and is proportional to the average rmc[53]; consequently, it has been implemented by the most recent computational models for transcript quantitative estimation, such as RSEM[63], Kallisto[64], and Salmon[65].

3.5. Web resources for RNA-Seq quantification

There is an increasing demand for simple, intuitive, and user-friendly procedures that allow researchers and personnel with limited bioinformatic skills to access gene expression studies.[66] These tools strive to offer precise and clear interpretation of results[66,67], while keeping up with the evolution of sequencing technologies.[66,68]

The Human Protein Atlas (https://www.proteinatlas.org), VastDB (https://vastdb.crg.eu/), and GTEx Portal (https://gtexportal.org/) are bioinformatic systems that make it simple and fast to access visualizations of RNA-Seq quantification datasets. However, these platforms are restricted to the genomes present therein, with some depending solely on model species and preventing the user from inputting additional non-model species to take advantage of the same visualizations. As its name implies, The Human Protein Atlas alongside GTEx Portal are exclusive to human data and do not provide data imports for the retrieval of comparable visualizations. VastDB, on the other hand, provides a broader variety of species, albeit limited to validated model species such as human, rat, cow,

chicken, and fruit fly. Yet, this platform is particularly useful for determining where a gene is expressed on an individual species.

Several web applications, such as FX[69] (http://fx.gmi.ac.kr), RaNA-Seq[70] (https://ranaseq.eu), RAP[66] and rQuant.web[71] (https://galaxy.inf.ethz.ch/) have emerged in response to the growing demand for procedures that allow users to import their own data; all these tools are up and running except RAP, which has been decommissioned.

FX, which stands for user-friendly RNA-Seq gene expression analysis tool, only permits the import of files directly from the Amazon Web Service (AWS), and instead of delivering ready-to-interpret visualizations, it provides the user with a text-based file containing a metric, bases per kilobase of gene model per million mapped bases (BPKM), derived from RPKM, which is normalized to the expression level of each gene; making the user rely on long spreadsheets rather than intuitive visualizations.

RaNA-Seq was conceived as a cloud-based tool for the efficient processing and display of RNA-Seq data. The creators of the program guarantee a complete study in few minutes by filtering and quantifying FASTQ files[70], calculating quality control measures, and performing differential expression analysis. This distinguishes it from the other applications described, since it lends itself more to differential expression analysis than to gene profiling.

As an almost direct rival to GeneAnalyst's gene expression component, rQuant.web is a Galaxy hub-hosted application that allows users to submit their own RNA-Seq data for analysis. Although it is a helpful tool, it seems to be restricted to a specific number of genomes (all from before 2009), rendering it unsuitable for anyone attempting to apply similar visualizations to species that are not yet accessible.

As will be discussed later, GeneAnalyst aimed to break these limitations and create an application that could be applied to any desired species, validated model species or not, allowing users to quickly visualize the annotation of a given genome, query through the genome using a BLAST function, and quickly and intuitively interrogate and visualize RNA-Seq datasets with minimal to no coding requirements.

4. From Models to Non-Models: gene expression hub for all

4.1. Validated Animal Species

The development of HTS technologies has had a massive impact in multiple fields of Biology. Yet, specific bottlenecks persist specifically related with the capacity to handle the vast amounts of data and bioinformatics training requirements for most users. This problematic scenario is particularly visible in fields using non-model biological species. In classical models such as humans or mice, the vast landscape of omic resources, from genomes to proteomes, have been treated, developed, and prepared for use by non-bioinformatic trained users. These interfaces facilitate the use and dissemination of studies. Yet, they are non-existent for the majority of the species.

Next, we describe the species used in this dissertation presenting a short description of each species and their biological relevance.

4.1.1. Callorhinchus milii

The Australian ghostshark (*Callorhinchus milii*), often referred to as the Australian elephant shark, is a cartilaginous fish (Chondrichthyes class) of the subclass Holocephali (Chimaera) found in temperate continental shelves (Figure 10). Primarily found in the seas off the southern coast of Australia, Tasmania, and New Zealand, elephant sharks are captured for commercial and recreational purposes, such as the preparation of "fish and chips".[72] *C. milii* was proposed as a model cartilaginous fish genome due to its relatively small genome size and its importance as a representative of the oldest living group of jawed vertebrates. Its study allows for the understanding of the origin and evolution of vertebrate genomes, including those of humans, who share a common ancestor with these sharks that lived approximately 450 million years ago.[73,74]

Figure 10 – Illustration of the Australian ghostshark (*Callorhinchus milii*) (*Source: MarineWise's database, accessed on 25th August of 2022*)

Currently, *C. milii* has two genomes published the NCBI's RefSeq database. The `Callorhinchus_milii-6.1.3` assembly was the first one produced for this species and its present in NCBI's RefSeq and Ensembl's database.[74] Recently, the same authors did a new version of the genome using cutting edge technologies, and established this assembly `IMCB_Cmil_1.0` as reference in NCBI's RefSeq database.[75] In terms of RNA-Seq data, a search on NCBI's database returned 340 SRAs (Sequence read archives datasets) ranging from RNA-Seq data collected from tissue to data collected from different embryo stages.

4.1.2. Gallus gallus

The red junglefowl (*Gallus gallus*), formerly known as the Bankiva or Bankiva Fowl, is a tropical bird belonging to the Phasianidae family that is generally found in Southeast Asia and portions of South Asia (Figure 11).[76] The domestication of the chicken from the red junglefowl occurred around 8,000 years ago, as shown by whole-genome sequencing.[76]

Regulation mechanisms for ovarian follicular growth and maturation, subsequent ovulation, as well as age-related alterations in ovarian functions make it the only animal model available to investigate pathogenesis and progression of human ovarian cancer. Also, its sensitivity and rapid response to toxicants and expression of external indicators makes it an ideal model for toxicology studies.[77]

Figure 11 – Illustration of the Red Junglefowl (*Gallus gallus*) (*Source: Birds of the World's database, accessed on 25th August of 2022*)

Gallus gallus species counts with 21 genomes published the NCBI's Genbank database, yet, only 3 of them are published in NCBI's RefSeq database as references/models. `GRCg6a` representing the Red Jungle fowl, inbred line, and `bGalGal1.mat.GRCg7b`

and `bGalGal1.pat.GRCg7w`, corresponding to the maternal and parental cross of Cross of Boiler breed, with the maternal being noted as the reference genome for the *Gallus gallus*. In terms of RNA-Seq data, a search on NCBI's database returned over 79 thousand SRAs ranging from RNA-Seq data collected from tissue to data collected from different tissues' microbiome.

4.2. Non-model Species

Hydrolagus colliei

The spotted ratfish (*Hydrolagus colliei*) is a cartilaginous fish (Chondrichthyes) of the subclass Holocephali (Chimaera) found in the north-eastern Pacific Ocean, mostly from south-western Alaska to California, but also along the coasts of Mexico and Costa Rica (Figure 12). Although plentiful, little is known about its biology and it is not currently targeted by commercial fisheries.[78–80] *Hydrolagus colliei* has been used as a model in a variety of studies, including those examining the diversity and evolution of mineralized skeletal tissues in Chondrichtyans[81], the biological activity of a bombesin-like peptide[82], and as a potential biomonitoring organisms of mercury and selenium in the deep-waters of the northern Gulf of California.[83] To date (accessed on September 26, 2022) no genome or SRA archives have been published.

Figure 12 – Photograph of the spotted ratfish (*Hydrolagus colliei*). (*Source: Angulo et al.*[84])

5. Motivation and Objectives

This dissertation was driven by the need for an effective, user-friendly, and efficient method to examine created genomes, query these genomes, and query RNA-Seq datasets in an easy and efficient approach. The aim was to develop an approach and interface to execute these tasks without prior programming requirements, hence saving time, money and facilitating an improved workflow, benefiting the overall scientific community.

In the initial stages of this dissertation, two main goals were outlined: (1) the fine-tuning of a pipeline that leverages pre-existing software to quickly and accurately compute and extract RNA-Seq quantification in a genome; and (2) the creation of a tool that allowed for the display of not only a genome and its annotations but also RNA-Seq gene expression quantification, as these visualizations were typically done with spreadsheets rather than quick and intuitive tools.

Specific Aims:

- Adjustment and construction of a pipeline able to handle RNA-Seq data and derive reliable biological inferences regarding gene expression profiles. This pipeline was to be developed using two model species, *C. milii* and *G. gallus*, evaluated against previously described results and publicly available data, and then applied to one non-model species, *H. colliei*, an in-house generated dataset.
- Implementation of a visualization platform for genome sequence and gene annotation that enables the gene data exploration as well as development of synteny studies directly from the tool.
- Implementation of a BLAST system using NCBI's BLAST tool to provide an easy-to-use tool to study the mentioned generated datasets.
- Implementation of intuitive visualizations and exploration tools, allowing not only a quick retrieval of gene expression query results from RNA-Seq data, but also the examination of the entirety of a tissue's gene expression profile through interactive tables.
- Incorporate all these components into a web application hub that users can easily access the species they choose to investigate while enabling for future extensions of the tool.

6. Thesis outline

This dissertation is divided into six major chapters. This section will provide brief descriptions of each.

Chapter 1 – Introduction. Provides an overview of the species used for benchmarking and pipeline application, as well as presenting the major rationale, aims, contributions, structural organization of the dissertation and introduces file formats to be utilized in this dissertation, and gene quantification measures.

Chapter 2 – GeneAnalyst. The chapter begins by introducing the GeneAnalyst tool's conception, followed by a discussion of comparable applications and how GeneAnalyst differs from them. Then, the functional and non-functional requirements of the tool will be described, followed by a description of each pipeline phase and web application development procedure.

Chapter 3 – Results: GeneAnalyst's performance evaluation. Describes GeneAnalyst's performance evaluation procedures carried out throughout work.

Chapter 4 – Integrating a genome browser with gene expression profiles in non-model species: a test case. Provides a hypothetical real-world scenario to test and demonstrate the capabilities of the GeneAnalyst web application.

Chapter 6 – Conclusion. This final chapter summarizes and concludes the created work, concluding with prospective perspectives and tasks to be completed in the future.

Chapter 2

GeneAnalyst

The GeneAnalyst web application will be discussed in this chapter, along with the identification of functional and non-functional needs that led to the development of this dissertation. As mentioned before in Chapter 1, the lack of tools to aid biologists query RNA-Seq data led to the formulation of this dissertation. We sought to fill this gap by developing an application that could be applied to any desired species, validated model species or not, allowing users to quickly visualize the annotation of a given genome, query through the genome using a BLAST function, and quickly and intuitively enquire and visualize RNA-Seq datasets with minimal to no coding requirements.

The initial stage was to understand the knowledge gap this application had to fulfil, to outline the essential aspects of the web application. The basic functions of this tool required three main critical points; Figure 13 shows the original work plan delineated during the initial stages of GeneAnalyst.

1) The ability to visualize in-house produced genomes to perform various analyses, e.g., synteny analysis, over the cloud, removing the requirement to have these genomes within each personal machine.
2) The ability to query gene expression using RNA-Seq data within the app. Due to the accumulated RNA-Seq datasets, a user-friendly interface is required for the non-specialists to enquiry these datasets graphically.
3) The ability to query unpublished genomes, using BLAST search.

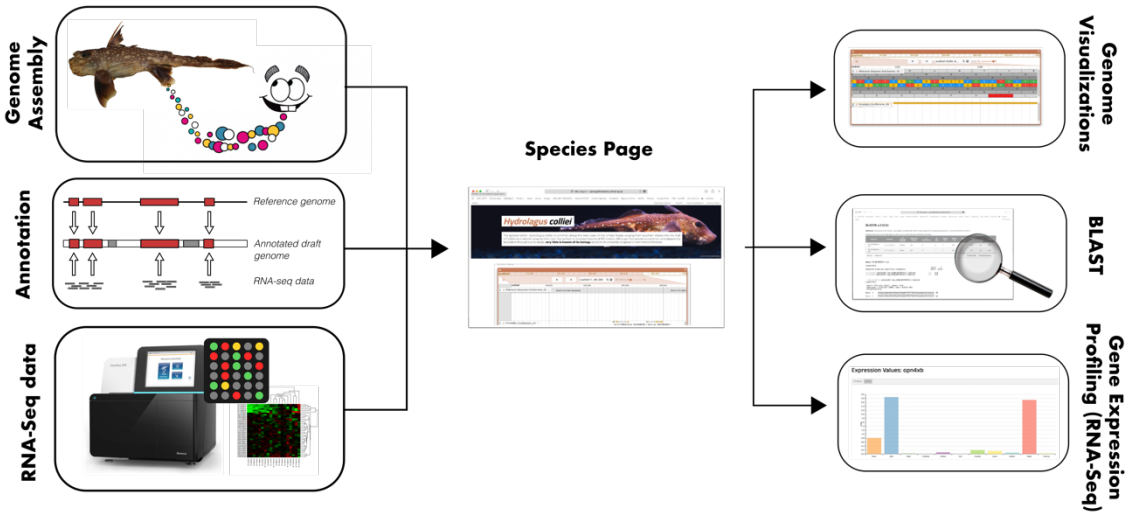


Figure 13 – Functional map schematic of GeneAnalyst during the early stages of development.

Another important criterion was to host GeneAnalyst inside the CIIMAR High-Performance Computing (HPC) cluster to allow a controlled access prior to release, *i.e.*, different members with granted access would be capable to open the application internally to perform such functions, mentioned in the above key points, without the need of downloading or computing within their personal computers.

The principle of this web application revolves around a "hub" where multiple omic resources would be added to provide quick, fast, and powerful dataset visualizations.

2.1. Characterization of the problem

As noted, the absence of tools to assist scientists in analyzing RNA-Seq data motivated the original development of this dissertation. Although web applications, such as the ones presented in Chapter 1, presented themselves as efficient solutions for generating gene profiling visualizations, these software lacked the user-friendliness, regular updates, and continuity needed to be used as regular tools; furthermore, these applications would require extensive file management and would need to be paired with other software such as BLAST, IGV[85,86], Geneious(http://www.geneious.com/) and GenomeBrowser[87] to efficiently explore the dataset at hand.

Some bioinformatics platforms such as the ones cited in Chapter 1 made it simple and fast to access visualizations of RNA-Seq quantification datasets; however, they are inflexible with respect to species selection by restricting the user to the available species and neglecting possible inputs from additional, user-provided, non-model species.

GeneAnalyst alleviates some of these obstacles by providing a hub that combines not only the quantification of gene expression, through RNA-Seq, but also a robust genome visualization component and the well-established BLAST tool. These features enable users to efficiently analyze and query different datasets, generated from a reference genome and its corresponding annotation, that could be used for both validated animal model species and non-validated species, without the need for heavy file manipulation or coding knowledge.

GeneAnalyst should be accessible to technicians with diverse backgrounds, ranging from biologists to bioinformaticians, and built to accommodate future expansions, such as the presentation of other file formats (not contemplated in this dissertation) and the addition of numerous additional species.

2.2. Application requirements

The web app requirements will be listed in this section, with functional requirements summarizing the operations required for users to perform within the platform and non-functional requirements summarizing the wholeness of the application other than its functionality, e.g., application reliability and user-friendliness.

2.2.1. Functional requirements

- Create a platform for all species genomes produced by the team.
- Create visually attractive visualizations for the produced genomes to undertake annotation visualization, perform gene synteny verification and quickly retrieval of information.
- Include a BLAST tool allowing the prospection of information in the generated genome.
- BLAST result retrieval, *i.e.*, the ability to download the table generated on the BLAST result page and choose specific hits for FASTA sequence download.
- The ability to search for a gene expression quantification profile and display it on a quick and understandable graph.
- The ability to export gene expression quantification for a single or multiple genes into a downloadable Excel file.
- The ability to browse spreadsheets by tissue if necessary and interact with the spreadsheet, *i.e.*, toggle or filter particular parts of the table.

2.2.2. Non-Functional requirements

- *Accessibility* – must be available to all common internet browsers using a consistent URL.
- *Adaptability* – must be versatile and allow for a wide range of test cases while also being adaptive to the team's demands.
- *Security* – while certain information may be confidential, certain sections of the web framework must be blocked if necessary.
- *Usability* – must be simple to understand, user-friendly, accessible to professionals with a variety of technical backgrounds.
- *Future Development* – must allow for future expansions by allowing not only the addition of new incoming species and RNA-Seq data, but also the possible addition of files formats not yet included on this dissertation, e.g., demonstrating how alignments perform with the visualization of BAM files generated by GeneAnalyst's pipelines.

2.3. Development of GeneAnalyst

A web application is a software that is kept on a remote server and delivered through the Internet using a browser interface. Web apps are abundant nowadays, from Google Docs to Evernote. Their design and execution have various options, with more being introduced every day. When building a web application, we begin by deciding on the type of application, followed by the environment, programming languages, technologies, and frameworks.

2.3.1. Application type

The first decision we had to make when considering the program's usage was to evaluate the domain of the application and its prospective expansions; in this line, two alternatives were feasible: construct a native application for desktop use or implement a web application.

We chose a web application as the focus of this dissertation, to not only avoided the overloading of the team's own devices, but also to allow for cloud-server computing, which eliminated the requirement for any user data computing and management.

While web apps have drawbacks — it is difficult to provide a pleasant user experience for consumers since they tend to become slower and less adjusted than to their native versions, *i.e.*, desktop or mobile applications designed to operate in these restricted environments — the benefits exceed the drawbacks.

2.3.2. Used technologies

Regarding the production environment and programming languages used, we considered several factors such as stability, popularity (as greater popularity translates to larger communities ready to help find solutions to problems that may arise), and most importantly, what was already available on CIIMAR's cluster server. Structurally GeneAnalyst is compartmentalized into two main components: back-end and front-end systems.

Back-end technologies

Python 3.6.8

Python is a dynamically typed programming language that supports a variety of programming paradigms such as structured, object-oriented, and functional programming.[88,89] Python was utilized in the work to automate processes such as the development of automated tabular files and HTML pages.

PHP 7.2 Development Server

PHP is a scripting language designed for web development; interpreting and executing PHP code on a web server results in the HTTP response of the webpage.[90,91] PHP was used to bundle all the created HTMLs together by providing redirections, to collect user inputs from HTML forms, and to generate and execute shell commands to take use of CIIMAR's HPC capabilities.

NCBI Blast v.2.12.0

BLAST is an NCBI algorithm and software for comparing biological sequences such as amino-acidic sequences of proteins, DNA or RNA sequences.[92,93] NCBI BLAST was used to establish a *blastable* database that allows subject sequences to be compared to the produced library.

Seqtk 1.3

Seqtk is a fast and lightweight tool for processing sequences in the FASTA or FASTQ format (https://github.com/lh3/seqtk.git). This software was used to extract sequences from all the available *blastable* files.

AGAT v0.9.2

AGAT (https://github.com/NBISweden/AGAT) is a toolkit that enables repairs, padding of missing information, and sorting of GTF and GFF annotation files, while attempting to standardize the GFF3 format.[94] AGAT was utilized to extract each sequence type, proteic, mRNA, gene, and CDS sequences, from the reference genome file utilizing the annotation information.

Front-end technologies

GeneAnalyst was constructed on the foundation of HTML, which was used to produce the content displayed on the sites, CSS for styling, and JavaScript, which was supported by the JavaScript library jQuery, to produce more interactive web pages.

JBrowse2

JBrowse is a genome browser application built for desktop and web-applications.[95] It was utilized as the foundation of the genome browser included with GeneAnalyst to display the genomic data. This decision was based on the extensive capabilities given by the program that related to aims of this work.

CSV to HTML Table

CSV to HTML table is a software available in GitHub (https://github.com/derekeder/csv-to-html-table) built on the principles of Datatables (http://datatables.net), a jQuery table plug-in. This program was used to incorporate the pipeline's tabular files into GeneAnalyst as a searchable, filterable HTML table.

Dimple.js

Dimple (http://dimplejs.org) is a robust and versatile open-source object-oriented charting API that utilizes D3, a JavaScript library, to generate dynamic and adaptable visualizations. It was utilized to generate interactive bar charts to rapidly present gene expression levels.

Other technologies

HISAT2

HISAT2[96–98] is a rapid and precise software for mapping reads from next-generation sequencing. The group utilized it to align reads from RNA sequencing studies into the created genomes by producing BAM files, which could then be used to determine the frequency of reads per gene.

FeatureCounts

FeatureCounts[99] is a very efficient tool that counts mapped RNA-Seq and DNA-seq reads for genomic features, such as genes, exons, promoters, and chromosomal locations, and was utilized for the counting of specific mRNA features.

R's Tidyverse collection

Tidyverse[100] is a collection of data science-oriented R tools. We utilized it to automate the processing of FeatureCounts files so that FPKM and TPM's gene expression measurement methods could be used.

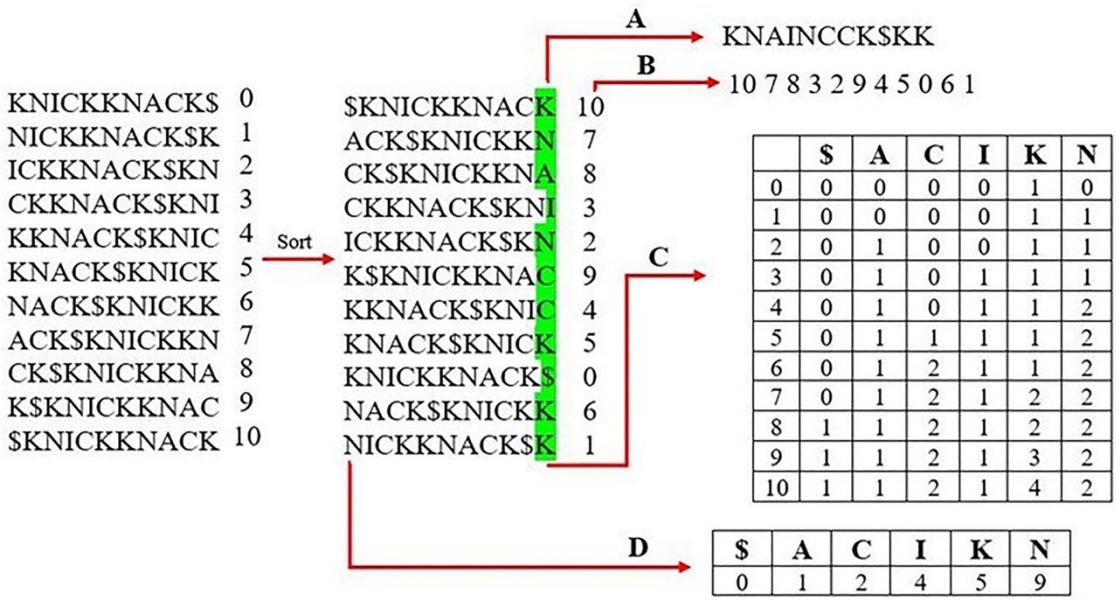
2.4. GeneAnalyst's pipeline in detail

As previously mentioned, GeneAnalyst was developed to provide a converged platform, a "hub", where multiple species could be added to provide quick, fast, and powerful dataset visualizations. The entire pipeline was designed taking in account the pre-requisition of the availability of reference genomes and respective structural and functional annotation.

2.4.1. Mapping next generation reads onto the genomes

Mapping next-generation reads to a reference genome to determine their genomic locations is the first step in RNA-Seq analysis. This mapping enables us to gather sets of reads for each gene and then measure the transcripts associated to those reads.

The majority of current aligners owe their efficiency to a data structure called Full-text index in Minute space or FM-index, which was theorized by Paolo Ferragina and Giovanni Manzini[101], in conjunction with the Burrows-Wheeler transform (BWT) (Figure 14) compression algorithm developed by Michael Burrows and David Wheeler[102] in 1994 and used to store the genome in a highly compressed form.



	$	A	C	I	K	N
0	0	0	0	0	1	0
1	0	0	0	0	1	1
2	0	1	0	0	1	1
3	0	1	0	1	1	1
4	0	1	0	1	1	2
5	0	1	1	1	1	2
6	0	1	2	1	1	2
7	0	1	2	1	2	2
8	1	1	2	1	2	2
9	1	1	2	1	3	2
10	1	1	2	1	4	2

$	A	C	I	K	N
0	1	2	4	5	9

Figure 14 – Brief representation of the creation of the FM-index of the word “knickknack”. (*Source: Musich et at.*[103])

To decide the correct aligner that we would use to map our sequence read archives (SRA) to the reference genome, we analyzed some studies[103–105] for both alignment distributions and runtime per read distribution. As a result, we determined that HISAT2 met our needs, not only did it compare favorably in terms of percentage of reads aligned to other widely used aligners such as BWA[106] and STAR[107], but also had a shorter runtime per read.

In order to accomplish the mapping of the SRA to the reference genome, we had to do the genome index using HISAT2's built-in methods. Two Python scripts within the software permitted the direct extraction of splice sites and exons straight from the annotation file used to produce this index.

The last step was to utilize the HISAT2 mapping program to map both FASTQ files within the SRA to the reference genome. In this mapping, the downstream transcriptome assembly option was used to improve computational and memory utilization with transcript assemblage, Appendix A shows the retrieved results of this step.

2.4.2. Quantification of gene expression using RNA-Seq data

After mapping the reads from an RNA-Seq experiment, the mapping findings must be translated into a gene expression profile. This profiling may be performed by counting the number of reads mapped to each gene.

With the growth of gene quantification utilizing RNA-Seq data, several studies[40,108–112] have been published attempting to demonstrate the efficacy of different read-quantification algorithms, discussing not just their processing time but also their precision.

During the development of the GeneAnalyst's pipeline, we were faced with the decision regarding quantification software. Initially, several alternatives were considered, including HTSeq[113], StringTie[114,115], and featureCounts[99].

As highlighted by Chandramohan *et al.*[109], HTSeq was the first to be discarded since it showed the highest deviation from RT-qPCR when RMSD (Root-mean-square deviation of atomic positions) analysis was done. In addition, according to a published study by featureCounts[99], it scored significantly worse than its counterpart in terms of computation time and memory use. As a result, we decided to test StringTie and featureCounts as our final alternatives, which were used on two of our chosen genomes and were compared.

To evaluate these approaches, gene expression profiling was computed using the reference genomes of two model species, *C. milii* and *G. gallus*, and 50 randomly selected genes were manually compared with gene expression profiling publicly available in NCBI. StringTie displayed a better rate of prediction than featureCounts, but had a much greater processing time and cost, preventing our system from using it for bigger genomes, the results are shown in Table 1.

Table 1 – Comparison of the Stringtie and featureCounts applications for predicting the gene expression profile of 100 randomly chosen genes with their corresponding computational elapsed time. Newly Found Expression refers to the discovery of expression on our dataset as opposed to NCBI's gene profiling.

	StringTie	***featureCounts***
	***Callorhinchus milii* gene expression profiling of 100 randomly selected genes (%)**	
Correctly Predicted	95	91.9
Wrongly Predicted	4.7	7.8
Newly Found Expression	0.3	0.3
Elapsed Time	12 hours	2 hours

Consequently, featureCounts was picked as our selected approach since it had the shortest processing time and did not deviate considerably from the findings produced with StringTie.

featureCounts receives SAM/BAM files and an annotation file with the chromosomal coordinates of features as inputs and returns a tabular file containing the number of correctly assigned reads. This program offers an option, `countReadPairs`, for counting paired-end readings by counting the reads as fragments, this option was used when performing the quantification.

The resulting tabular file was then edited in the R statistical software. A script was developed using Tidyverse's library for automatic use to calculate FPKM and TPM gene expression measurements and to manipulate the file for input for subsequent use. Code Snippet 1 unveils the functions developed in R to perform the calculation of FPKM and TPM gene measurements. Complete code source is available in Appendix B.

Code Snippet 1: Functions developed to perform the calculation of FPKM and TPM gene measurements. The script was partly inspired on the one published in https://haroldpimentel.wordpress.com/2014/05/08/what-the-fpkm-a-review-RNA-Seq-expression-units/ (Accessed on 15 March 2022)

```
mpkm <- function(counts, lengths) {
    N <- sum(counts)
    exp( log(counts) + log(1e9) - log(lengths) - log(N) )
}

tpm <- function(counts, lengths) {
    rate <- log(counts) - log(lengths)
    denom <- log(sum(exp(rate)))
    exp(rate - denom + log(1e6))
}
```

2.4.3. Web Application Architecture

The architecture of a web application refers to several design points of the application. In this section, a summary will be provided, outlying the rational between client and server sides while also describing the relationships and interactions between the application inner components: such as the tools used for computation or page connection bridges.

Figure 15 depicts the whole structure map of GeneAnalyst's architecture. This figure defines the conceptual framework and pages, their accompanying linkages, as well as the tools that make up the overall web application.

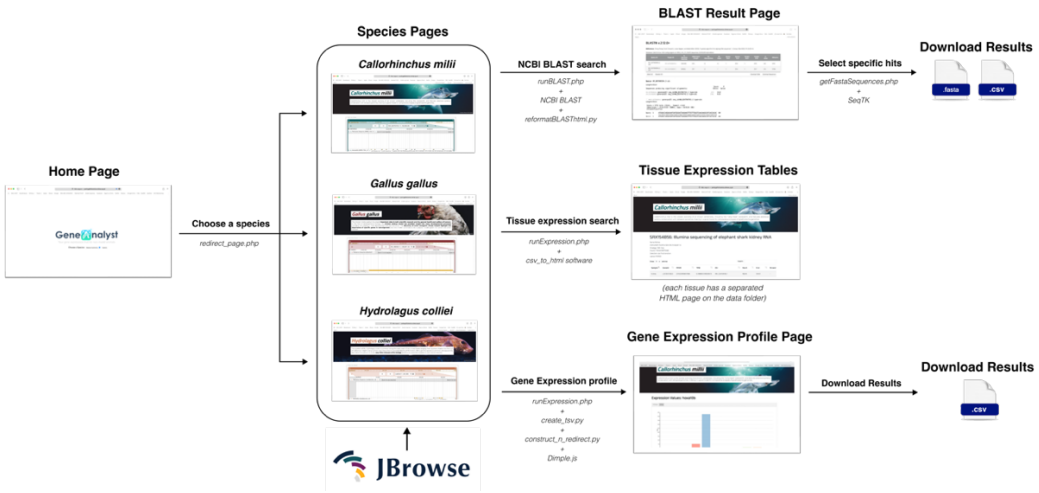


Figure 15 – Structural map of GeneAnalyst's architecture defining conceptual framework, pages and accompanying linkages and tools that make up the overall web application.

2.4.3.1. Development of the Front-End interface (http://portugalfishomics.ciimar.up.pt/app/geneanalyst/)

The home page or index page (Figure 16) is the first page a user encounters on accessing GeneAnalyst. This is the simplest page of the web application, simply displaying the logo and a drop-down menu allowing to pick the species the user is interested in. This was done on purpose since we did not want the user to be distracted from the aim of this page, which is to select a species.

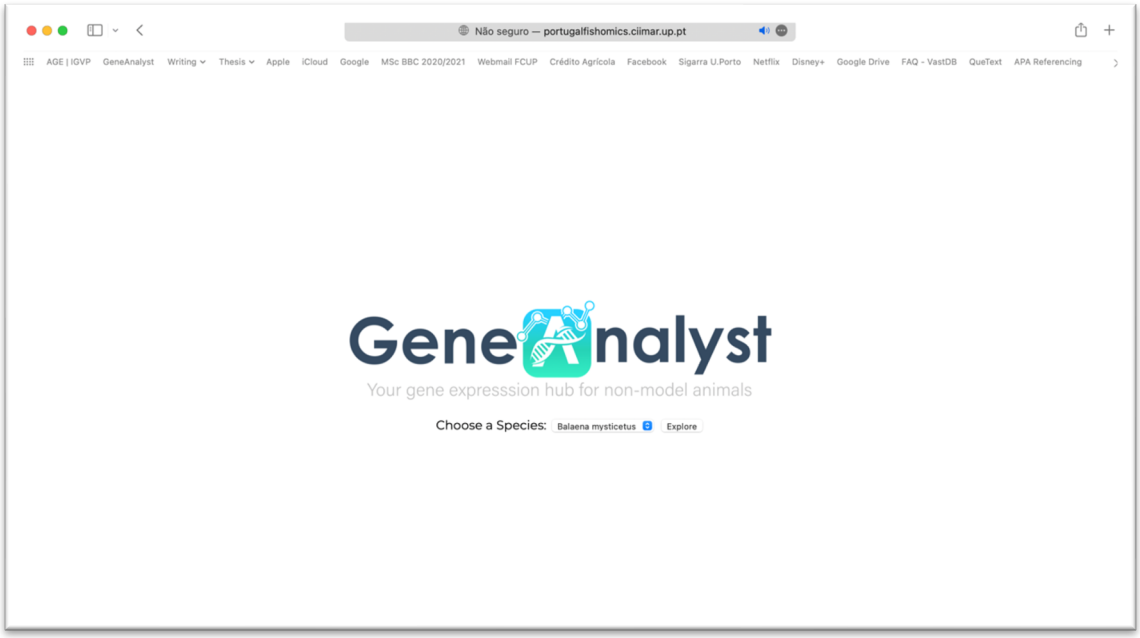


Figure 16 – GeneAnalyst's Home Page.

Within the HTML code, for each choice an identifier was assigned that would be transmitted to a PHP script that would, in turn, process the hyperlinkage to specific species pages. This strategy was used so that an automatic species introduction might be implemented in the future.

2.4.3.2. Creation of individual species pages

As previously mentioned, we created separate websites for each species added to GeneAnalyst. Because this page will access files within subfolders, some of which may be user-locked, these pages were retained in the application's root folder to facilitate navigation.

This page contains a top menu, a visually appealing photograph of the species accompanied by a brief description of the species, and several sections that are easily accessible via the top menu, which contains the implemented genome visualization tool,

the BLAST search form, and the gene expression visualization form. Figure 17 depicts the website for a selected model species, *C. milii*.

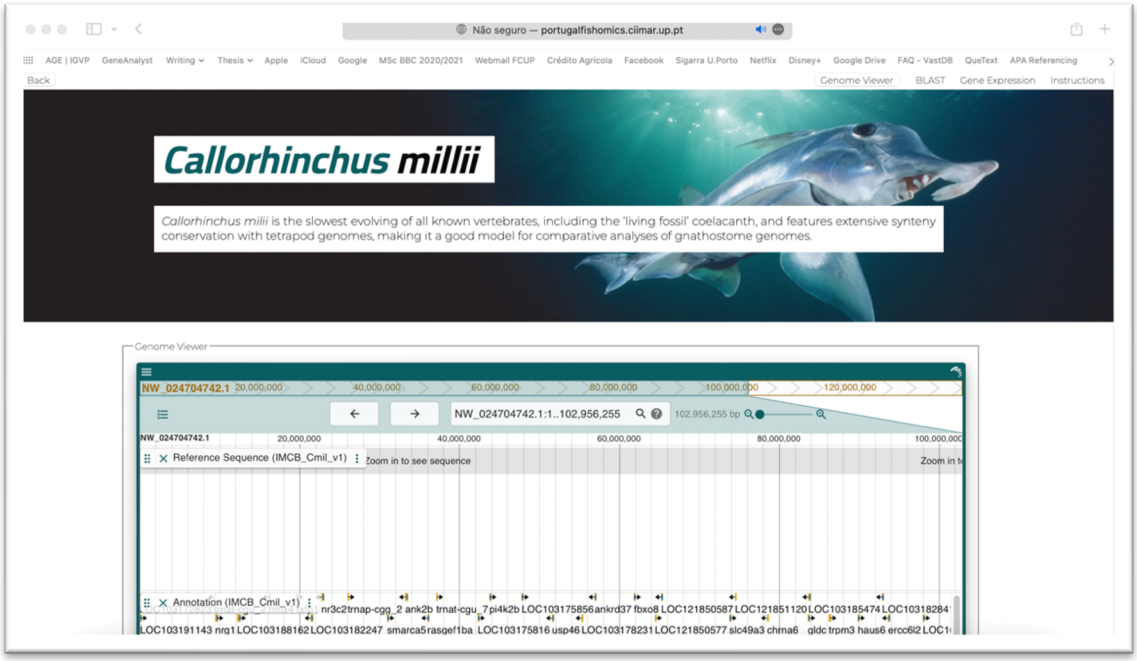


Figure 17 – *Callorhinchus milii* species page.

2.4.3.2.1 Implementation of the genome visualization tool

When working with a newly produced genome, some of the most common tasks to assess the quality of an assembly include searching for chimeric genomes and performing gene synteny. When performed manually, these tasks require a significant amount of digging through the annotation and FASTA sequences of the produced genome. Programs like IGV[85,86], Geneious (http://www.geneious.com/) and GenomeBrowser[87] may merge annotation files and FASTA sequences into a single, sophisticated display, simplifying these tasks.

One of the goals of this web application was to build an on-page display of these genomes stored on CIIMAR's high-performance computing system. JBrowse2[95], a genome browser built in the JavaScript programming language, was chosen because it not only offered the capability for embedding into web applications — allowing for the integration into a wide range of web applications architecture — but it also ran natively into one of the newest JavaScript frameworks and offered a wide variety of plugins that could permit future expansion.

JBrowse provided a command-line version that operated under the Node.js interpreter, which enabled the construction of this tool, as well as the building of an automatic

pipeline that would create the files required by JBrowse2 to implement an appealing visualization. This automation script combines samtools[116], JBrowse, and Python into a single pipeline that generates the appropriate files; the pipeline may be viewed in Code Snippet 2 below.

Code Snippet 2: Overview of the pipeline developed using samtools, JBrowse and Python for the creation of required files by JBrowse2. `generateJBrowseJSFiles.py` can be seen in Appendix C.

```
Input: FASTA file containing the genome sequence of a given species
       ($INFILE1) and the GFF file containing the annotation of said
       reference genome ($INFILE2).
Output: Required files by JBrowse2 for the implementation of the
       visualization.

for FASTA file inputted do
     compress the reference genome with:
      bgzip $INFILE1
     index the reference genome with:
      samtools faidx $INFILE1
     add assembly into JBrowse2 with:
      jbrowse add-assembly $INFILE1 --load inPlace
     Result: creation of "config.json"

for GFF file inputted do
     compress the annotation of reference genome with:
      bgzip $INFILE2
     index the annotation of reference genome with:
      tabix $INFILE2
     add track into JBrowse2 with:
      jbrowse add-track $INFILE1 -load inPlace
     text index the annotation into JBrowse2 with:
      jbrowse text-index --file $INFILE2
     Result: add track information into "config.json" and creation of
             "trix" folder allowing for gene search directly through
             the viewer

for config.json inputted do
     compute transformation of json file into two separate assembly
     and track js files:
      python3 generateJBrowseJSFiles.py config.json
```

The two files and text index folder created by the pipeline mentioned before were then imported into a JavaScript script that was immediately incorporated into the HTML

pages. In this script, we had to configure the visualization itself by setting the file paths and every single option according to our needs. We also aggregated text search adapters using `text-index` generated files and created a default session of the viewer for every species through the use of a JavaScript event listener function that could capture the state of the viewer at a given time, allowing us to set this state as the default visualization when the species was selected. Figure 18 depicts the implemented genome viewer in GeneAnalyst.

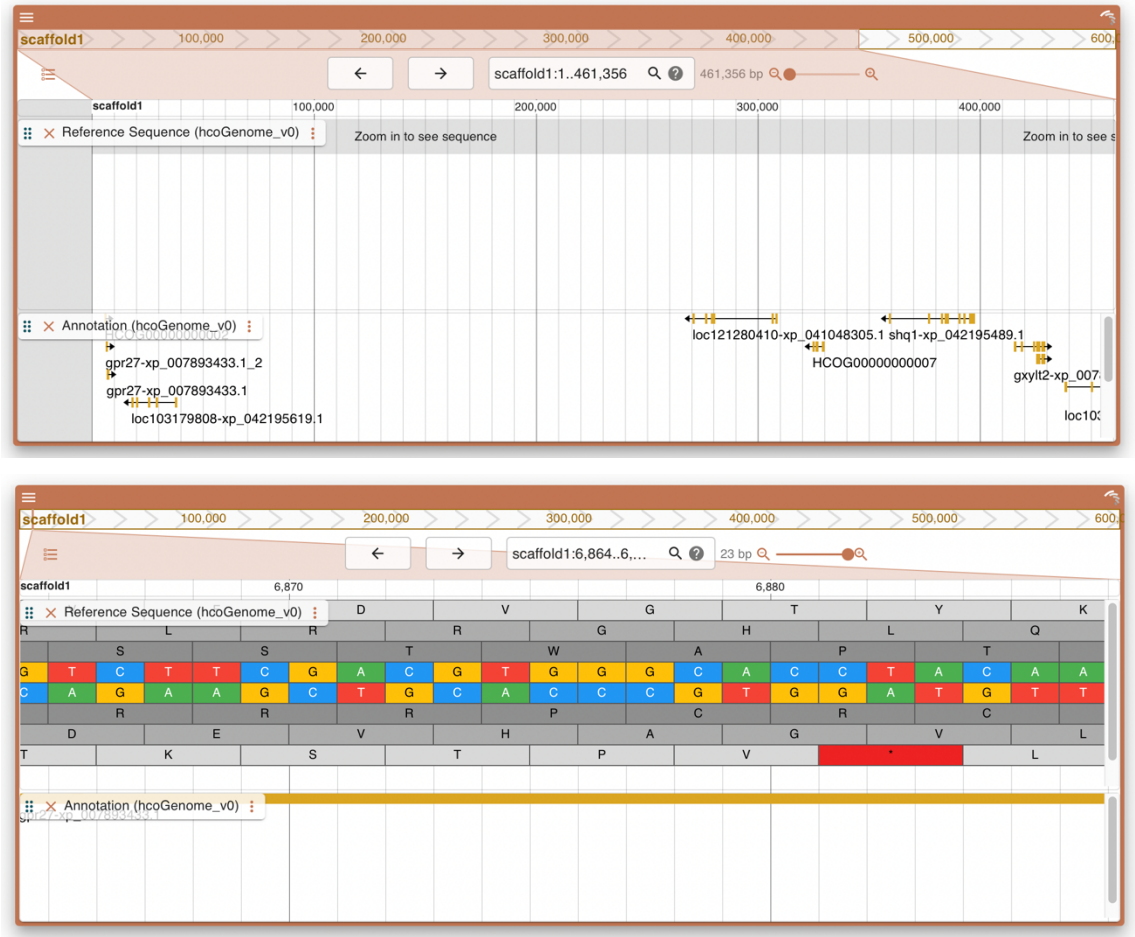


Figure 18 – *Hydrolagus colliei* GeneAnalyt's genome viewer segment. On top is an overview of scaffold 1, while below is a magnified look of the *Gpr27* gene and its associated nucleotide sequence.

2.4.3.2.2 Implementation of BLAST search

BLAST[92,93] is a commonly used algorithm with the capability of aligning and comparing query DNA or RNA sequences with a database of sequences, making it a crucial tool in current genomic and transcriptomic research because it can be used to infer functional and evolutionary links between sequences and find members of gene families; therefore, we could not forego implementing it.

The reference genome and annotation were used in conjunction with AGAT[94] v.0.9.2, whose functions are illustrated in Code Snippet 3, to build separate files containing the genes, mRNA, proteins, and CDS sequences in order to establish a database for each type of sequence, which was then utilized to generate the databases required for NCBI BLAST using the built-in `makeblastdb` function.

Code Snippet 3: Code schematic of actions performed utilizing FASTA and GFF files with AGAT and NCBI BLAST.

```
Input: FASTA file containing the genome sequence of a given species
       ($INFILE1) and the GFF file containing the annotation of said
       reference genome ($INFILE2).
Output: Individualized NCBI BLAST databases per sequence type.

for FASTA file inputted do
    extract specific type sequences, e.g., protein or mRNA with:
     agat_v0.9.2.sif agat_sp_extract_sequences.pl -g $ANNOTATION -f
     $GENOME -p $TYPE -o $TYPE.fasta
    Result: creation of FASTA file containing each type sequences

for FASTA file generated by AGAT do
    make database of each type of sequence with:
     if type sequence is nucleotide do
        makeblastdb -in $GENOME -parse_seqids -title "$SPECIES
        $TYPE_DESCRIPTION ($ACESSION)" -dbtype nucl
     if type sequence is proteic do
        makeblastdb -in $GENOME -parse_seqids -title "$SPECIES
        $TYPE_DESCRIPTION ($ACESSION)" -dbtype prot
    Result: creation of each type sequences database
```

The BLAST tool was then integrated into GeneAnalyst using HTML forms. These forms allow users to choose the alignment program they would like to use, e.g., BLASTP, paste a query FASTA format sequence or upload a query from a file, choose which available databases they would like to query, and input advanced parameters to tweak the BLAST search.

Some of the form's choices contained unique identifiers that allowed the entry to be provided to the `runBLAST` PHP script using PHP's POST function, allowing these options and the query sequence to be used as variables for processing on CIIMAR's high-performance computing system. Within this PHP script, the user-supplied settings were then placed on a command line, along with the right file directories that the tool was

to blast from, which were defined within the script and executed to obtain the results from BLAST (Figure 19).

As one of the goals of this thesis was to provide the ability to download the generated table displayed on the BLAST result page and select specific hits for FASTA sequence download, a Python script was developed. This script would parse through the original HTML result generated by NCBI BLAST modifying this page in order to accommodate new elements, such as new database information, a new table (produced from the outputted tabular file from NCBI BLAST for more appealing visualizations) and interactive elements (e.g., checkboxes for hit selection and download buttons, allowing for the export of tabular files and FASTA sequences of selected sequences). The full developed script can be seen in Appendix D.

Figure 19 – BLAST of *Psd3l* coding region in *Callorhinchus milii*.

2.4.3.2.3 Implementation of gene expression visualization

As indicated earlier, the HISAT2 software was used to map reads from RNA-Seq studies to a reference genome, and featureCounts used to quantify the mapped reads. This approach generated several files that were then processed using a R script to transform the raw read counts into FPKM and TPM gene expression values. In the end, we obtained several CSV files with a great deal of information that required intuitive visualizations.

In order to construct these visualizations, a new section was added to the displayed species page with the possibility to search for the expression of a certain gene or to search the CSV files for each tissue using a sophisticated table tool. This HTML form was then processed by a PHP script that gathered the options defined by the user and would conduct in either of two ways:

<u>If the user selected to analyze a tissue gene expression table (Figure 20):</u> the script would redirect the user to a page where he would be met with an interactive table generated by the CSV to HTML Table (https://github.com/derekeder/csv-to-html-table) which is fully constructed under JavaScript and makes use of the JQuery library allowing to filter and sort the table under specific queries by the user.

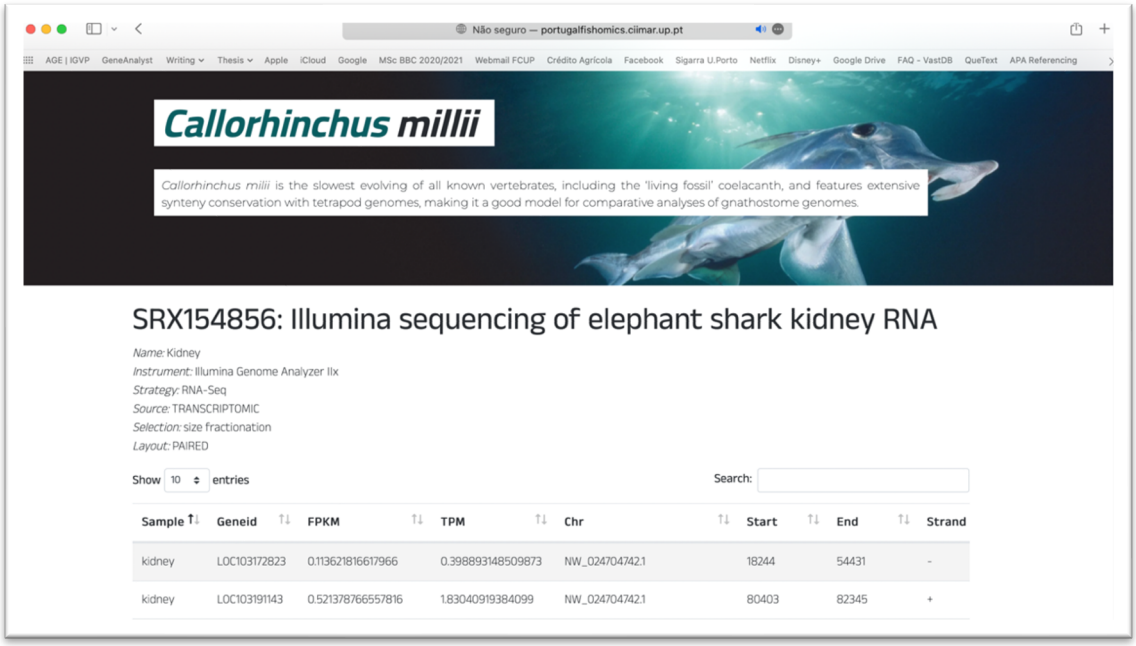


Figure 20 – Search by tissue option of kidney in *Callorhinchus milii* within GeneAnalyst's gene expression component.

<u>If the user queried a specific gene expression (Figure 21):</u> The script would utilize its command line functions. Firstly, the script scans the CSV files for the specified gene's expression values and after deploys two Python scripts (both scripts can be consulted in the Appendix E and F), to prepare and build an automated HTML page where a bar chart is generated for both FPKM and TPM measurements using Dimple.js.

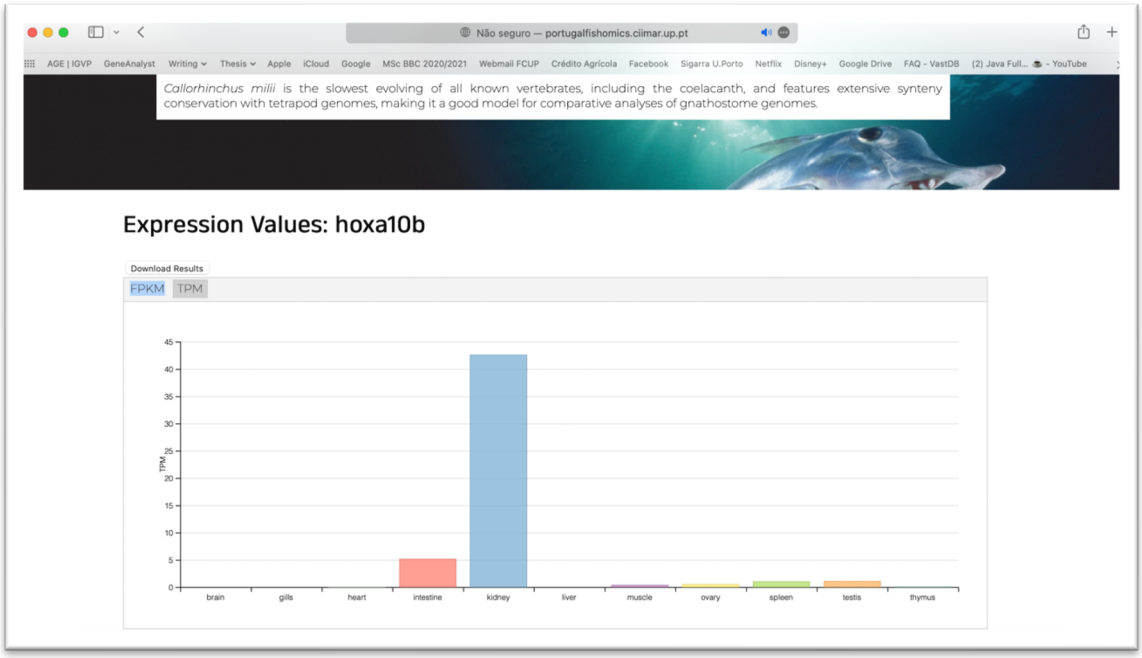


Figure 21 – *Hoxa10b* gene profiling retrieved using GeneAnalyst's gene expression segment in *Callorhinchus milii*.

Chapter 3

Results - GeneAnalyst's performance evaluation

3.1. Experimental design

To evaluate the performance of GeneAnalyst gene expression quantification process, we compared our findings with gene expression profiles from NCBI (https://www.ncbi.nlm.nih.gov/), and VastDB (https://vastdb.crg.eu/), in two distinct species. For each species, 100 genes were randomly selected for analysis utilizing the sample function within R.

In the first scenario, we examined *C. milii* gene expression profiles from publicly available RNA-Seq datasets for the IMCB Cmil v1.0 genome assembly in the NCBI database and compared our findings to some Real Time Polymerase Chain Reaction (RT-PCR) determined expression patterns available in literature.[117–119]

In summary, we used NCBI's Genome Data Viewer to manually analyze the exon expression variations for each individual gene within a log base 2 scaled, as it stabilizes the data variance of high intensities but increases the variance at low intensities, and compare it to our profiled gene expression. This approach allowed for a more efficient classification of a gene as expressed or not expressed, as NCBI did not provide gene expression measurements with this method.

The purpose of this evaluation was to determine how our gene expression profiles compared to those that had already been established. Yet, we compared them, by analyzing the exon expression behavior in the NCBI platform and comparing them to our results, based on the assumption that a gene is always expressed when its TPM value is greater than 1 (TPM > 1).

In the second instance, we examined gene expression profiles of 100 randomly selected genes, present in *G. gallus* genome, and compared them to expression patterns derived from RNA-Seq studies published in VastDB platform. In this case, we followed the earlier premise that a gene is expressed when its TPM value is higher than 1.

3.2. Comparative expression analysis

3.2.1. *C. milii* NCBI gene profilling

Regarding the NCBI gene profiling of *C. milii*, Table 2 illustrates the comparison of our results with the gene profiling available at NCBI. Of the total, 91.9% of the cases displayed the same expression pattern, *i.e.*, was expressed or not expressed with 7,8% of the cases failing to predict the same expression patterns; nevertheless, our pipeline identified 0.3% of occurrences where expression of the gene was shown despite not being included in NCBI's gene profiling. The full extent of the results can be consulted in Appendix G.

Table 2 – Evaluation of 100 randomly selected gene expression profiles in *Callorhinchus milii.* Newly Found Expression refers to the discovery of expression on our dataset as opposed to NCBI's gene profiling.

	***Callorhinchus milii* gene expression profiling of 100 randomly selected genes (%)**
Correctly Predicted	91.9
Wrongly Predicted	7.8
Newly Found Expression	0.3

3.2.2. GeneAnalyst versus RT-PCR-determined expression patterns in *C. milii*

3.2.2.1. Test Case 1: Evolution and Functional Characterization of Melanopsins in a Deep-Sea Chimaera

In the study published by *Davies et al.*[117], an RT-PCR-determined expression pattern of both isoforms of the *opn4* gene (*opn4m1* and *opn4m2*) and *opn4x* gene in elephant shark tissues was presented (Figure 22). These results indicate that the *opn4* gene was expressed in higher abundance in eye, liver, fin, and gills, while *opn4x* was more abundantly expressed in eye, gills, snout, and testis.

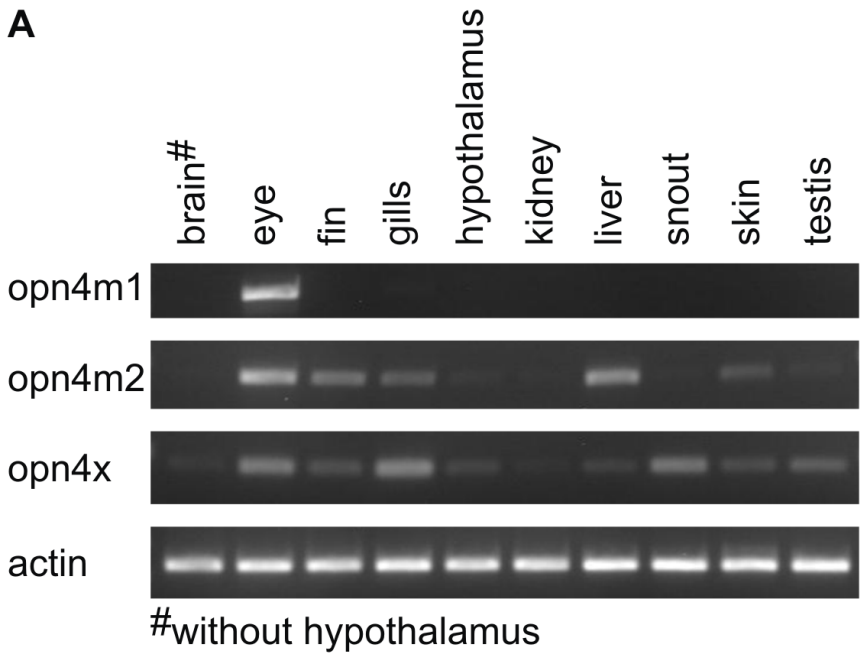


Figure 22 – *Davies et al.* RT-PCR-determined expression patterns of *opn4* and *opn4x* genes in *Callorhinchus milii.*

Even though we do not have the exact same panel of tissues, we nonetheless compared our results to the RT-PCR panel. As shown in Figure 23, regarding the *opn4* gene, although the TPM values do not exceed one, suggesting that they are not expressed or have low expression in some tissues, we still can observe patterns in the liver, gills, kidney, and testis comparable to the RT-PCR analysis. In contrast to the RT-PCR pattern (as it seems that both genes are not expressed or have low expression values in brain) we have a comparable abundance in the gills and brain, which might be explained by the fact that the brain sample utilized in our RNA-Seq experiment could have included the hypothalamus and some faint expression can be observed in the hypothalamus fraction of the *opn4m2* isoform in Davies' study (Figure 22).

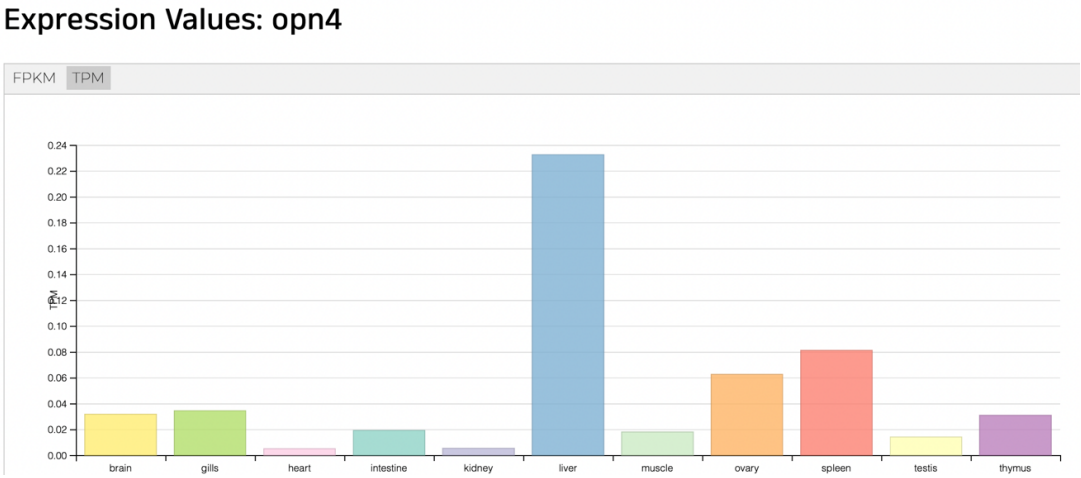


Figure 23 – Gene expression profiling for *opn4* gene in *Callorhinchus milii* retrieved within GeneAnalyst.

Further, when we compare the gene expression pattern of opn4x to our results, we notice a very similar pattern, as illustrated in Figure 24: with a high abundance of expression in the gills and testis, consistent with Davies' findings (Figure 22).

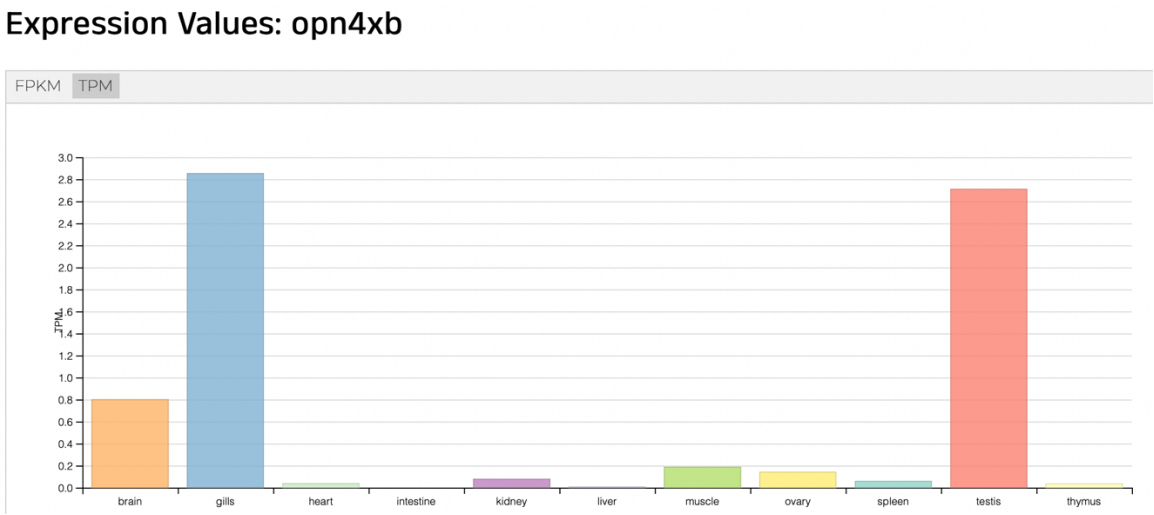


Figure 24 – Gene expression profiling for *opn4xb* gene in *Callorhinchus milii* retrieved within GeneAnalyst.

3.2.2.2. Test Case 2: Evolutionary Plasticity in Detoxification Gene Modules: The Preservation and Loss of the Pregnane X Receptor in Chondrichthyes Lineages

The nuclear receptor Pregnane X Receptor (PXR) is present in the genome of the elephant shark and the expression of this gene was previously assayed by RT-PCR (Figure 25).[118] The mentioned study shows a discernible expression in the skin, with a more subtle expression in the gills.

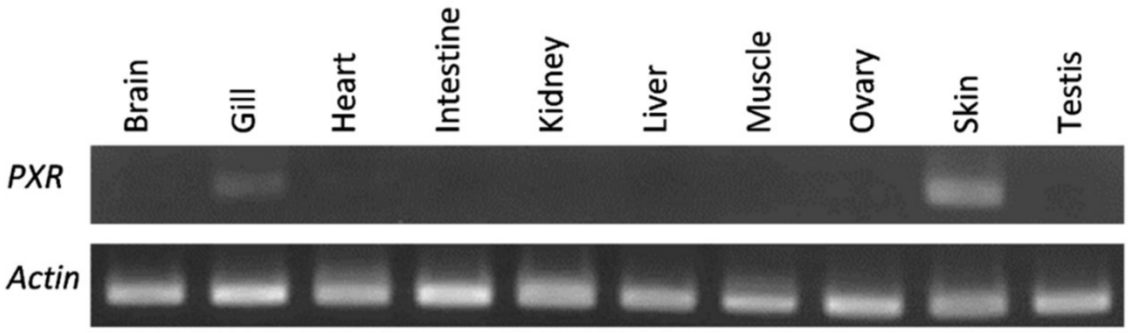


Figure 25 – *Fonseca et al.* RT-PCR-determined expression patterns of the *PXR* gene in *Callorhinchus milii*.

The expression analysis with the GeneAnalyst tool showed a very significant expression in gills (Figure 26). We also detected expression in the spleen, although this tissue does not appear in the original panel of Fonseca *et al*. [118] A skin sample was not included in our test samples.

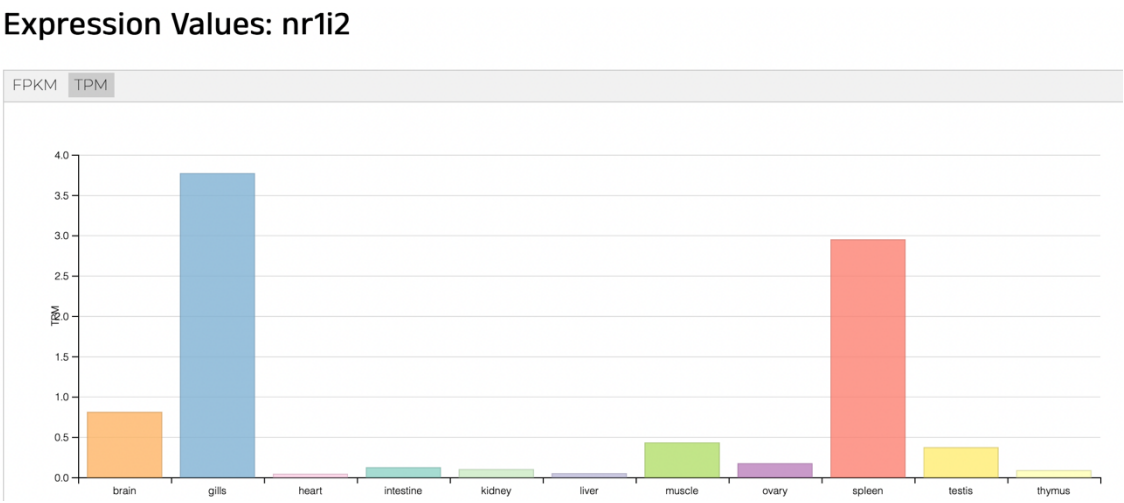


Figure 26 – Gene expression profiling for *PXR* gene in *Callorhinchus milii* retrieved within GeneAnalyst.

3.2.2.3. Test Case 3: Sequencing of Pax6 Loci from the Elephant Shark Reveals a Family of Pax6 Genes in Vertebrate Genomes, Forged by Ancient Duplications and Divergences

Ravi et al.[119] described the *Pax6* gene diversity and loci. This gene is a development regulatory gene required for eye development. In the elephant shark two gene paralogues were found and the RT-PCR-determined tissue panel showed a highly specific pattern. Pax6.1 and Pax6.2 are expressed in brain, eye, and pancreas; and eye and kidney respectively (Figure 27).

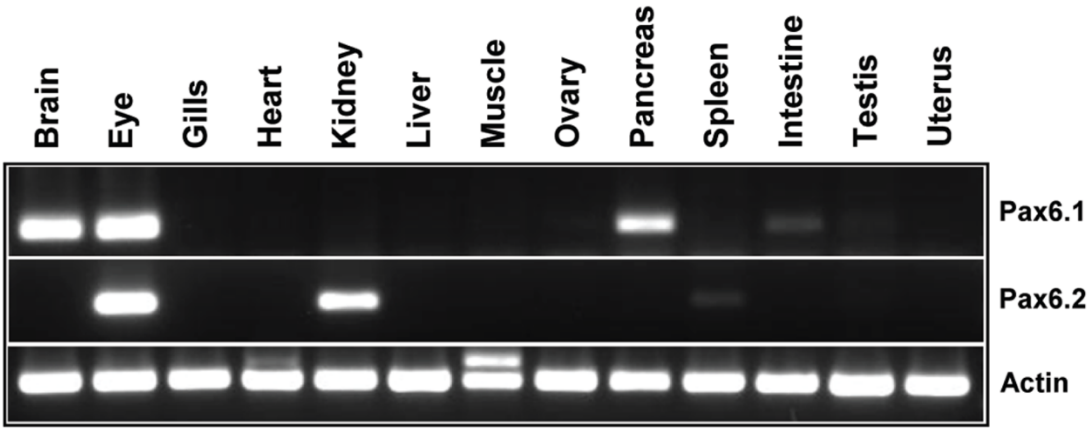


Figure 27 – *Ravi et al.* RT-PCR-determined expression patterns of *Pax6.1* and *Pax6.2* genes in *Callorhinchus milii.*

Our independent analysis, using the GeneAnalyst tool shows a similar output in the expression pattern (Figure 28).

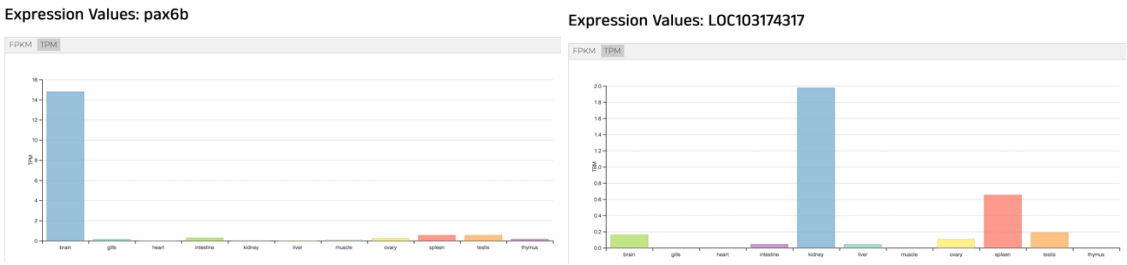


Figure 28 – Gene expression profilings for *Pax6.1* (on the left) and *Pax6.2* genes (on the right) in *Callorhinchus milii* retrieved within GeneAnalyst.

3.2.3. Comparative analysis with VastDB: the case of *G. gallus* gene profiling

Similarly to the methodology used with *C. milii* we compared our findings to 100 randomly chosen gene profiles accessible at the VastDB utilizing RNA-Seq data; the results of this comparison are shown in Table 3 (the full scope of this comparison can be consulted in Appendix H).

Table 3 – Evaluation of 100 randomly selected gene expression profiles in *Gallus gallus.* Newly Found Expression refers to the discovery of expression on our dataset as opposed to NCBI's gene profiling.

	***Gallus gallus* gene expression profiling of 100 random selected genes (%)**
Correctly Predicted	77.3
Wrongly Predicted	19.3
Newly Found Prediction	3.3

In our findings, we observed that 77.3% of gene expression was accurately predicted, while 19.3% did not match exactly the prediction presented in VastDB, with 3.3% of the results showing a pattern contrary to what VastDB stated.

3.3. Evaluation Discussion

Linking functional and phenotypic information from RNA-Sequencing data requires the careful assessment of quantification measurements. This is particularly relevant considering the vast amounts of expression data being generated daily. While species-specific dedicated hubs have been developed (e.g., Human Protein Atlas), the vast majority while available at standard repositories as raw data, do not provide accessible and easy to use interfaces. The intrinsic biological and research value of RNA-Seq datasets is critical to decipher for example tissue function or validating gene models and isoform presence (alternative splicing). Thus, the development of GeneAnalyst was

driven by the need for an effective, user-friendly, and efficient method to examine created assembled genomes, query these genomes, and query RNA-Seq datasets in an easy and efficient approach, without requiring programming knowledge, hence facilitating an improved workflow.

Although the scientific community has not yet reached a consensus regarding the optimal gene expression quantification method for these procedures[53], here we employed the assumption that a gene is always expressed when its TPM value is greater than 1 (TPM > 1), as the purpose of this evaluation was to determine how our gene expression profiling compared to previously established methodologies.

To validate the pipeline, we conducted an exhaustive comparative analysis described above. When comparing our results to NCBI's gene profiling for *C. milii*, we obtained a reasonable performance, with 91.9% of genes accurately classified as expressed, utilizing our established premise to compare these profiling. Analogously, GeneAnalyst correctly predicted expression when compared to conventional RT-PCR-determined expression patterns, accurately predicting expression in the *opn4xb*, *PXR*, *Pax6.1*, and *Pax6.2* genes, only failing to predict the shown expression for the *opn4* gene.

Our accuracy dropped when we compared our findings with the VastDB's *G. gallus* gene profiling: achieving only 77.3% of correctly categorized expressed genes. The basis for this misfit could be explained by the fact that VastDB's authors used an RPKM in single-end RNA-Seq data, instead of the FPKM or TPM (https://vastdb.crg.eu/wiki/FAQ), which could influence the expression quantification, as the SRA files utilized corresponded to paired-end RNA-Seq data. Moreover, the genome assembly used was different, with GeneAnalyst *Gallus gallus*' gene expression profiles being predicted using a more recent version of the genome assembly (`bGalGal1.mat.broiler.GRCg7b`) rather than VastDB's assembly (`galGal4`). In addition, more genes seemed to be annotated when we first compared both versions of the genomes, this difference could also contribute to the discrepancy of inferences, as a higher number of transcripts will be inversely proportional to TPM values (Equation 2).

Chapter 4

Integrating a genome browser with gene expression profiles in non-model species: a test case

4.1. Introduction

To further test the functionality of the developed hub including the GeneAnalyst application, we used a set of genes involved in the evolution gastric/intestinal function in vertebrates.[120] The selected genes typically display a very tissue restricted gene expression pattern along the digestive tube (Human Protein Atlas) and have an unknown phylogenetic distribution.

4.2. The gene TRIM50

Tripartite Motif Containing 50 or TRIM50 is a protein coding gene predicted to be involved in protein ubiquitination.[121] Currently, TRIM50 has 145 orthologs according to Ensembl (https://www.ensembl.org), but no gene orthologs have been found in *C. milii*; hence, we utilized it as a negative control for *C. milii* and *H. colliei*. As TRIM50 is not present in the annotation, we utilized the largest single-nucleotide variant of the *Homo sapiens* TRIM50 gene (NM_178125.3) to BLAST GeneAnalyst's databases for both species and, as anticipated, there was no significant hit (Figure 29).

```
Query= NM_178125.3:202-1665 Homo sapiens tripartite motif containing 50
(TRIM50), transcript variant 1, mRNA

Length=1464


***** No hits found *****



Lambda      K        H
    1.33    0.621     1.12

Gapped
Lambda      K        H
    1.28    0.460    0.850

Effective search space used: 1639687618350


  Database: Callorhinchus millii mRNA sequences on IMCB_Cmil_1.0
    Posted date:  Aug 13, 2022  6:14 PM
  Number of letters in database: 1,144,486,875
  Number of sequences in database:  35,020
```

Figure 29 – BLAST result of TRIM50 gene query in *Callorhinchus milii*'s genome displaying no significant hits.

4.3. KCNE2

Potassium Voltage-Gated Channel Subfamily E Regulatory Subunit 2 or KCNE2 is a protein coding gene that assembles as a beta subunit that modulates the gating kinetics and enhances the stability of the channel complex.[122]

This gene counts with 164 orthologs in the Ensembl database and similarly to the previous case study of the TRIM50 gene, no ortholog can be found in Ensembl for *C. milii*; therefore, we used this gene as a negative control for *C. milii and H. colliei*, as previously done.

The query sequence used in BLAST GeneAnalyst's databases for both species was the largest single-nucleotide variant of the *H. sapiens* KCNE2 (NM_172201.2) as KCNE2 was not present in the annotation. Using this sequence as the query search yielded no significant hit (Figure 30), as expected.

```
Query= NM_172201.2:159-530 Homo sapiens potassium voltage-gated channel
subfamily E regulatory subunit 2 (KCNE2), mRNA

Length=372


***** No hits found *****



Lambda      K        H
    1.33    0.621     1.12

Gapped
Lambda      K        H
    1.28    0.460    0.850

Effective search space used: 393366172360


  Database: Callorhinchus millii mRNA sequences on IMCB_Cmil_1.0
    Posted date:  Aug 13, 2022  6:14 PM
  Number of letters in database: 1,144,486,875
  Number of sequences in database:  35,020
```

Figure 30 – BLAST result of KCNE2 gene query in *Callorhinchus milii*'s genome displaying no significant hits.

4.4. Isx

Specific to the intestine, Isx, or Intestine Specific Homeobox, is a gene that codes for a transcription factor that controls gene expression in the intestine and is believed to be crucial in early embryonic development.[123]

C. milii ortholog can be found among the 106 orthologs genes identified in the Ensembl database (ENSCMIG00000019885). The first step was to use GeneAnalyst's Genome Viewer tool to determine whether this gene was already annotated in the most recent published annotation; no annotated genes were found, so we began with blasting (blastn) the Ensembl's ortholog CDS sequence directly to *C. milii*'s CDS BLAST database, which

is available on our species page, with the default parameters. There was one hit (LOC103181849) with 100% identity, 0.0 e-value, and 1275 bit score.

To confirm that this was the gene we were looking for in our annotation, we examined the synteny and compared it to the Ensembl data (Figure 31).

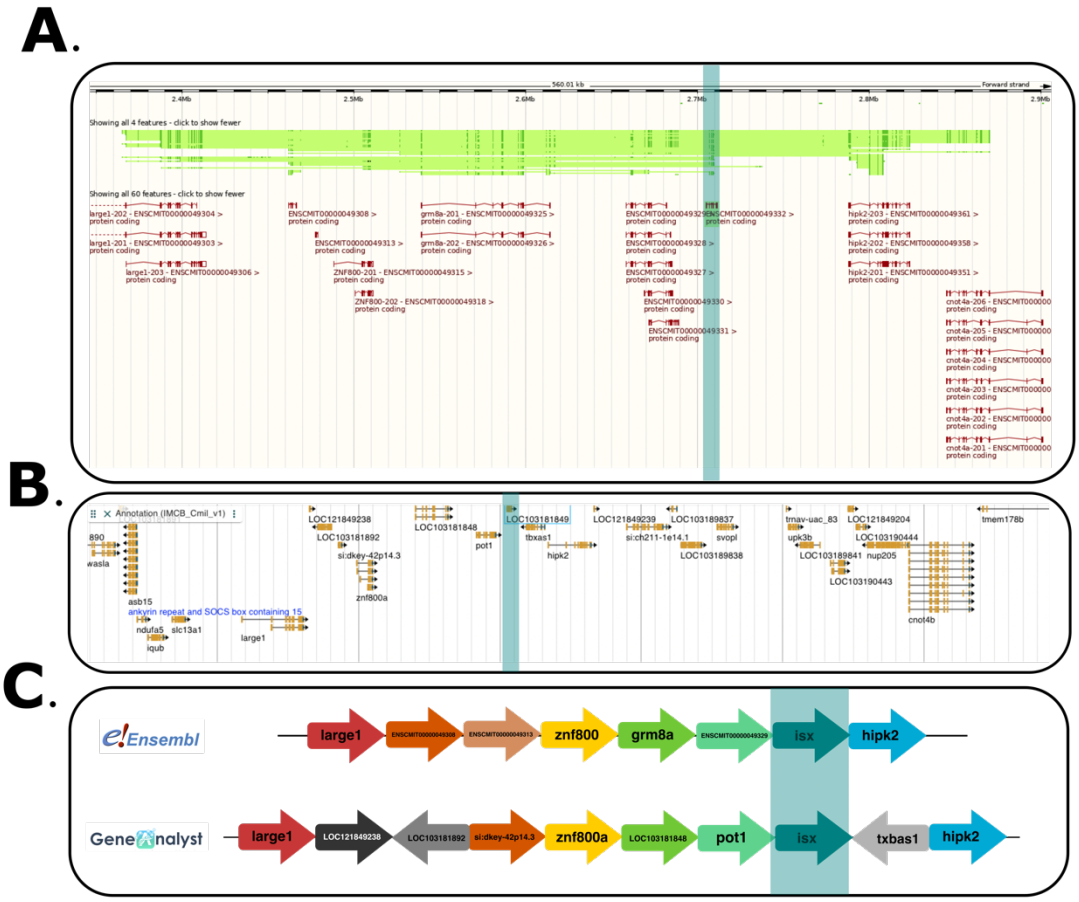


Figure 31 – Ensembl and GeneAnalyst Isx gene synteny comparison. (A) Ensembl's gene synteny (B) GeneAnalyst's gene synteny. (C) Illustration representing the synteny comparison between the two, the genes were paired-match colored and grayed out genes represent found additions in GeneAnalyst's annotation. The green highlighted region reflects the *Isx* gene of interest.

While comparing the syntenies, we saw that most of the Ensembl genes are present in our version of the genome, apart from ENSCMIT00000049313 for which we did not find any significant hits during the Blast search. We can also see that three genes have been annotated in the IMCB's version of the genome but are absent in Ensembl's version. Due to the accuracy of this comparison, we may conclude that this is the true ortholog we were searching.

Having identified our target gene in our assembly, we further continued to analyze its expression using GeneAnalyst's gene expression visualization tool. As shown in Figure

32, the elephant shark *Isx* gene orthologue is expressed in the gut, as expected given that this gene is known to be specifically expressed the intestine.[123]

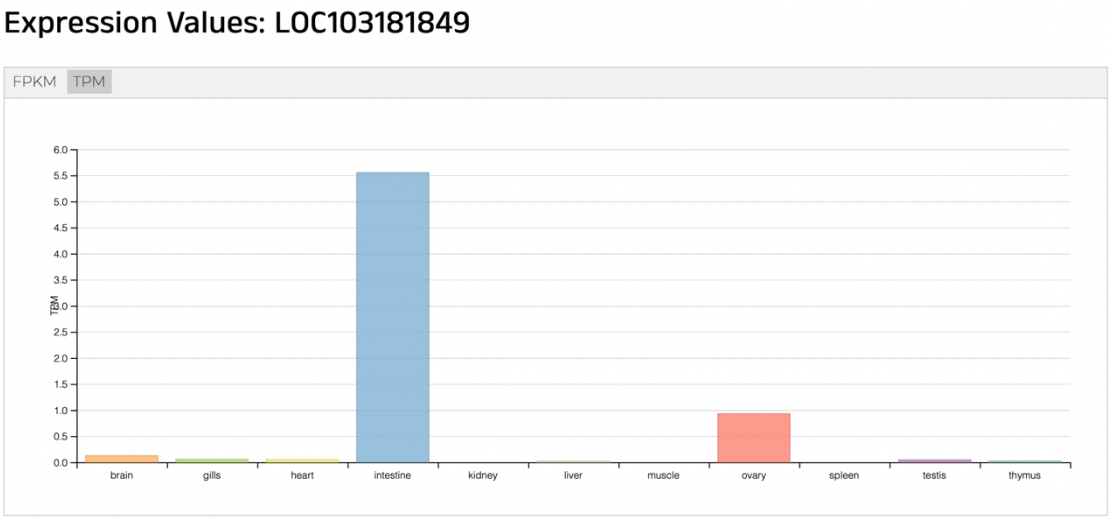


Figure 32 – Gene expression profiling for the *Isx* gene in *Callorhinchius milii* retrieved within GeneAnalyst.

We also conducted a similar search for this gene in a genome assembly of *H. colliei* (in-house produced). Initially, we extracted *C. milii*'s *Isx* gene sequence directly from the GeneAnalyst Genome Viewer and used it as our query against *H. colliei* CDS BLAST database with the default parameters, which returned one significant hit (loc103181849-xp_042191363.1) with 91% identity, 0.0 e-value, and 917 bit score.

C.milii Isx syntenic region is mostly identical to that of the *H. colliei* gene (Figure 33). However, the LOC121849238 appears to be absent in *H. colliei* assembly, while an additional gene, represented by LOC123481373, is predicted in the *H.* colliei syntenic region; based on the accuracy of this comparison, we can conclude *H.* colliei gene is an ortholog of Isx and continued with the analysis of its expression using the gene expression visualization tool within GeneAnalyst.

As shown in Figure 34, we can see that the gene appears to have expression in the Spiral Valve, a lower portion of the intestine of some sharks, showing its tissue specificity.

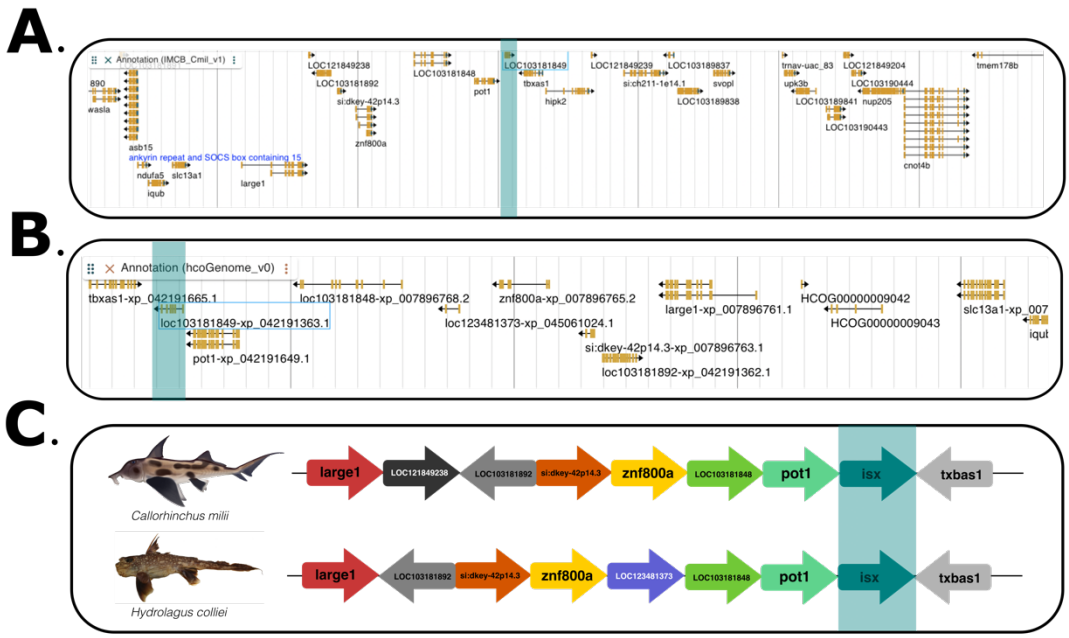


Figure 33 – *Callorhinchus milii's* and *Hydrolagus colliei's Isx* gene synteny comparison. (A) *Callorhinchus milii's* gene synteny (B) *Hydrolagus colliei's* gene synteny. (C) Illustration representing the synteny comparison between the two, the genes were paired-match colored and the purple represent found additions in the spotted ratfish's annotation. The green highlighted region reflects the *Isx* gene of interest.

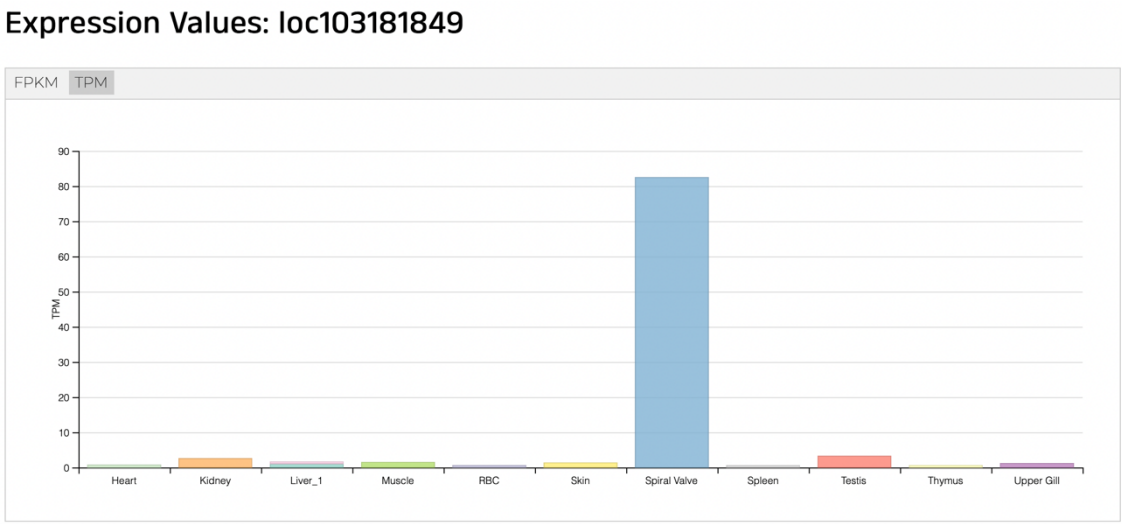


Figure 34 – Gene expression profiling for the *Isx* gene in *Hydrolagus colliei* retrieved within GeneAnalyst.

4.5. Pdx1

Similarly to the prior test case with the *Isx* gene, we ran a similar investigation with the Pancreatic and Duodenal Homeobox 1 (*Pdx1*) gene. This is a transcription factor found in the ParaHox gene cluster and is essential for pancreatic development in vertebrates.[124,125] This transcription factor is an activator of multiple genes, e.g., insulin, somatostatin and glucokinase, involved in the early development of the pancreas playing a crucial role in glucose-dependent control of insulin gene expression.[124,125]

When looking for orthologs on PDX1 in *C. milii*, none was discovered; nevertheless, *Pdx1* seemed to be annotated in the genome and was automatically predicted using the Gnonom (annotation software) prediction approach. Due to the possibility that an orthologue had yet to be manually curated, we proceeded with the study, this time using the synteny of the gene in *H. sapiens*. The results are shown in Figure 35. We were able to match almost the full annotation during the synteny analysis, except for *polr1d*, which was assigned to a different scaffold, but having found a smaller gene (*si:ch211-140b10.6*) on its original position, and *cdx2*.

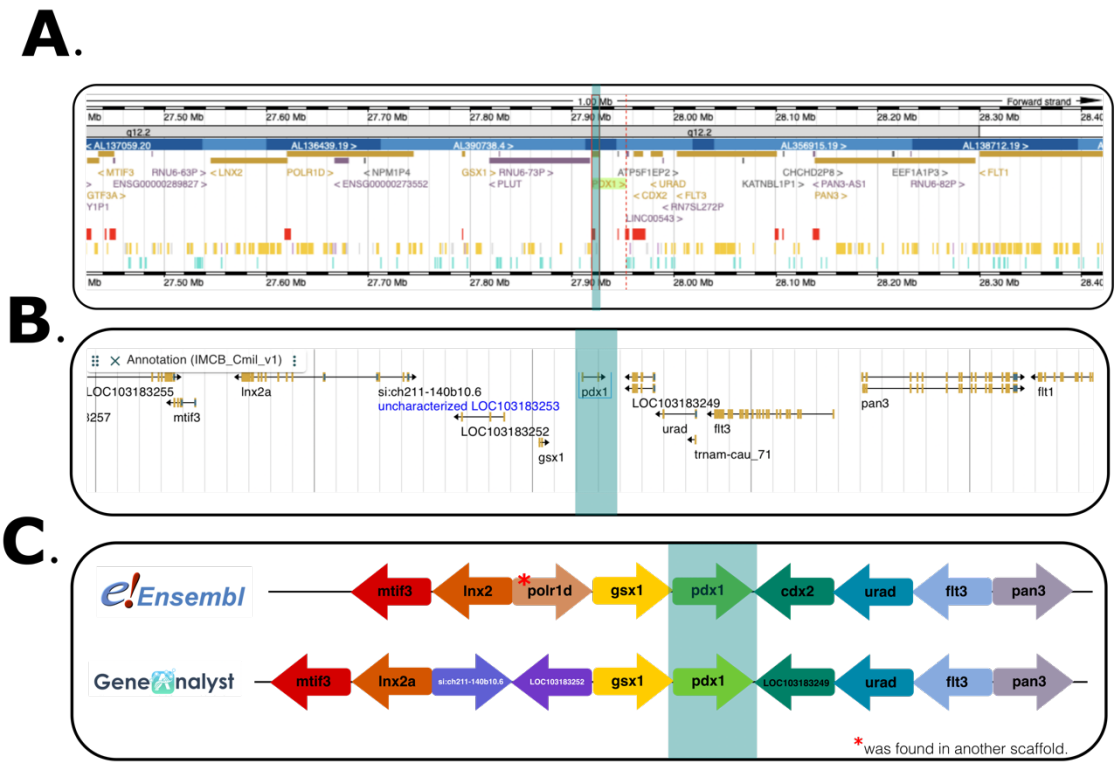


Figure 35 – Ensembl and GeneAnalyst *Pdx1* gene synteny comparison. (A) Ensembl's gene synteny (B) GeneAnalyst's gene synteny. (C) Illustration representing the synteny comparison between the two, the genes were paired-match colored and purple genes represent found additions in GeneAnalyst's annotation. The green highlighted region reflects the *Pdx1* gene of interest.

Having identified our target gene in our assembly, we analyzed its expression using GeneAnalyst's gene expression visualization tool. As shown in Figure 36, we did not find gene expression values above of 1 TPM, which was expected as our tissue panel did not integrate pancreas.

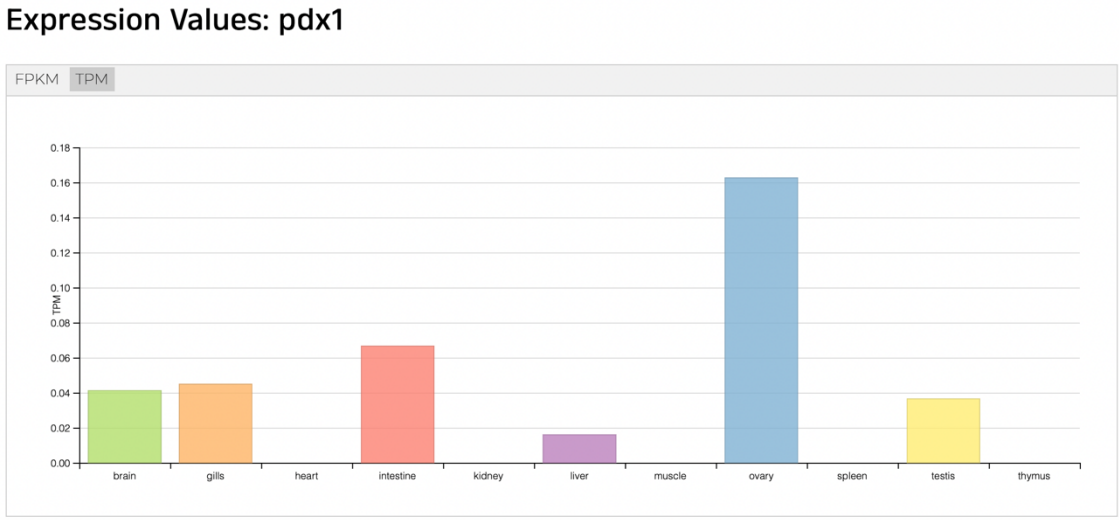


Figure 36 – Gene expression for the *Pdx1* gene for *Callorhinchius milii*, shown in GeneAnalyst's gene expression visualization tool.

Regarding *Pdx1* in *H. colliei*, the observed gene synteny appeared to have been mostly conserved, as depicted in Figure 37. However, two new genes HCOG000018115 and HCOG000000181 were annotated with LOC103183252 not being present in the syntenic region. Based on the accuracy of this comparison, we can conclude that the retrieved gene is a *Pdx1* ortholog and continued with the analysis of its expression using the gene expression visualization tool within GeneAnalyst.

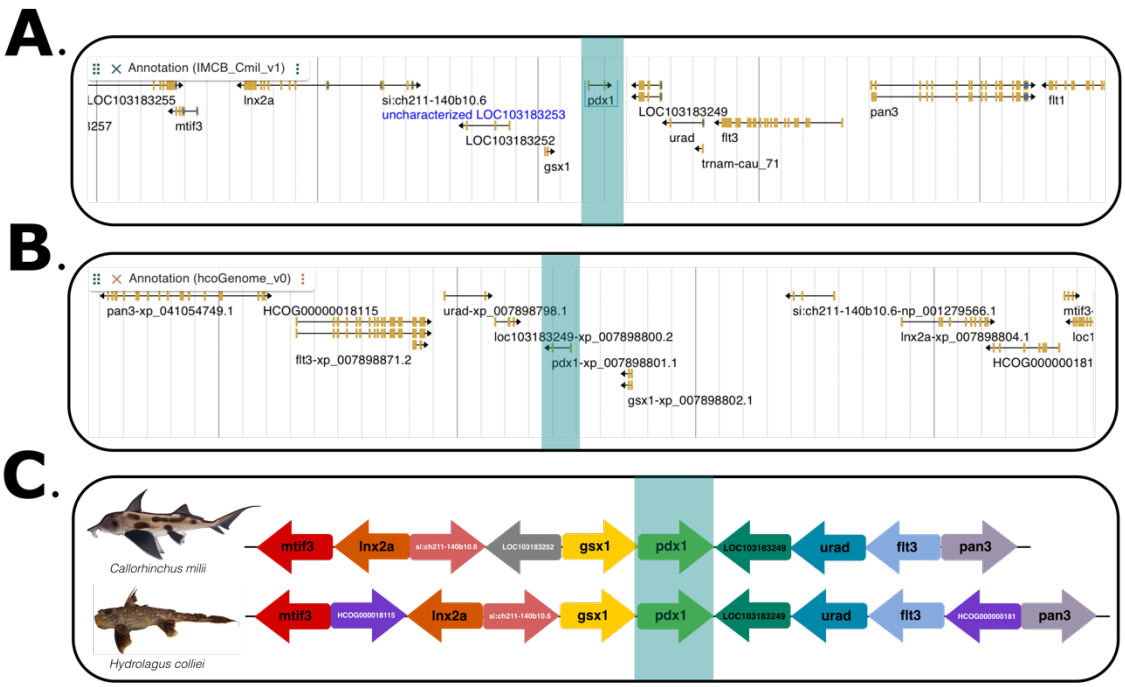


Figure 37 – *Callorhinchus milii's* and *Hydrolagus colliei's Pdx1* gene synteny comparison. (A) *Callorhinchus milii's* gene synteny (B) *Hydrolagus colliei's* gene synteny. (C) Illustration representing the synteny comparison between the two, the genes were paired-match colored, and the purple represent found additions in the spotted ratfish's annotation. The green highlighted region reflects the *Pdx1* gene of interest.

As shown in Figure 38, *Pdx1* appears to be represented in the Spiral Valve, a lower portion of the intestine of some sharks, however it did not reach our expression threshold.

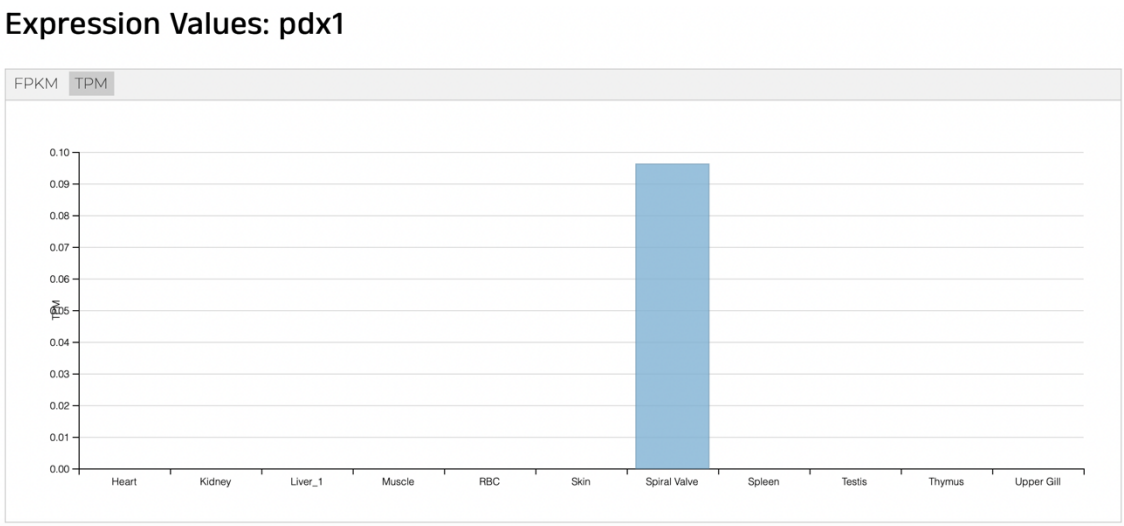


Figure 38 – Gene expression for the *Pdx1* gene for *Hydrolagus colliei*, shown in GeneAnalyst's gene expression visualization tool.

Overall, the tested cases provide a clear application value of the GeneAnalyst hub platform.

General Discussion

GeneAnalyst aimed to alleviate some of the obstacles for the analysis and investigation of gene expression by implementing a pipeline that would generate gene expression quantification based on RNA-Seq data, while also incorporating a robust genome visualization component and BLAST tools: to enable users to efficiently analyze and query different datasets generated for both validated animal model species and non-validated species, without the need for heavy file management or coding knowledge. GeneAnalyst was assessed on two validated model species, *C. milii* and *G. gallus*, and was subsequently deployed on a non-model species, *H. colliei*.

Regarding to the computational steps, our attention was particularly drawn to the mapping of RNA-Seq reads to the genome. The results displayed on Appendix A reveal a reasonable performance when working with *C. milii* but dropped when working with *G.gallus* and *H.colliei*. To reconcile this loss of performance, perhaps an additional step of read trimming could be in order to filter low quality reads and trim adapters if needed.

JBrowser, a genome browser application written in the JavaScript programming language, was used to create a genome visualization component that enabled "right-on-page" visualization of the genome, as well as the retrieval of genomic sequences directly through the application, without the need to handle the core omic files of these species. Future implementations of plugins could be developed or deployed to display other filetypes, such as SAM and BAM alignment files, in the future.

In addition, we included a BLAST tool component to every species page so that users may rapidly query these datasets without leaving our hub, so making our application quicker and more convenient.

Similarly, to web applications discussed in Chapter 1, such as FX and rQuant.web, GeneAnalyst promised to mitigate some of the obstacles of RNA-Seq gene quantification by providing a way for technicians with diverse backgrounds, ranging from biologists to bioinformaticians, to easily, quickly and efficiently retrieve gene expression profiling generated from RNA-Seq data. Unlike the mentioned applications, GeneAnalyst offered these features without the need for heavy file management or code knowledge from the user's point-of-view. GeneAnalys, as rQuant.web can have an advantage over web because it allows for the import of data directly from the source, *i.e.,* allowing the user to input a list of SRA accession numbers that would be downloaded internally. Yet, due to rQuant.web's 2GB upload limit, we were unable to use the elephant shark's dataset to compare it to our application.

In addition, while the user is not required to have extensive file management or coding skills, the process behind GeneAnalyst still requires human involvement since it is not yet entirely automated, owing to some constraints in the HPC internal arrangement that have prevented the implementation of its full automation. This manual intervention entails downloading the reference genomes, its related annotation, and the SRA archives, as well as carrying out the appropriate mapping. After that, automatic scripts will add the species straight to the hub's internal system by executing the necessary actions, such as, alterations in the HTML source code.

Future possibilities would require refining in the HPC interactions to enable users to submit their data directly to GeneAnalyst in order to acquire gene profiling. Additionally, the visualizations of gene expression might be modified to allow for greater user engagement, and additional visualizations such as instructive heatmaps could be supplied.

Regarding GeneAnalyst's performance in the inference of gene profiling, we were satisfied with the findings recovered, as it successfully predicted the gene expression patterns of the majority of our test cases when comparing to published RT-PCR-determined gene patterns and RNA-Seq generated gene profiling available, as described before, in a very quick manner and with little computing costs.

Conclusion

A new technique for profiling gene expression based on next-generation sequencing (NGS), referred commonly as RNA-Seq, has been replacing microarrays over several advantages: including a high level of reproducibility and reduction of the number of technical replicates required for experiments, when compared to its predecessor, if combined with informatic and statistical methods.[18]

There is a growing desire for easy, intuitive, and user-friendly procedures that enable researchers and workers with less bioinformatic expertise to access gene expression studies.[66] These methodologies aim to provide accurate and easily interpretable results[66,67], while keeping pace with the development of sequencing technologies.[66,68]

Some bioinformatics platforms offer easy and quick-to-access visualizations of RNA-Seq quantification datasets: but such platforms lack flexibility notably regarding the visualization of user-inputted non-model species' data.

GeneAnalyst aims to overcome these limitations by providing an application that could be applied to any desired species, validated model species or not, allowing users to quickly visualize the annotation of a given genome, query through the genome using a BLAST function, and quickly and intuitively query and visualize RNA-Seq datasets with minimal or no coding requirements.

Our application was evaluated on two validated model species, *C. milii* and *G. gallus*, and showed satisfactory results, as it accurately predicted the gene expression patterns of most of our test cases when compared to published RT-PCR-determined gene patterns and RNA-Seq generated gene profiling available in a very quick manner and with minimal computing costs. It has also been applied to *H. colliei*, an in-house generated dataset, in which we were able to demonstrate the potential of GeneAnalyst to be applied to novel datasets as novel genomes are being produced and will be used for comparative genomic studies.

References

1. Frankham, R., Ballou, J. D. & Briscoe, D. A. *Introduction to conservation genetics*. (Cambridge University Press, 2010).

2. Watson, J. D. & Crick, F. H. C. Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid. *Nature* **171**, 737–738 (1953).

3. Sanger, F. & Thompson, E. O. P. The amino-acid sequence in the glycyl chain of insulin. 1. The identification of lower peptides from partial hydrolysates. *Biochem. J.* **53**, 353–366 (1953).

4. Sanger, F. & Thompson, E. O. P. The amino-acid sequence in the glycyl chain of insulin. 2. The investigation of peptides from enzymic hydrolysates. *Biochem. J.* **53**, 366–374 (1953).

5. Giani, A. M., Gallo, G. R., Gianfranceschi, L. & Formenti, G. Long walk to genomics: History and current approaches to genome sequencing and assembly. *Comput. Struct. Biotechnol. J.* **18**, 9–19 (2020).

6. Metzker, M. L. Sequencing technologies — the next generation. *Nat. Rev. Genet.* **11**, 31–46 (2010).

7. Mardis, E. R. A decade's perspective on DNA sequencing technology. *Nature* **470**, 198–203 (2011).

8. Kulski, J. K. Next-Generation Sequencing — An Overview of the History, Tools, and "Omic" Applications. *Next Generation Sequencing - Advances, Applications and Challenges*n (2016)

9. Ekblom, R. & Wolf, J. B. W. A field guide to whole-genome sequencing, assembly and annotation. *Evol. Appl.* **7**, 1026–1042 (2014).

10. Dominguez Del Angel, V. *et al.* Ten steps to get started in Genome Assembly and Annotation. *F1000Research* **7**, 148 (2018).

11. Ellegren, H. Genome sequencing and population genomics in non-model organisms. *Trends Ecol. Evol.* **29**, 51–63 (2014).

12. Romanov, M. N. *et al.* The value of avian genomics to the conservation of wildlife. *BMC Genomics* **10**, S10 (2009).

13. Ejigu, G. F. & Jung, J. Review on the Computational Genome Annotation of Sequences Obtained by Next-Generation Sequencing. *Biology* **9**, 295 (2020).

14. Phillippy, A. M. New advances in sequence assembly. *Genome Res.* **27**, XI–XIII (2017).

15. Agarwal, A. *et al.* Comparison and calibration of transcriptome data from RNA-Seq and tiling arrays. *BMC Genomics* **11**, 383 (2010).

16. Humann, J. L., Lee, T., Ficklin, S. & Main, D. Structural and Functional Annotation of Eukaryotic Genomes with GenSAS. *Gene Prediction* **1962**, 29–51 (2019).

17. Stein, L. Genome annotation: from sequence to biology. *Nat. Rev. Genet.* **2**, 493–503 (2001).

18. Finotello, F. & Di Camillo, B. Measuring differential gene expression with RNA-seq: challenges and strategies for data analysis. *Brief. Funct. Genomics* **14**, 130–142 (2015).

19. Hrdlickova, R., Toloue, M. & Tian, B. RNA -Seq methods for transcriptome analysis. *WIREs RNA* **8** (2017).

20. Maxam, A. M. & Gilbert, W. A new method for sequencing DNA. *Proc. Natl. Acad. Sci.* **74**, 560–564 (1977).

21. Prober, J. M. *et al.* A System for Rapid DNA Sequencing with Fluorescent Chain-Terminating Dideoxynucleotides. *Science* **238**, 336–341 (1987).

22. Padmanabhan, R., Jay, E. & Wu, R. Chemical Synthesis of a Primer and Its Use in the Sequence Analysis of the Lysozyme Gene of Bacteriophage T4. *Proc. Natl. Acad. Sci.* **71**, 2510–2514 (1974).

23. Adams, M. D. *et al.* Complementary DNA Sequencing: Expressed Sequence Tags and Human Genome Project. *Science* **252**, 1651–1656 (1991).

24. Feldmann, H. *et al.* Complete DNA sequence of yeast chromosome II. *EMBO J.* **13**, 5795–5809 (1994).

25. Jay, E., Bambara, R., Padmanabhan, R. & Wu, R. DNA sequence analysis: a general, simple and rapid method for sequencing large oligodeoxyribonucleotide fragments by mapping. *Nucleic Acids Res.* **1**, 331–354 (1974).

26. Beck, S. & Pohl, F. M. DNA sequencing with direct blotting electrophoresis. *EMBO J.* **3**, 2905–2909 (1984).

27. Smith, L. M. *et al.* Fluorescence detection in automated DNA sequence analysis. *Nature* **321**, 674–679 (1986).

28. International Human Genome Sequencing Consortium *et al.* Initial sequencing and analysis of the human genome. *Nature* **409**, 860–921 (2001).

29. Padmanabhan, R., Padmanabhan, R. & Wu, R. Nucleotide sequence analysis of DNA. *Biochem. Biophys. Res. Commun.* **48**, 1295–1302 (1972).

30. Onaga, L. A. Ray Wu as Fifth Business: Deconstructing collective memory in the history of DNA sequencing. *Stud. Hist. Philos. Sci. Part C Stud. Hist. Philos. Biol. Biomed. Sci.* **46**, 1–14 (2014).

31. Gilbert, W. & Maxam, A. The Nucleotide Sequence of the *lac* Operator. *Proc. Natl. Acad. Sci.* **70**, 3581–3584 (1973).

32. Venter, J. C. *et al.* The Sequence of the Human Genome. *Science* **291**, 1304–1351 (2001).

33. Fleischmann, R. D. *et al.* Whole-Genome Random Sequencing and Assembly of *Haemophilus influenzae* Rd. *Science* **269**, 496–512 (1995).

34. Jordan, B. Historical Background and Anticipated Developments. *Ann. N. Y. Acad. Sci.* **975**, 24–32 (2002).

35. Ewis, A. A. *et al.* A history of microarrays in biomedicine. *Expert Rev. Mol. Diagn.* **5**, 315–328 (2005).

36. Mlakar, V. & Glavac, D. DNA microarrays and their use in dermatology. *Acta Dermatovenerol. Alp. Pannonica Adriat.* **16**, 7–12 (2007).

37. Costa-Silva, J., Domingues, D. & Lopes, F. M. RNA-Seq differential expression analysis: An extended review and a software tool. *PLOS ONE* **12** (2017).

38. Kratz, A. & Carninci, P. The devil in the details of RNA-seq. *Nat. Biotechnol.* **32**, 882–884 (2014).

39. Li, P., Piao, Y., Shon, H. S. & Ryu, K. H. Comparing the normalization methods for the differential analysis of Illumina high-throughput RNA-Seq data. *BMC Bioinformatics* **16**, 347 (2015).

40. Corchete, L. A. *et al.* Systematic comparison and assessment of RNA-seq procedures for gene expression quantitative analysis. *Sci. Rep.* **10**, 19737 (2020).

41. BLAST topics. *National Library of Medicine* https://blast.ncbi.nlm.nih.gov/Blast.cgi (2022).

42. What is FASTA format? *Zhang Lab* https://zhanggroup.org/FASTA/ (2022).

43. FastA Format. *NGS Analysis* https://learn.gencore.bio.nyu.edu/ngs-file-formats/fastaa-format/ (2022).

44. Hosseini, M., Pratas, D. & Pinho, A. A Survey on Data Compression Methods for Biological Sequences. *Information* **7**, 56 (2016).

45. IUPAC-IUB Comm. Biochem. Nomenclatu. IUPAC-IUB (International Union of Pure and Applied Chemistry-International Union of Biochemistry) Commission of Biochemical Nomenclature. Abbreviated nomenclature of synthetic polypeptides (polymerized amino acids). Revised recommendations (1971). *Biochemistry* **11**, 942–944 (1972).

46. Alignment File Formats. *Alignment FileFormats* https://www.jalview.org/help/html/io/fileformats.html (2022).

47. GFF files. *John Hopkins University Center For Computational Biology* http://ccb.jhu.edu/software/stringtie/gff.shtml (2022).

48. GFF/GTF File Format - Definition and supported options. *Ensembl* https://www.ensembl.org/info/website/upload/gff.html (2022).

49. What are SAM & BAM files? *Zymo Research* https://www.zymoresearch.com/blogs/blog/what-are-sam-and-bam-files (2022).

50. SAM/BAM/CRAM Format. *NGS Analysis* https://learn.gencore.bio.nyu.edu/ngs-file-formats/sambam-format/ (2022).

51. SAM/BAM Format. *Bioinformatics Remarks* https://sites.google.com/site/bioinformaticsremarks/bioinfo/sam-bam-format (2022).

52. Zhao, S., Ye, Z. & Stanton, R. Misuse of RPKM or TPM normalization when comparing across samples and sequencing protocols. *RNA N. Y. N* **26**, 903–909 (2020).

53. Zhao, Y. *et al.* TPM, FPKM, or Normalized Counts? A Comparative Study of Quantification Measures for the Analysis of RNA-seq Data from the NCI Patient-Derived Models Repository. *J. Transl. Med.* **19**, 269 (2021).

54. GTEx Consortium. The GTEx Consortium atlas of genetic regulatory effects across human tissues. *Science* **369**, 1318–1330 (2020).

55. Du, T. *et al.* Key regulators of lipid metabolism drive endocrine resistance in invasive lobular breast cancer. *Breast Cancer Res. BCR* **20**, 106 (2018).

56. Begik, O. *et al.* Integrative analyses of the RNA modification machinery reveal tissue- and cancer-specific signatures. *Genome Biol.* **21**, 97 (2020).

57. Yu, S. *et al.* Comprehensive analysis of the SLC16A gene family in pancreatic cancer via integrated bioinformatics. *Sci. Rep.* **10**, 7315 (2020).

58. Gao, J. *et al.* Integrative analysis of complex cancer genomics and clinical profiles using the cBioPortal. *Sci. Signal.* **6**, 11 (2013).

59. Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L. & Wold, B. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat. Methods* **5**, 621–628 (2008).

60. Gene expression units explained: RPM, RPKM, FPKM, TPM, DESeq, TMM, SCnorm, GeTMM, and ComBat-Seq. *Data science blog* https://www.reneshbedre.com/blog/expression_units.html (2022).

61. Bullard, J. H., Purdom, E., Hansen, K. D. & Dudoit, S. Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. *BMC Bioinformatics* **11**, 94 (2010).

62. Love, M. I., Huber, W. & Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol.* **15**, 550 (2014).

63. Li, B. & Dewey, C. N. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC Bioinformatics* **12**, 323 (2011).

64. Bray, N. L., Pimentel, H., Melsted, P. & Pachter, L. Near-optimal probabilistic RNA-seq quantification. *Nat. Biotechnol.* **34**, 525–527 (2016).

65. Patro, R., Duggal, G., Love, M. I., Irizarry, R. A. & Kingsford, C. Salmon provides fast and bias-aware quantification of transcript expression. *Nat. Methods* **14**, 417–419 (2017).

66. D'Antonio, M. *et al.* RAP: RNA-Seq Analysis Pipeline, a new cloud-based NGS web application. *BMC Genomics* **16**, S3 (2015).

67. Koboldt, D. C., Ding, L., Mardis, E. R. & Wilson, R. K. Challenges of sequencing human genomes. *Brief. Bioinform.* **11**, 484–498 (2010).

68. Schatz, M. C., Langmead, B. & Salzberg, S. L. Cloud computing and the DNA data race. *Nat. Biotechnol.* **28**, 691–693 (2010).

69. Hong, D. *et al.* FX: an RNA-Seq analysis tool on the cloud. *Bioinformatics* **28**, 721–723 (2012).

70. Prieto, C. & Barrios, D. RaNA-Seq: Interactive RNA-Seq analysis from FASTQ files to functional analysis. *Bioinformatics* (2019).

71. Bohnert, R. & Ratsch, G. rQuant.web: a tool for RNA-Seq-based transcript quantitation. *Nucleic Acids Res.* **38**, W348–W351 (2010).

72. Boyes, K. Callorhinchus milii - Elephant fish (Also: Ghost shark; Plownose chimaera; Reperepe; Silver fish). *Animal Diversity Web* https://animaldiversity.org/accounts/Callorhinchus_milii/ (2022)

73. Borrell, B. Why sharks have no bones. *Nature* **14487** (2014).

74. Ravi, V. *et al.* Elephant shark (*Callorhinchus milii*) provides insights into the evolution of Hox gene clusters in gnathostomes. *Proc. Natl. Acad. Sci.* **106**, 16327–16332 (2009).

75. Nakatani, Y. *et al.* Reconstruction of proto-vertebrate, proto-cyclostome and proto-gnathostome genomes provides new insights into early vertebrate evolution. *Nat. Commun.* **12**, 4489 (2021).

76. Lawal, R. A. *et al.* The wild species genome ancestry of domestic chickens. *BMC Biol.* **18**, 13 (2020).

77. Bahr, J. M. The Chicken as a Model Organism. in *Sourcebook of Models for Biomedical Research* 161–167 (2008).

78. Hydrolagus colliei (Lay & Bennett, 1839). *World Register of Marine Species (WORMS)* https://www.marinespecies.org/aphia.php?p=taxdetails&id=271406 (2022).

79. Mazza, G. Hydrolagus colliei. *Monaco Nature Encyclopedia* https://www.monaconatureencyclopedia.com/hydrolagus-colliei/?lang=em (2022).

80. Didier, D. A. & Rosenberger, L. J. The spotted ratfish, Hydrolagus colliei: Notes on its biology with a redescription of the species (Holocephali: Chimaeridae). *Calif. Fish Game* **88**, 112–125 (2002).

81. Berio, F. *et al.* Diversity and Evolution of Mineralized Skeletal Tissues in Chondrichthyans. *Front. Ecol. Evol.* **9**, 660767 (2021).

82. Thorndyke, M. C., Reeve, J. R. & Vigna, S. R. Biological activity of a bombesin-like peptide extracted from the intestine of the ratfish, Hydrolagus colliei. *Comp. Biochem. Physiol. Part C Comp. Pharmacol.* **96**, 135–140 (1990).

83. Bergés-Tiznado, M. E. *et al.* The spotted ratfish Hydrolagus colliei as a potential biomonitor of mercury and selenium from deep-waters of the northern Gulf of California. *Mar. Pollut. Bull.* **164**, 112102 (2021).

84. Angulo, A., López, M. I., Bussing, W. A. & Murase, A. Records of chimaeroid fishes (Holocephali: Chimaeriformes) from the Pacific coast of Costa Rica, with the description of a new species of Chimera (Chimaeridae) from the eastern Pacific Ocean. *Zootaxa* **3861**, 554 (2014).

85. Thorvaldsdottir, H., Robinson, J. T. & Mesirov, J. P. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Brief. Bioinform.* **14**, 178–192 (2013).

86. Robinson, J. T. *et al.* Integrative genomics viewer. *Nat. Biotechnol.* **29**, 24–26 (2011).

87. Kent, W. J. *et al.* The Human Genome Browser at UCSC. *Genome Res.* **12**, 996–1006 (2002).

88. PEP 206 - Python Advanced Library. *Python Enhancement Proporsals* https://peps.python.org/pep-0206/ (2000).

89. The Making of Python - A Conversation with Guido van Rossum, Part I. *artima* https://www.artima.com/articles/the-making-of-python (2003).

90. PHP: Preface. *PHP* https://www.php.net/manual/en/preface.php (2022).

91. PHP: History of PHP and Related Projects. https://www.php.net/manual/en/history.php (2022).

92. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. *J. Mol. Biol.* **215**, 403–410 (1990).

93. Zhang, Z., Schwartz, S., Wagner, L. & Miller, W. A Greedy Algorithm for Aligning DNA Sequences. *J. Comput. Biol.* **7**, 203–214 (2000).

94. Dainat, J. *et al.* NBISweden/AGAT: AGAT-v0.9.2. (2022)

95. Buels, R. *et al.* JBrowse: a dynamic web platform for genome visualization and analysis. *Genome Biol.* **17**, 66 (2016).

96. Kim, D., Langmead, B. & Salzberg, S. L. HISAT: a fast spliced aligner with low memory requirements. *Nat. Methods* **12**, 357–360 (2015).

97. Kim, D., Paggi, J. M., Park, C., Bennett, C. & Salzberg, S. L. Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype. *Nat. Biotechnol.* **37**, 907–915 (2019).

98. Zhang, Y., Park, C., Bennett, C., Thornton, M. & Kim, D. Rapid and accurate alignment of nucleotide conversion sequencing reads with HISAT-3N. *Genome Res.* **31**, 1290–1295 (2021).

99. Liao, Y., Smyth, G. K. & Shi, W. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics* **30**, 923–930 (2014).

100. Wickham, H. *et al.* Welcome to the Tidyverse. *J. Open Source Softw.* **4**, 1686 (2019).

101. Ferragina, P. & Manzini, G. Opportunistic data structures with applications. in *Proceedings 41st Annual Symposium on Foundations of Computer Science* 390–398 (2000).

102. Burrows, M. & Wheeler, D. J. A Block-sorting Lossless Data Compression Algorithm. (1994).

103. Musich, R., Cadle-Davidson, L. & Osier, M. V. Comparison of Short-Read Sequence Aligners Indicates Strengths and Weaknesses for Biologists to Consider. *Front. Plant Sci.* **12**, 657240 (2021).

104. Alser, M. *et al.* Technology dictates algorithms: recent developments in read alignment. *Genome Biol.* **22**, 249 (2021).

105. Donato, L., Scimone, C., Rinaldi, C., D'Angelo, R. & Sidoti, A. New evaluation methods of read mapping by 17 aligners on simulated and empirical NGS data: an updated comparison of DNA- and RNA-Seq data from Illumina and Ion Torrent technologies. *Neural Comput. Appl.* **33**, 15669–15692 (2021).

106. Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **25**, 1754–1760 (2009).

107. Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29**, 15–21 (2013).

108. Jin, H., Wan, Y.-W. & Liu, Z. Comprehensive evaluation of RNA-seq quantification methods for linearity. *BMC Bioinformatics* **18**, 117 (2017).

109. Chandramohan, R., Po-Yen Wu, Phan, J. H. & Wang, M. D. Benchmarking RNA-Seq quantification tools. *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* 647–650 (2013).

110. Teng, M. *et al.* A benchmark for RNA-seq quantification pipelines. *Genome Biol.* **17**, 74 (2016).

111. Zhang, C., Zhang, B., Lin, L.-L. & Zhao, S. Evaluation and comparison of computational tools for RNA-seq isoform quantification. *BMC Genomics* **18**, 583 (2017).

112. Sarantopoulou, D. *et al.* Comparative evaluation of full-length isoform quantification from RNA-Seq. *BMC Bioinformatics* **22**, 266 (2021).

113. Anders, S., Pyl, P. T. & Huber, W. *HTSeq - A Python framework to work with high-throughput sequencing data* (2014).

114. Kovaka, S. *et al.* Transcriptome assembly from long-read RNA-seq alignments with

StringTie2. *Genome Biol.* **20**, 278 (2019).

115. Shumate, A., Wong, B., Pertea, G. & Pertea, M. Improved transcriptome assembly using a hybrid of long and short reads with StringTie. *PLOS Comput. Biol.* **18** (2022).

116. Danecek, P. *et al.* Twelve years of SAMtools and BCFtools. *GigaScience* **10** (2021).

117. Davies, W. I. L. *et al.* Evolution and Functional Characterisation of Melanopsins in a Deep-Sea Chimaera (Elephant Shark, Callorhinchus milii). *PLoS ONE* **7** (2012).

118. Fonseca, E. S. S. *et al.* Evolutionary Plasticity in Detoxification Gene Modules: The Preservation and Loss of the Pregnane X Receptor in Chondrichthyes Lineages. *Int. J. Mol. Sci.* **20**, 2331 (2019).

119. Ravi, V. *et al.* Sequencing of Pax6 Loci from the Elephant Shark Reveals a Family of Pax6 Genes in Vertebrate Genomes, Forged by Ancient Duplications and Divergences. *PLoS Genet.* **9** (2013).

120. Castro, L. F. C. *et al.* Recurrent gene loss correlates with the evolution of stomach phenotypes in gnathostome history. *Proc. R. Soc. B Biol. Sci.* **281**, 20132669 (2014).

121. TRIM50. *Alliance of Genome Resources* https://www.alliancegenome.org/gene/HGNC:19017 (2022).

122. Q9Y6J6 · KCNE2_HUMAN. *UniProt* https://www.uniprot.org/uniprotkb/Q9Y6J6/entry (2022).

123. Q2M1V0 · ISX_HUMAN. *UniProt* https://www.uniprot.org/uniprotkb/Q2M1V0 (2022).

124. Brooke, N. M., Garcia-Fernàndez, J. & Holland, P. W. The ParaHox gene cluster is an evolutionary sister of the Hox gene cluster. *Nature* **392**, 920–922 (1998).

125. PDX1 Gene - Pancreatic And Duodenal Homeobox 1. https://www.genecards.org/cgi-bin/carddisp.pl?gene=PDX1#expression (2022).

Appendix A

Retrieved results of the mapping of next generation reads onto the genomes using HISAT2 mapping program.

Callorhinchus milii

SRA	Tissue	Assigned	Assigned (%)	MultiMapping	MultiMapping (%)	Ambiguity	Ambiguity (%)	Unmapped	Unmapped (%)	No Features	No Features (%)	Total Reads	Total Reads (%)
SRR1735385	Thymus	35568575	62,05	8214916	14,33	1036066	1,81	7706069	13,44	4794141	8,36	57319767	100,00
SRR513757	Testis	32731401	60,20	8363221	15,38	663598	1,22	6970582	12,82	5645297	10,38	54374099	100,00
SRR513758	Spleen	25095993	51,57	9079081	18,66	547836	1,13	9032125	18,56	4909902	10,09	48664937	100,00
SRR513759	Ovary	33805529	58,74	7071164	12,29	620210	1,08	11269597	19,58	4788712	8,32	57555212	100,00
SRR513760	Liver	70009052	52,22	30967795	23,10	1940436	1,45	23399463	17,45	7760122	5,79	134076868	100,00
SRR514104	Muscle	35086157	41,94	13801505	16,50	1317324	1,57	25563375	30,56	7884832	9,43	83653193	100,00
SRR514105	Kidney	33562382	48,85	12240216	17,81	1136164	1,65	14106982	20,53	7662459	11,15	68708203	100,00
SRR514106	Intestine	48263118	55,02	17894532	20,40	1208332	1,38	12974419	14,79	7372883	8,41	87713284	100,00
SRR514107	Heart	30602321	61,16	6864716	13,72	576441	1,15	7401936	14,79	4592025	9,18	50037439	100,00
SRR514109	Brain	41434212	51,75	13213343	16,50	744320	0,93	13031344	16,28	11637935	14,54	80061154	100,00
SRR534176	Gills	20416475	47,95	8789147	20,64	448715	1,05	8236074	19,34	4691511	11,02	42581922	100,00

Gallus gallus

SRA	Tissue	Assigned	Assigned (%)	MultiMapping	MultiMapping (%)	Ambiguity	Ambiguity (%)	Unmapped	Unmapped (%)	No Features	No Features (%)	Total Reads	Total Reads (%)
ERR348569	Fem. Heart	48799111	63,52	16482093	21,45	6279035	8,17	3893398	5,07	1370578	1,78	76824215	100,00
ERR348573	Fem. Liver	36520341	26,97	90639973	66,93	4065079	3,00	3432875	2,53	770587	0,57	135428855	100,00
ERR348582	Fem. Lung	30935517	57,84	9742225	18,21	4274140	7,99	6320567	11,82	2216430	4,14	53488879	100,00

Hydrolagus colliei

SRA	Tissue	Assigned	Assigned (%)	MultiMapping	MultiMapping (%)	Ambiguity	Ambiguity (%)	Unmapped	Unmapped (%)	No Features	No Features (%)	Total Reads	Total Reads (%)
Hydrolagus10	Muscle	6705352	28,18	3037965	12,77	1555548	6,54	7850720	32,99	4649100	19,54	23798685	100,00
Hydrolagus11	Thymus	8271419	34,56	2400228	10,03	1962796	8,20	6352772	26,55	4943194	20,66	23930409	100,00
Hydrolagus12	Skin	8002791	30,13	3682933	13,87	2275130	8,57	6110255	23,01	6489297	24,43	26560406	100,00
Hydrolagus13	Spiral Valve	7549960	30,87	4026088	16,46	2175830	8,90	5231901	21,39	5473367	22,38	24457146	100,00
Hydrolagus14	RBC	9359441	30,01	9401665	30,15	1932664	6,20	5259180	16,86	5231563	16,78	31184513	100,00
Hydrolagus18	Spleen	9540792	34,37	5668366	20,42	1975281	7,11	4889447	17,61	5689060	20,49	27762946	100,00
Hydrolagus19	Kidney	7020006	24,84	4118991	14,58	2169500	7,68	7126681	25,22	7821062	27,68	28256240	100,00
Hydrolagus1	Upper Gill	8499727	31,78	4328704	16,19	2461218	9,20	5394061	20,17	6060399	22,66	26744109	100,00
Hydrolagus20	Liver	8220114	29,40	5639411	20,17	2017166	7,22	6327078	22,63	5753645	20,58	27957414	100,00
Hydrolagus25	Testis	8916609	34,51	4398701	17,03	1957748	7,58	4978970	19,27	5584020	21,61	25836048	100,00
Hydrolagus4	Heart	6516066	24,99	3198511	12,27	1744556	6,69	9094514	34,88	5521527	21,18	26075174	100,00
Hydrolagus5	Liver	8421023	33,61	3622182	14,46	1853343	7,40	6035893	24,09	5124788	20,45	25057229	100,00

Appendix B

`rawToFPKMnTPM.R`: R script developed to transform featurecounts raw read outputs to FPKM and TPM gene expression measurements using the Tidyverse package.

```
#! /usr/bin/env Rscript
args = commandArgs(trailingOnly=TRUE)

## packages
library(dplyr)
library(tidyr)
## upload the dataset
count_table <- read.table(args[1], sep="\t", stringsAsFactors=FALSE, header=TRUE, skip = 0)

## measurement functions
## from: https://haroldpimentel.wordpress.com/2014/05/08/what-the-fpkm-a-review-RNA-Seq-expression-units/
mpkm <- function(counts, lengths) {
    N <- sum(counts)
    exp( log(counts) + log(1e9) - log(lengths) - log(N) )
}

tpm <- function(counts, lengths) {
    rate <- log(counts) - log(lengths)
    denom <- log(sum(exp(rate)))
    exp(rate - denom + log(1e6))
}

## global standardization of the file
standardized_file <- count_table %>%
  gather(Sample, cnt, 7:ncol(count_table)) %>%
  group_by(Sample) %>%
  mutate(FPKM=mpkm(cnt, Length)) %>%
  mutate(TPM=tpm(cnt, Length)) %>%
  select(-Sample) %>%
  relocate(FPKM, .after=Geneid) %>%
  relocate(TPM, .after=FPKM) %>%
  relocate(Raw_Count=cnt) %>%
  relocate(Raw_Count, .after=Length)
## all numerics to string for quoting
standardized_file$FPKM=as.character(standardized_file$FPKM)
standardized_file$TPM=as.character(standardized_file$TPM)
standardized_file$Length=as.character(standardized_file$Length)
standardized_file$Raw_Count=as.character(standardized_file$Raw_Count)

## saving the table as an output
write.table(standardized_file, file=args[2], sep=";", row.names=FALSE, quote=TRUE)
```

Appendix C

`generateJBrowseJSFiles.py`: Python script that generates the corresponding assembly and track files requires for JBrowse2 based on the generated `config.json` file.

```
1  # generateJBrowseJSFiles.py
2  # !/usr/bin/env python
3  # coding: utf-8
4  # author: Filipe Caramelo
5
6  import argparse
7
8  parser = argparse.ArgumentParser(description='Generates assembly and track files for
   GeneAnalyst JBrowse')
9  parser.add_argument('i', metavar='config.json Input', type=str)
10 parser.add_argument('-n', metavar='assembly_name', type=str, help='Assembly Name
   (e.g. Callorhinchus millii genome assembly)', required=True)
11 parser.add_argument('-id', metavar='assemble_id', type=str, help='Assembly ID (e.g.
   IMCB_Cmil_1.0)', required=True)
12 args = parser.parse_args()
13
14 assembly_js=['export default {\n','\t\tname: "$ASSEMBLY_NAME$",\n','\t\tsequence:
   {\n','\t\t\ttype: "ReferenceSequenceTrack",\n','\t\t\ttrackId:
   "$TRACK_ID$",\n','\t\t\tadapter: {\n','\t\t\t\ttype:
   "IndexedFastaAdapter",\n','\t\t\t\tfastaLocation: {\n','\t\t\t\t\turi:
   "$FASTA_URL$"\n','\t\t\t\t},\n','\t\t\t\tfaiLocation: {\n','\t\t\t\t\turi:
   "$FASTA_FAI_URL$"\n','\t\t\t\t},\n','\t\t\t},\n','\t\t},\n','\t}']
15 tracks_js=['export default [\n','\t\t{\n','\t\t\ttype:
   "FeatureTrack",\n','\t\t\ttrackId: "$GZTRACKID$",\n','\t\t\tname:
   "Annotation",\n','\t\t\tassemblyNames:
   [\n','\t\t\t\t"$ASSEMBLY_NAME$"\n','\t\t\t],\n','\t\t\tadapter: {\n','\t\t\t\ttype:
   "Gff3TabixAdapter",\n','\t\t\t\tgffGzLocation: {\n','\t\t\t\t\turi:
   "$GZLOCATION$"\n','\t\t\t\t},\n','\t\t\t\tindex: {\n','\t\t\t\t\tlocation:
   {\n','\t\t\t\t\t\turi: "$TBILOCATION$"\n','\t\t\t\t\t},\n','\t\t\t\t\tindexType:
   "TBI"\n','\t\t\t\t},\n','\t\t\t},\n','\t\t},\n','\t]']
16 dic={'assembly_name':'','assembly_id':'','trackId':'','gztrackId':'','fastaLocation'
   :'','faiLocation':'','gzLocation':'','tbiLocation':''}
17
18 def get_pwd():
19     #get pwd of the config file
20     if args.i.find('/') != -1:
21         path=args.i.split('/')
22         path=path[0:-1]
23         path='/'.join(path)
24         path=path+'/'
25         return path
26     else: return ''
27
```

```
28 def fill_dic():
29     #fill dictionary information for generation
30     dic['assembly_name']=args.n
31     dic['assembly_id']=args.id
32     dic['trackId']=args.id+'-ReferenceSequenceTrack'
33     dic['gztrackId']=args.id+'-Annotation'
34
35 def create_assembly_js (pwd):
36     json_file=open(args.i,'r').readlines()
37     c=-1
38     index_assemblies=0
39     ## finds the line index of where the assembly part begins
40     for line in json_file:
41         c+=1
42         if line.find('assemblies') != -1: index_assemblies=c
43         elif line.find('tracks') != -1: index_tracks=c
44     ## cuts the assembly piece out
45     assemblies_json=json_file[index_assemblies:index_assemblies+20]
46     ## gets needed information
47     for line in assemblies_json:
48         if line.find('.fa"') != -1: fasta_url=line
49         elif line.find('.fai') != -1: fai_url=line
50     ## treats the gathered info
51     fasta_url=fasta_url.split(':')[1].replace('"','').replace('
   ','').replace('\t','').replace(',\n','')
52     dic['fastaLocation']=fasta_url
53     fai_url=fai_url.split(':')[1].replace('"','').replace('
   ','').replace('\t','').replace(',\n','')
54     dic['faiLocation']=fai_url
55     ## will add the information and generate the assembly.js file
56     output_file_pwd=pwd+'assembly.js'
57     out_assembly=open(output_file_pwd,'w')
58     for line in assembly_js:
59         if line.find('$ASSEMBLY_NAME$')!=-1:
60             line=line.replace('$ASSEMBLY_NAME$',dic['assembly_name'])
61             out_assembly.write(line)
62         elif line.find('$TRACK_ID$')!=-1:
63             line=line.replace('$TRACK_ID$',dic['trackId'])
64             out_assembly.write(line)
65         elif line.find('$FASTA_URL$')!=-1:
66             temp=pwd+dic['fastaLocation']
67             line=line.replace('$FASTA_URL$',temp)
68             out_assembly.write(line)
69         elif line.find('$FASTA_FAI_URL$')!=-1:
70             temp=pwd+dic['faiLocation']
71             line=line.replace('$FASTA_FAI_URL$',temp)
72             out_assembly.write(line)
73         else: out_assembly.write(line)
74
75 def create_tracks_js (pwd):
76     json_file=open(args.i,'r').readlines()
```

```
    c=-1
    index_tracks=0
    ## finds the line index of where the assembly part begins
    for line in json_file:
        c+=1
        if line.find('tracks') != -1: index_tracks=c
    ## cuts the assembly piece out
    tracks_json=json_file[index_tracks:index_tracks+21]
    ## gets needed information
    for line in tracks_json:
        if line.find('.gz"') != -1: annotation_url=line
        elif line.find('.tbi') != -1: tbi_url=line
    ## treats the gathered info
    annotation_url=annotation_url.split(':')[1].replace('"','').replace('
,'').replace('\t','').replace('\n','')
    dic['gzLocation']=annotation_url
    tbi_url=tbi_url.split(':')[1].replace('"','').replace('
,'').replace('\t','').replace('\n','')
    dic['tbiLocation']=tbi_url
    ## will add the information and generate the tracks.js file
    output_file_pwd=pwd+'tracks.js'
    out_assembly=open(output_file_pwd,'w')
    for line in tracks_js:
        if line.find('$ASSEMBLY_NAME$')!=-1:
            line=line.replace('$ASSEMBLY_NAME$',dic['assembly_name'])
            out_assembly.write(line)
        elif line.find('$GZTRACKID$')!=-1:
            line=line.replace('$GZTRACKID$',dic['gztrackId'])
            out_assembly.write(line)
        elif line.find('$GZLOCATION$')!=-1:
            temp=pwd+dic['gzLocation']
            line=line.replace('$GZLOCATION$',temp)
            out_assembly.write(line)
        elif line.find('$TBILOCATION$')!=-1:
            temp=pwd+dic['tbiLocation']
            line=line.replace('$TBILOCATION$',temp)
            out_assembly.write(line)
        else: out_assembly.write(line)

if __name__ == '__main__':
    pwd=get_pwd()
    fill_dic()
    create_assembly_js(pwd)
    create_tracks_js(pwd)
    #terminal output
    print('\n>> assembly.js and tracks.js files were created from config.json\n')
    print('\nRetrieved information:')
    print(dic)
    print('\n')
```

Appendix D

`reformatBLASThtmlFinal.py`: Python script that reformats the HTML page of the BLAST results page generated directly from NCBI's BLAST software to include a table, checkboxes for sequence selection and corresponding button functions.

```
1  # reformatBLASThtmlFinal.py
2  # !/usr/bin/env python
3  # coding: utf-8
4  # author: Filipe Caramelo
5
6  import argparse
7  import sys
8  import os
9
10 # Create the parser
11 my_parser = argparse.ArgumentParser(description='automatic guideline (for usage)')
12 my_parser.add_argument('inputHTML',
13                        metavar='inputHTML',
14                        type=str,
15                        help='the HTML you want to treat')
16 my_parser.add_argument('inputTable',
17                        metavar='inputTable',
18                        type=str,
19                        help='the paralel table')
20 my_parser.add_argument('-t',
21                        metavar='target path',
22                        type=str,
23                        help='internal code')
24 # Execute parse_args()
25 args = my_parser.parse_args()
26
27 def removePRE (filepath, output):
28     file=open(filepath)
29     file=file.readlines()
30     tempFile=open(output,'w')
31     for line in file:
32         if line.startswith('<PRE>'):
33             tempFile.write('\n')
34         else: tempFile.write(line)
35
36 def reformatHEADER (filepath,output):
37     '''this function reformats the header'''
38     file=open(filepath)
39     file=file.readlines()
40     tempFile=open(output,'w')
41     for line in file:
42         if line.startswith('<HEAD>'):
43             tempFile.write('\n')
44         elif line.startswith('<BODY BG'):
```

```
            tempFile.write('\n')
        elif line.find('<b>BLAST')!=-1:
            '''here im going to retrieve the blast version and present it as I want it'''
            version=line.split(' ')
            function=version[0]
            version=version[1]
            function=function.replace('<b>','')
            version=version.replace('</b>\n','')
            versionString='<h1>'+function+' v.'+version+'</h1>\n'
            tempFile.write(versionString)
        elif line.startswith('<b><a\n'):
            tempFile.write('<p><b>Reference:</b> Zheng Zhang, Scott Schwartz, Lukas Wagner, and Webb Miller (2000), "A greedy algorithm for aligning DNA sequences", J Comput Biol 2000; 7(1-2):203-14.</p>')
        elif line.startswith('href='):
            tempFile.write('\n')
        elif line.startswith('etrieve&list_uids=10890397&dopt=Citation'):
            tempFile.write('\n')
        elif line.startswith('Zheng Zhang'):
            tempFile.write('\n')
        elif line.startswith('"A greedy'):
            tempFile.write('\n')
        elif line.startswith('7(1-2):'):
            tempFile.write('\n')
        else:
            tempFile.write(line)

def joinDatabaseInfo(filepath,output):
    file=open(filepath)
    file=file.readlines()
    tempFile=open(output,'w')
    newLine=''
    dbLine=-1
    c=-1
    for line in file:
        c+=1
        if line.startswith('Database:'):
            dbLine=c+1
            newLine+=line
        elif c==dbLine:
            newLine=newLine.replace('\n','')
            newLine+=' | '+ line[11::]
            tempFile.write(newLine)
        else: tempFile.write(line)

def treatTab (filepath, output):
    file=open(filepath)
    file=file.readlines()
    tempFile=open(output,"w")
```

```
    header='Query ID\tTarget ID\t% Identical Matches\tAlignment Length\tNr Mismatches\tNr Gaps\tQuery: Start\tQuery: End\tTarget: Start\tTarget: End\te-value\tbitscore\n'
    tempFile.write(header)
    for line in file:
        tempFile.write(line)
    tempFile.close()

def createDbLinkage(filepath):
    file=open(filepath)
    file=file.readlines()
    dbPath=''
    for line in file:
        if line.find('Database:'):
            line=line.split('|')
            line=line[0]
            if line.find('Draft Genome')!=-1:
                dbPath='data/Sco_p.fa'
            if line.find('Genes')!=-1:
                dbPath='data/Sco_Annotated_genes.fasta'
            if line.find('mRNA')!=-1:
                dbPath='data/Sco_Annotated_mrna.fasta'
            if line.find('Proteins')!=-1:
                dbPath='data/Sco_Annotated_proteins.fasta'
            if line.find('Transcripts')!=-1:
                dbPath='data/Sco_Annotated_transcripts.fasta'
    return dbPath

def createTable (filepath, table, output):
    file=open(filepath)
    file=file.readlines()
    table=open(table)
    table=table.readlines()
    tempFile=open(output,"w")
    dbLine=-1
    c=-1
    for line in file:
        c+=1
        if c==2:
            tempFile.write('<BODY>\n')
        if line.startswith('Database:'):
            dbLine=c+2
            tempFile.write(line)
        elif c==dbLine:
            #create table
            tempFile.write('<table border="0" class="styled-table">\n')
            tempFile.write('\t<thead>\n')
            tempFile.write('\t\t<tr style="text-align: center; font-family: "Montserrat";font-weight: 700;">\n')
            tempFile.write('\t\t\t<th>Query ID</th>\n')
            tempFile.write('\t\t\t<th>Target ID</th>\n')
```

```
140             tempFile.write('\t\t\t<th>% Identical Matches</th>\n')
141             tempFile.write('\t\t\t<th>Alignment Length</th>\n')
142             tempFile.write('\t\t\t<th>Nr. Mismatches</th>\n')
143             tempFile.write('\t\t\t<th>Nr. Gaps</th>\n')
144             tempFile.write('\t\t\t<th>Query: Start</th>\n')
145             tempFile.write('\t\t\t<th>Query: End</th>\n')
146             tempFile.write('\t\t\t<th>Target: Start</th>\n')
147             tempFile.write('\t\t\t<th>Target: End</th>\n')
148             tempFile.write('\t\t\t<th>e-value</th>\n')
149             tempFile.write('\t\t\t<th>Bitscore</th>\n')
150             tempFile.write('\t\t</tr>\n')
151             tempFile.write('\t</thead>\n')
152             tempFile.write('\t<tbody>\n')
153             #automatic filling .. 12 columns
154             for entry in table[1::]:
155                 entry=entry.split('\t')
156                 tempFile.write('\t\t<tr>\n')
157                 tempFile.write('\t\t\t<td>'+entry[0]+'</td>\n')
158                 tempFile.write('\t\t\t<td><a
    href=#'+entry[1]+'>'+entry[1]+'</a></td>\n')
159                 tempFile.write('\t\t\t<td>'+entry[2]+'</td>\n')
160                 tempFile.write('\t\t\t<td>'+entry[3]+'</td>\n')
161                 tempFile.write('\t\t\t<td>'+entry[4]+'</td>\n')
162                 tempFile.write('\t\t\t<td>'+entry[5]+'</td>\n')
163                 tempFile.write('\t\t\t<td>'+entry[6]+'</td>\n')
164                 tempFile.write('\t\t\t<td>'+entry[7]+'</td>\n')
165                 tempFile.write('\t\t\t<td>'+entry[8]+'</td>\n')
166                 tempFile.write('\t\t\t<td>'+entry[9]+'</td>\n')
167                 tempFile.write('\t\t\t<td>'+entry[10]+'</td>\n')
168                 tempFile.write('\t\t\t<td>'+entry[11]+'</td>\n')
169                 tempFile.write('\t\t</tr>\n')
170             tempFile.write('\t</tbody>\n')
171             tempFile.write('</table>\n')
172             # agr era adicionar botoes de select e download
173             tempFile.write('<ul class="button" style="margin:0;margin-
    top:10px;padding:0;list-style:none;display:flex;justify-content: space-between;box-
    sizing:border-box;">\n')
174             tempFile.write('\t<li>\n')
175             tempFile.write('\t\t<input type="button" onclick="selects()"
    value="Select All"/>\n')
176             tempFile.write('\t\t<input type="button" onclick="deSelect()"
    value="Deselect All"/>\n')
177             tempFile.write('\t</li>\n')
178             tempFile.write('\t<li>\n')
179             tempFile.write('\t\t<a href="treated.resultTab.csv"><input type="button"
    name="downTab" value="Download Table"/></a>')
180             tempFile.write('\t\t<input type="submit" name="submit" value="Download
    Sequences"/>')
181             tempFile.write('\t</li>\n')
182             tempFile.write('</ul>\n')
183             # divisor
```

```
            tempFile.write('<hr>\n')
            tempFile.write('<PRE>\n')
        else:
            tempFile.write(line)

def addForm(filepath, dbSelected, output):
    file=open(filepath)
    file=file.readlines()
    tempFile=open(output,"w")
    identific='>'
    for line in file:
        if line.startswith('<ul'):
            tempFile.write('<form method="post" action="getFastaSequences.php">\n')
            string='<input type="hidden" name="dbSelected" value="'+ dbSelected +'">\n'
            tempFile.write(string)
            tempFile.write(line)
        elif line.startswith(identific):
            newLine=line.split('<a')
            name=newLine[0][1::]
            tempFile.write('<input type="checkbox" name="check_list[]" value="'+name+'"> ')
            tempFile.write(line)
        else:
            tempFile.write(line)

def addJavascript(filepath,output):
    file=open(filepath)
    file=file.readlines()
    tempFile=open(output,"w")
    for line in file:
        if line.startswith('<BODY>\n'):
            tempFile.write('<HEAD>\n')
            #adicionar java + css
            tempFile.write('<script type="text/javascript">\n')
            tempFile.write('\t$parent.find("button[name=submit]").click(function (){\n')
            tempFile.write('\t\twindow.location.href = "getFastaSequences.php";\n')
            tempFile.write('\t});\n')
            tempFile.write('\tfunction selects(){\n')
            tempFile.write('\t\t\tvar ele=document.getElementsByName("check_list[]");\n')
            tempFile.write('\t\t\tfor(var i=0; i<ele.length; i++){\n')
            tempFile.write('\t\t\t\t\tif(ele[i].type=="checkbox")\n')
            tempFile.write('\t\t\t\t\t\t\tele[i].checked=true;\n')
            tempFile.write('\t\t\t\t}\n')
            tempFile.write('\t}\n')
            tempFile.write('\tfunction deSelect(){\n')
            tempFile.write('\t\t\tvar ele=document.getElementsByName("check_list[]");\n')
            tempFile.write('\t\t\tfor(var i=0; i<ele.length; i++){\n')
```

```
            tempFile.write('\t\t\t\t\tif(ele[i].type=="checkbox")\n')
            tempFile.write('\t\t\t\t\t\t\tele[i].checked=false;\n')
            tempFile.write('\t\t\t}\n')
            tempFile.write('\t}\n')
            tempFile.write('</script>\n')
            tempFile.write('<style>\n')
            tempFile.write('@import
    url("https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,300;0,700;1,700
    &display=swap");\n\n')
            tempFile.write('.styled-table {\n')
            tempFile.write('\tborder-collapse: collapse;\n')
            tempFile.write('\tmargin: 0px 0;\n')
            tempFile.write('\tfont-size: 12px;\n')
            tempFile.write('\tfont-family: "Montserrat";\n')
            tempFile.write('\tfont-weight: 300;\n')
            tempFile.write('\tmin-width: 200px;\n')
            tempFile.write('\tbox-shadow: 0 0 20px rgba(0, 0, 0, 0.15);\n')
            tempFile.write('}\n\n\n')
            tempFile.write('.styled-table thead tr {\n')
            tempFile.write('\tbackground-color: rgb(141,196,219);\n')
            tempFile.write('\tcolor: #ffffff;\n')
            tempFile.write('\ttext-align: center;\n')
            tempFile.write('}\n\n')
            tempFile.write('.styled-table th,\n')
            tempFile.write('.styled-table td {\n')
            tempFile.write('\tpadding: 8px 20px;\n')
            tempFile.write('}\n\n')
            tempFile.write('.styled-table tbody tr {\n')
            tempFile.write('\tborder-bottom: 1px solid #dddddd;\n')
            tempFile.write('}\n\n')
            tempFile.write('.styled-table tbody tr:nth-of-type(even) {\n')
            tempFile.write('\tbackground-color: #f3f3f3;\n')
            tempFile.write('}\n\n')
            tempFile.write('.styled-table tbody tr:last-of-type {\n')
            tempFile.write('\tborder-bottom: 2px solid rgb(141,196,219);\n')
            tempFile.write('}.styled-table tbody tr.active-row {\n')
            tempFile.write('\tfont-family: "Montserrat";\n')
            tempFile.write('\tfont-weight: 700;\n')
            tempFile.write('\tcolor: rgb(141,196,219);\n')
            tempFile.write('}\n')
            tempFile.write('</style>\n')
            tempFile.write('</HEAD>\n')
            tempFile.write('<BODY>\n')
        else:
            tempFile.write(line)

if __name__ == '__main__':
    removePRE(args.inputHTML,'temp1')
    reformatHEADER('temp1','temp2')
    joinDatabaseInfo('temp2','temp3')
    treatTab(args.inputTable,'treated.resultTab.csv')
```

```
    createTable('temp3', 'treated.resultTab.csv', 'temp4')
    dbChosen=args.t
    addForm('temp4', dbChosen, 'temp5')
    addJavascript('temp5','finalResult.html')
    os.system("rm temp*"
```

Appendix E

`create_tsv.py`: Python script that retrieves the data gathered from a previous `grep` function step using PHP and transforms it to the format required to be inputted onto csv_to_html_tables component.

```
# create_tsv.py
# !/usr/bin/env python3
# coding: utf-8
# author: Filipe Caramelo

import argparse

# passing arguments through the command line
parser = argparse.ArgumentParser(description='Script will read the grepped output and create FPKM and TPM interactive graphs')
parser.add_argument('i', metavar='Input Grepped Output', type=str, help='the file I grepped before with PHP')
args = parser.parse_args()

inputFile='tmp/'+args.i

file=open(inputFile)
out=open('tmp/geneExpQuery.tsv','w')
head='SRA'+'\t'+'Tissue'+'\t'+'FPKM'+'\t'+'TPM'+'\t'+'Raw_Count'+'\n'
out.write(head)

for line in file:
    line=line.replace('./','').replace('_final','').replace('"','').replace('\n','')
    tmplist=[]
    tmp=line.split('.txt:')
    tmplist.append(tmp[0])
    [tmplist.append(x) for x in tmp[1].split(';')]

newLine=tmplist[0]+'\t'+tmplist[1]+'\t'+tmplist[3]+'\t'+tmplist[4]+'\t'+tmplist[-1]+'\n'
    out.write(newLine)
```

Appendix F

`construct_n_redirect.py`: Python script that generates an automatic HTML page containing the boxplot visualizations powered by dimple.JS and automatically redirects the user.

```
# construct_n_redirect.py
# !/usr/bin/env python3
# coding: utf-8
# author: Filipe Caramelo

import argparse

# passing arguments through the command line
parser = argparse.ArgumentParser(description='Script will read the grepped output and create FPKM and TPM interactive graphs')
parser.add_argument('i', metavar='Input Grepped Output', type=str, help='the file I grepped before with PHP')
parser.add_argument('-g', metavar='Gene', type=str, help='FPKM/TPM', required=True)
parser.add_argument('-s', metavar='Species', type=str, help='FPKM/TPM', required=True)
args = parser.parse_args()

species = args.s
htmlStructure_millii=['<!doctype html><!DOCTYPE html><html lang="pt">\n', '<head>\n', '  <meta charset="utf-8">\n', '  <meta name="viewport" content="width=device-width, initial-scale=1">\n', '  <title>GeneAnalyst</title>\n', '  <meta name="description" content="Web application for whole genome visualization and analysis of gene expression data">\n', '  <meta name="author" content="Filipe Caramelo">\n', '  <meta property="og:title" content="GeneAnalyst">\n', '  <meta property="og:type" content="website">\n', '  <meta property="og:url" content="https://www.frontend-a.ciimar.up.pt/app/geneanalyst">\n', '  <meta property="og:description" content="Web application for whole genome visualization and analysis of gene expression data">\n', '  <!-- <link rel="icon" href="/favicon.ico"> -->\n', '  <!-- <link rel="icon" href="/favicon.svg" type="image/svg+xml"> -->\n', '  <!-- <link rel="apple-touch-icon" href="/apple-touch-icon.png"> -->\n', '  <link rel="stylesheet" href="../src/css/species.css">\n', "    <style>@import url('https://fonts.googleapis.com/css2?family=Anek+Malayalam:wght@200;300;400&display=swap');</style>\n", '    <style>\n', '      /* Style the tab */\n', '      .tab {\n', '        overflow: hidden;\n', '        border: 1px solid #ccc;\n', '        background-color: #f1f1f1;\n', '        height: 30px;\n', '      }\n', '      \n', '      /* Style the buttons inside the tab */\n', '      .tab button {\n', '        background-color: inherit;\n', '        float: left;\n', '        border: none;\n', '        outline: none;\n', '        cursor: pointer;\n', '        transition: 0.3s;\n', '        font-size: 15px;\n', '      }\n', '      \n', '      /* Change background color of buttons on hover */\n', '      .tab button:hover {\n', '        background-color: #ddd;\n', '      }\n', '      \n', '      /* Create an active/current tablink class */\n', '      .tab button.active {\n', '        background-color: #ccc;\n', '      }\n', '      \n', '      /* Style the tab content
```

```
*/\n', '      .tabcontent {\n', '        display: none;\n', '        padding: 6px
12px;\n', '        border: 1px solid #ccc;\n', '        border-top: none;\n', '
}\n', '      </style>\n', '</head>\n', '\n', '<body class="body">\n', '  <!-- MENU
HEADER -->\n', '  <div class="header">\n', '      <div class="column column-
one"><button onclick="history.back()">Back</button></div>\n', '      </div>\n',
'\n', '    <!-- IMAGE PLACEHOLDER -->\n', '    <div class="img-feature"
style="background-image: url(\'$$IMGSPECIES$$\');">\n', '      <div class="img-
feature.header" style="margin-top: 60px; margin-left: 13%; margin-right: 12.5%;
background-color: white; width:430px; height:60px; position:absolute; font-family:
\'Cairo\', sans-serif; font-weight: 700; font-size: 50px; margin-bottom: 30px; line-
height: 60px; padding-left: 10px;"><font style="font-style: italic;"><font
style="color: #02595d">Callorhinchus</font> millii</font></div>\n', '      <div
class="img-feature.header" style="margin-top: 150px; margin-left: 13%; margin-right:
12.5% ; background-color: white; width:1000px; height:40px; position: absolute;
font-family: \'Montserrat\', sans-serif; font-weight: 200; font-size: 15px; align-
items: center; align-content: center; text-align: justify; padding: 10px;"><font
style="font-style: italic;">Callorhinchus milii </font> is the slowest evolving of
all known vertebrates, including the coelacanth, and features extensive synteny
conservation with tetrapod genomes, making it a good model for comparative analyses
of gnathostome genomes.</div>\n', '  </div>\n', '\n', '  <div class="main-body"
style="font-family: \'Anek Malayalam\', sans-serif; font-weight: 200; font-size:
16px;">\n', '    <div class="container-fluid">\n', '      <main class="row">\n', '
<div class="col">\n', '            <h1>Expression Values: $$GENE$$</h1>\n<a
href="geneExpQuery.tsv"><input type="button" name="downTab" value="Download
Results"/></a>\n', '              <div id="table-container" style="overflow:
auto;"></div>\n', '          </div>\n', '      </main>\n', '      <footer
class="row">\n', '          <div class="col">\n', '          </div>\n', '
</footer>\n', '  </div>\n', '    <div class="tab">\n', '      <button
class="tablinks" onclick="openCity(event, \'FPKM\')">FPKM</button>\n', '
<button class="tablinks" onclick="openCity(event, \'TPM\')">TPM</button>\n', '
</div>\n', '\n', '    <div id="FPKM" class="tabcontent">\n', '        <div
id="chartContainer_FPKM">\n', '            <script
src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var FPKMsvg =
dimple.newSvg("#chartContainer_FPKM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var FPKMChart = new
dimple.chart(FPKMsvg, data);\n', '                FPKMChart.setBounds(60, 45, 1000,
315)\n', '                var x = FPKMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                FPKMChart.addMeasureAxis("y",
"FPKM");\n', '                FPKMChart.addMeasureAxis("z", null, "SRA");\n', '
FPKMChart.addSeries("SRA", dimple.plot.bar);\n', '
FPKMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '    \n', '
<div id="TPM" class="tabcontent">\n', '        <div id="chartContainer_TPM">\n', '
<script src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var TPMsvg =
dimple.newSvg("#chartContainer_TPM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var TPMChart = new
dimple.chart(TPMsvg, data);\n', '                TPMChart.setBounds(60, 45, 1000,
```

```
315)\n', '                var x = TPMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                TPMChart.addMeasureAxis("y",
"TPM");\n', '                TPMChart.addMeasureAxis("z", null, "SRA");\n', '
TPMChart.addSeries("SRA", dimple.plot.bar);\n', '
TPMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '\n',
'<script>\n', 'function openCity(evt, cityName) {\n', '  var i, tabcontent,
tablinks;\n', '  tabcontent = document.getElementsByClassName("tabcontent");\n', '
for (i = 0; i < tabcontent.length; i++) {\n', '    tabcontent[i].style.display =
"none";\n', '  }\n', '  tablinks = document.getElementsByClassName("tablinks");\n',
'  for (i = 0; i < tablinks.length; i++) {\n', '    tablinks[i].className =
tablinks[i].className.replace(" active", "");\n', '  }\n', '
document.getElementById(cityName).style.display = "block";\n', '
evt.currentTarget.className += " active";\n', '}\n', '</script>\n', '   \n', '\n',
'</div>\n', '<div id="end_piece" class="end_piece" style="text-align: center;
padding-top: 20px; padding-bottom: 20px; font-family: \'Anek Malayalam\', sans-
serif; font-weight: 200; font-size: 13px; color: #696969;">\n']
17 htmlStructure_gallus=['<!doctype html><!DOCTYPE html><html lang="pt">\n',
'<head>\n', '  <meta charset="utf-8">\n', '  <meta name="viewport"
content="width=device-width, initial-scale=1">\n', '  <title>GeneAnalyst</title>\n',
'  <meta name="description" content="Web application for whole genome visualization
and analysis of gene expression data">\n', '  <meta name="author" content="Filipe
Caramelo">\n', '  <meta property="og:title" content="GeneAnalyst">\n', '  <meta
property="og:type" content="website">\n', '  <meta property="og:url"
content="https://www.frontend-a.ciimar.up.pt/app/geneanalyst">\n', '  <meta
property="og:description" content="Web application for whole genome visualization
and analysis of gene expression data">\n', '  <!-- <link rel="icon"
href="/favicon.ico"> -->\n', '  <!-- <link rel="icon" href="/favicon.svg"
type="image/svg+xml"> -->\n', '  <!-- <link rel="apple-touch-icon" href="/apple-
touch-icon.png"> -->\n', '  <link rel="stylesheet"
href="../src/css/species.css">\n', "    <style>@import
url('https://fonts.googleapis.com/css2?family=Anek+Malayalam:wght@200;300;400&displa
y=swap');</style>\n", '    <style>\n', '      /* Style the tab */\n', '      .tab
{\n', '        overflow: hidden;\n', '        border: 1px solid #ccc;\n', '
background-color: #f1f1f1;\n', '        height: 30px;\n', '      }\n', '      \n', '
/* Style the buttons inside the tab */\n', '      .tab button {\n', '
background-color: inherit;\n', '        float: left;\n', '        border: none;\n',
'        outline: none;\n', '        cursor: pointer;\n', '        transition:
0.3s;\n', '        font-size: 15px;\n', '      }\n', '      \n', '      /* Change
background color of buttons on hover */\n', '      .tab button:hover {\n', '
background-color: #ddd;\n', '      }\n', '      \n', '      /* Create an
active/current tablink class */\n', '      .tab button.active {\n', '
background-color: #ccc;\n', '      }\n', '      \n', '      /* Style the tab content
*/\n', '      .tabcontent {\n', '        display: none;\n', '        padding: 6px
12px;\n', '        border: 1px solid #ccc;\n', '        border-top: none;\n', '
}\n', '      </style>\n', '</head>\n', '\n', '<body class="body">\n', '  <!-- MENU
HEADER -->\n', '  <div class="header">\n', '      <div class="column column-
one"><button onclick="history.back()">Back</button></div>\n', '      </div>\n',
'\n', '    <!-- IMAGE PLACEHOLDER -->\n', '    <div class="img-feature"
style="background-image: url(\'$$IMGSPECIES$$\');">\n', '      <div class="img-
feature.header" style="margin-top: 60px; margin-left: 13%; margin-right: 12.5%;
```

```
background-color: white; width:300px; height:60px; position:absolute; font-family:
\'Cairo\', sans-serif; font-weight: 700; font-size: 50px; margin-bottom: 30px; line-
height: 60px; padding-left: 10px;"><font style="font-style: italic;"><font
style="color: #812a2e">Gallus</font> gallus</font></div>\n', '      <div class="img-
feature.header" style="margin-top: 150px; margin-left: 13%; margin-right: 12.5% ;
background-color: white; width:1000px; height:90px; position: absolute; font-family:
\'Montserrat\', sans-serif; font-weight: 200; font-size: 15px; align-items: center;
align-content: center; text-align: justify; padding: 10px;">The chicken <font
style="font-style: italic;">(Gallus gallus)</font> has played <font style="font-
weight:700;">important roles in both scientific research and the general health and
welfare of humans</font>. In the field of developmental biology, the <font
style="font-weight:700;">chicken embryo model has provided insight into many
developmental processes</font> including cell migration, limb development and eye
formation. The <font style="font-weight:700;">discovery of avian oncogenic viruses
helped highlight the importance of specific genes in tumorigenesis</font> and the
chicken continues to be a popular model system for cancer and other
diseases.</div>\n', '  </div>\n', '\n', '  <div class="main-body" style="font-
family: \'Anek Malayalam\', sans-serif; font-weight: 200; font-size: 16px;">\n', '
<div class="container-fluid">\n', '      <main class="row">\n', '          <div
class="col">\n', '            <h1>Expression Values: $$GENE$$</h1>\n<a
href="geneExpQuery.tsv"><input type="button" name="downTab" value="Download
Results"/></a>\n', '              <div id="table-container" style="overflow:
auto;"></div>\n', '          </div>\n', '      </main>\n', '      <footer
class="row">\n', '          <div class="col">\n', '          </div>\n', '
</footer>\n', '  </div>\n', '    <div class="tab">\n', '      <button
class="tablinks" onclick="openCity(event, \'FPKM\')">FPKM</button>\n', '
<button class="tablinks" onclick="openCity(event, \'TPM\')">TPM</button>\n', '
</div>\n', '\n', '    <div id="FPKM" class="tabcontent">\n', '        <div
id="chartContainer_FPKM">\n', '            <script
src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var FPKMsvg =
dimple.newSvg("#chartContainer_FPKM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var FPKMChart = new
dimple.chart(FPKMsvg, data);\n', '                FPKMChart.setBounds(60, 45, 1000,
315)\n', '                var x = FPKMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                FPKMChart.addMeasureAxis("y",
"FPKM");\n', '                FPKMChart.addMeasureAxis("z", null, "SRA");\n', '
FPKMChart.addSeries("SRA", dimple.plot.bar);\n', '
FPKMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '    \n', '
<div id="TPM" class="tabcontent">\n', '        <div id="chartContainer_TPM">\n', '
<script src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var TPMsvg =
dimple.newSvg("#chartContainer_TPM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var TPMChart = new
dimple.chart(TPMsvg, data);\n', '                TPMChart.setBounds(60, 45, 1000,
315)\n', '                var x = TPMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                TPMChart.addMeasureAxis("y",
"TPM");\n', '                TPMChart.addMeasureAxis("z", null, "SRA");\n', '
```

```
TPMChart.addSeries("SRA", dimple.plot.bar);\n', '
TPMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '\n',
'<script>\n', 'function openCity(evt, cityName) {\n', '  var i, tabcontent,
tablinks;\n', '  tabcontent = document.getElementsByClassName("tabcontent");\n', '
for (i = 0; i < tabcontent.length; i++) {\n', '    tabcontent[i].style.display =
"none";\n', '  }\n', '  tablinks = document.getElementsByClassName("tablinks");\n',
'  for (i = 0; i < tablinks.length; i++) {\n', '    tablinks[i].className =
tablinks[i].className.replace(" active", "");\n', '  }\n', '
document.getElementById(cityName).style.display = "block";\n', '
evt.currentTarget.className += " active";\n', '}\n', '</script>\n', '   \n', '\n',
'</div>\n', '<div id="end_piece" class="end_piece" style="text-align: center;
padding-top: 20px; padding-bottom: 20px; font-family: \'Anek Malayalam\', sans-
serif; font-weight: 200; font-size: 13px; color: #696969;">\n']
18 htmlStructure_colliei=['<!doctype html><!DOCTYPE html><html lang="pt">\n',
'<head>\n', '  <meta charset="utf-8">\n', '  <meta name="viewport"
content="width=device-width, initial-scale=1">\n', '  <title>GeneAnalyst</title>\n',
'  <meta name="description" content="Web application for whole genome visualization
and analysis of gene expression data">\n', '  <meta name="author" content="Filipe
Caramelo">\n', '  <meta property="og:title" content="GeneAnalyst">\n', '  <meta
property="og:type" content="website">\n', '  <meta property="og:url"
content="https://www.frontend-a.ciimar.up.pt/app/geneanalyst">\n', '  <meta
property="og:description" content="Web application for whole genome visualization
and analysis of gene expression data">\n', '  <!-- <link rel="icon"
href="/favicon.ico"> -->\n', '  <!-- <link rel="icon" href="/favicon.svg"
type="image/svg+xml"> -->\n', '  <!-- <link rel="apple-touch-icon" href="/apple-
touch-icon.png"> -->\n', '  <link rel="stylesheet"
href="../src/css/species.css">\n', "    <style>@import
url('https://fonts.googleapis.com/css2?family=Anek+Malayalam:wght@200;300;400&displa
y=swap');</style>\n", '    <style>\n', '      /* Style the tab */\n', '      .tab
{\n', '        overflow: hidden;\n', '        border: 1px solid #ccc;\n', '
background-color: #f1f1f1;\n', '        height: 30px;\n', '      }\n', '      \n', '
/* Style the buttons inside the tab */\n', '      .tab button {\n', '
background-color: inherit;\n', '        float: left;\n', '        border: none;\n',
'        outline: none;\n', '        cursor: pointer;\n', '        transition:
0.3s;\n', '        font-size: 15px;\n', '      }\n', '      \n', '      /* Change
background color of buttons on hover */\n', '      .tab button:hover {\n', '
background-color: #ddd;\n', '      }\n', '      \n', '      /* Create an
active/current tablink class */\n', '      .tab button.active {\n', '
background-color: #ccc;\n', '      }\n', '      \n', '      /* Style the tab content
*/\n', '      .tabcontent {\n', '        display: none;\n', '        padding: 6px
12px;\n', '        border: 1px solid #ccc;\n', '        border-top: none;\n', '
}\n', '      </style>\n', '</head>\n', '\n', '<body class="body">\n', '  <!-- MENU
HEADER -->\n', '  <div class="header">\n', '      <div class="column column-
one"><button onclick="history.back()">Back</button></div>\n', '      </div>\n',
'\n', '    <!-- IMAGE PLACEHOLDER -->\n', '    <div class="img-feature"
style="background-image: url(\'$$IMGSPECIES$$\');">\n', '      <div class="img-
feature.header" style="margin-top: 60px; margin-left: 13%; margin-right: 12.5%;
background-color: white; width:400px; height:60px; position:absolute; font-family:
\'Cairo\', sans-serif; font-weight: 700; font-size: 50px; margin-bottom: 30px; line-
height: 60px; padding-left: 10px;"><font style="font-style: italic;"><font
```

```
style="color: #c27552">Hydrolagus</font> colliei</font></div>\n', '      <div
class="img-feature.header" style="margin-top: 150px; margin-left: 13%; margin-right:
12.5% ; background-color: white; width:1000px; height:60px; position: absolute;
font-family: \'Montserrat\', sans-serif; font-weight: 200; font-size: 15px; align-
items: center; align-content: center; text-align: justify; padding: 10px;">The
spotted ratfish, <font style="font-style: italic;">Hydrolagus
colliei</font></Hydrolagus>, is common along the west coast of the United States
ranging from southern Alaska into the Gulf of California in depths ranging from near
the surface to recorded depths of 913 meters. Although this species is common, and
apparently abundant throughout its range, <font style="font-weight:700;">very little
is known of its biology</font> and it is not presently targeted in commercial
fisheries.</div>\n', '  </div>\n', '\n', '  <div class="main-body" style="font-
family: \'Anek Malayalam\', sans-serif; font-weight: 200; font-size: 16px;">\n', '
<div class="container-fluid">\n', '      <main class="row">\n', '          <div
class="col">\n', '            <h1>Expression Values: $$GENE$$</h1>\n<a
href="geneExpQuery.tsv"><input type="button" name="downTab" value="Download
Results"/></a>\n', '              <div id="table-container" style="overflow:
auto;"></div>\n', '          </div>\n', '      </main>\n', '      <footer
class="row">\n', '          <div class="col">\n', '          </div>\n', '
</footer>\n', '  </div>\n', '    <div class="tab">\n', '      <button
class="tablinks" onclick="openCity(event, \'FPKM\')">FPKM</button>\n', '
<button class="tablinks" onclick="openCity(event, \'TPM\')">TPM</button>\n', '
</div>\n', '\n', '    <div id="FPKM" class="tabcontent">\n', '        <div
id="chartContainer_FPKM">\n', '            <script
src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var FPKMsvg =
dimple.newSvg("#chartContainer_FPKM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var FPKMChart = new
dimple.chart(FPKMsvg, data);\n', '                FPKMChart.setBounds(60, 45, 1000,
315)\n', '                var x = FPKMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                FPKMChart.addMeasureAxis("y",
"FPKM");\n', '                FPKMChart.addMeasureAxis("z", null, "SRA");\n', '
FPKMChart.addSeries("SRA", dimple.plot.bar);\n', '
FPKMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '    \n', '
<div id="TPM" class="tabcontent">\n', '        <div id="chartContainer_TPM">\n', '
<script src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var TPMsvg =
dimple.newSvg("#chartContainer_TPM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var TPMChart = new
dimple.chart(TPMsvg, data);\n', '                TPMChart.setBounds(60, 45, 1000,
315)\n', '                var x = TPMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                TPMChart.addMeasureAxis("y",
"TPM");\n', '                TPMChart.addMeasureAxis("z", null, "SRA");\n', '
TPMChart.addSeries("SRA", dimple.plot.bar);\n', '
TPMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '\n',
'<script>\n', 'function openCity(evt, cityName) {\n', '  var i, tabcontent,
tablinks;\n', '  tabcontent = document.getElementsByClassName("tabcontent");\n', '
```

```
for (i = 0; i < tabcontent.length; i++) {\n', '    tabcontent[i].style.display =
"none";\n', '  }\n', '  tablinks = document.getElementsByClassName("tablinks");\n',
'  for (i = 0; i < tablinks.length; i++) {\n', '    tablinks[i].className =
tablinks[i].className.replace(" active", "");\n', '  }\n', '
document.getElementById(cityName).style.display = "block";\n', '
evt.currentTarget.className += " active";\n', '}\n', '</script>\n', '   \n', '\n',
'</div>\n', '<div id="end_piece" class="end_piece" style="text-align: center;
padding-top: 20px; padding-bottom: 20px; font-family: \'Anek Malayalam\', sans-
serif; font-weight: 200; font-size: 13px; color: #696969;">\n']
19 htmlStructure_balaeana=['<!doctype html><!DOCTYPE html><html lang="pt">\n',
'<head>\n', '  <meta charset="utf-8">\n', '  <meta name="viewport"
content="width=device-width, initial-scale=1">\n', '  <title>GeneAnalyst</title>\n',
'  <meta name="description" content="Web application for whole genome visualization
and analysis of gene expression data">\n', '  <meta name="author" content="Filipe
Caramelo">\n', '  <meta property="og:title" content="GeneAnalyst">\n', '  <meta
property="og:type" content="website">\n', '  <meta property="og:url"
content="https://www.frontend-a.ciimar.up.pt/app/geneanalyst">\n', '  <meta
property="og:description" content="Web application for whole genome visualization
and analysis of gene expression data">\n', '  <!-- <link rel="icon"
href="/favicon.ico"> -->\n', '  <!-- <link rel="icon" href="/favicon.svg"
type="image/svg+xml"> -->\n', '  <!-- <link rel="apple-touch-icon" href="/apple-
touch-icon.png"> -->\n', '  <link rel="stylesheet"
href="../src/css/species.css">\n', "    <style>@import
url('https://fonts.googleapis.com/css2?family=Anek+Malayalam:wght@200;300;400&displa
y=swap');</style>\n", '    <style>\n', '      /* Style the tab */\n', '      .tab
{\n', '        overflow: hidden;\n', '        border: 1px solid #ccc;\n', '
background-color: #f1f1f1;\n', '        height: 30px;\n', '      }\n', '      \n', '
/* Style the buttons inside the tab */\n', '      .tab button {\n', '
background-color: inherit;\n', '        float: left;\n', '        border: none;\n',
'        outline: none;\n', '        cursor: pointer;\n', '        transition:
0.3s;\n', '        font-size: 15px;\n', '      }\n', '      \n', '      /* Change
background color of buttons on hover */\n', '      .tab button:hover {\n', '
background-color: #ddd;\n', '      }\n', '      \n', '      /* Create an
active/current tablink class */\n', '      .tab button.active {\n', '
background-color: #ccc;\n', '      }\n', '      \n', '      /* Style the tab content
*/\n', '      .tabcontent {\n', '        display: none;\n', '        padding: 6px
12px;\n', '        border: 1px solid #ccc;\n', '        border-top: none;\n', '
}\n', '      </style>\n', '</head>\n', '\n', '<body class="body">\n', '  <!-- MENU
HEADER -->\n', '  <div class="header">\n', '      <div class="column column-
one"><button onclick="history.back()">Back</button></div>\n', '      </div>\n',
'\n', '    <!-- IMAGE PLACEHOLDER -->\n', '    <div class="img-feature"
style="background-image: url(\'$$IMGSPECIES$$\');">\n', '      <div class="img-
feature.header" style="margin-top: 60px; margin-left: 13%; margin-right: 12.5%;
background-color: white; width:440px; height:60px; position:absolute; font-family:
\'Cairo\', sans-serif; font-weight: 700; font-size: 50px; margin-bottom: 30px; line-
height: 60px; padding-left: 10px;"><font style="font-style: italic;"><font
style="color: #046b7a">Balaena</font> mysticetus</font></div>\n', '      <div
class="img-feature.header" style="margin-top: 150px; margin-left: 13%; margin-right:
12.5% ; background-color: white; width:1000px; height:75px; position: absolute;
font-family: \'Montserrat\', sans-serif; font-weight: 200; font-size: 15px; align-
items: center; align-content: center; text-align: justify; padding: 10px;">The
```

```
mechanisms for the longevity and resistance to aging-related diseases of bowhead
whales are unknown, but <font style="font-weight:700;">it is clear these animals
must possess aging prevention mechanisms</font>. In particular in context of <font
style="font-weight:700;">cancer</font>, bowhead whales must <font style="font-
weight:700;">have anti-tumour mechanisms</font>, because <font style="font-
weight:700;">given their large size and longevity, their cells must have a massively
lower chance of developing into cancer</font> when compared to human
cells.</div>\n', '  </div>\n', '\n', '  <div class="main-body" style="font-family:
\'Anek Malayalam\', sans-serif; font-weight: 200; font-size: 16px;">\n', '    <div
class="container-fluid">\n', '      <main class="row">\n', '          <div
class="col">\n', '            <h1>Expression Values: $$GENE$$</h1>\n<a
href="geneExpQuery.tsv"><input type="button" name="downTab" value="Download
Results"/></a>\n', '              <div id="table-container" style="overflow:
auto;"></div>\n', '          </div>\n', '      </main>\n', '      <footer
class="row">\n', '          <div class="col">\n', '          </div>\n', '
</footer>\n', '  </div>\n', '    <div class="tab">\n', '      <button
class="tablinks" onclick="openCity(event, \'FPKM\')">FPKM</button>\n', '
<button class="tablinks" onclick="openCity(event, \'TPM\')">TPM</button>\n', '
</div>\n', '\n', '    <div id="FPKM" class="tabcontent">\n', '        <div
id="chartContainer_FPKM">\n', '            <script
src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var FPKMsvg =
dimple.newSvg("#chartContainer_FPKM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var FPKMChart = new
dimple.chart(FPKMsvg, data);\n', '                FPKMChart.setBounds(60, 45, 1000,
315)\n', '                var x = FPKMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                FPKMChart.addMeasureAxis("y",
"FPKM");\n', '                FPKMChart.addMeasureAxis("z", null, "SRA");\n', '
FPKMChart.addSeries("SRA", dimple.plot.bar);\n', '
FPKMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '    \n', '
<div id="TPM" class="tabcontent">\n', '        <div id="chartContainer_TPM">\n', '
<script src="../dimpleJS/lib/d3.v4.3.0.js"></script>\n', '            <script
src="http://dimplejs.org/dist/dimple.v2.3.0.min.js"></script>\n', '
<script type="text/javascript">\n', '              var TPMsvg =
dimple.newSvg("#chartContainer_TPM", \'100%\', 400);\n', '
d3.tsv("$$EXPFILE$$", function (data) {\n', '                var TPMChart = new
dimple.chart(TPMsvg, data);\n', '                TPMChart.setBounds(60, 45, 1000,
315)\n', '                var x = TPMChart.addCategoryAxis("x", "Tissue");\n', '
x.addOrderRule("Tissue");\n', '                TPMChart.addMeasureAxis("y",
"TPM");\n', '                TPMChart.addMeasureAxis("z", null, "SRA");\n', '
TPMChart.addSeries("SRA", dimple.plot.bar);\n', '
TPMChart.draw();\n', '                x.titleShape.remove();\n', '
});\n', '            </script>\n', '          </div>\n', '    </div>\n', '\n',
'<script>\n', 'function openCity(evt, cityName) {\n', '  var i, tabcontent,
tablinks;\n', '  tabcontent = document.getElementsByClassName("tabcontent");\n', '
for (i = 0; i < tabcontent.length; i++) {\n', '    tabcontent[i].style.display =
"none";\n', '  }\n', '  tablinks = document.getElementsByClassName("tablinks");\n',
'  for (i = 0; i < tablinks.length; i++) {\n', '    tablinks[i].className =
tablinks[i].className.replace(" active", "");\n', '  }\n', '
```

```
    document.getElementById(cityName).style.display = "block";\n', '
    evt.currentTarget.className += " active";\n', '}\n', '</script>\n', '   \n', '\n',
    '</div>\n', '<div id="end_piece" class="end_piece" style="text-align: center;
    padding-top: 20px; padding-bottom: 20px; font-family: \'Anek Malayalam\', sans-
    serif; font-weight: 200; font-size: 13px; color: #696969;">\n']
20
21  dic={'hydrolagus_colliei':'../src/assets/hydrolagus.png','callorhinchus_millii':'../
    src/assets/millii.png', 'gallus_gallus':'../src/assets/gallus.png',
    'balaena_mysticetus':'../src/assets/balaeana.png'} #tmp imagem
22
23  if species=='hydrolagus_colliei':
24      htmlStructure=htmlStructure_colliei
25  elif species=='gallus_gallus':
26      htmlStructure=htmlStructure_gallus
27  elif species=='balaena_mysticetus':
28      htmlStructure=htmlStructure_balaeana
29  else: htmlStructure=htmlStructure_millii
30
31  outFilePath='tmp/expressionResult.html'
32  outFile=open(outFilePath,"w")
33
34  for line in htmlStructure:
35      if line.find('$$IMGSPECIES$$'):
36          line=line.replace('$$IMGSPECIES$$',dic[species])
37      if line.find('$$GENE$$'):
38          line=line.replace('$$GENE$$',args.g)
39      if line.find('$$EXPFILE$$'):
40          line=line.replace('$$EXPFILE$$','geneExpQuery.tsv')
41      outFile.write(line)
```

Appendix G

Full scope of the comparison between GeneAnalyst's *Callorhinchus milii* gene expression profiles and NCBI's gene expression profile using StringTie and featureCounts.

Green color indicates the same expression pattern was found, red indicates different pattern, orange indicates that the same pattern was found despite the counterparts failing to estimate the same behavior and blue indicates newly found expression that wasn't present in NCBI's gene expression profile.

Link: https://figshare.com/s/11d5c44f3f98d2c078bd

	brain		gills		heart		intestine		kidney		liver		muscle		spleen		testis		thymus	
geneid	stringtie	featurecounts	stringtie2	featurecounts2	stringtie3	featurecounts3	stringtie4	featurecounts4	stringtie8	featurecounts8	stringtie10	featurecounts10	stringtie12	featurecounts12	stringtie16	featurecounts16	stringtie18	featurecounts18	stringtie20	featurecounts20
septin8	270,373	67,499	138,065	40,795	31,797	5,19	95,174	25,721	48,015	8,3	13,768	4,505	31,191	9,93	143,316	48,014	81,72	29,562	13,339	3,662
tmub2	118,348	74,766	127,314	94,573	56,806	23,343	52,092	35,514	63,742	27,251	38,644	30,358	50,193	39,962	90,423	71,445	127,854	115,149	49,872	26,296
adamts13	0,466	0,258	0,559	0,428	0,165	0,034	0,242	0,093	0	0,02	0,03	0,01	0,07	0,182	0,18	0,088	0,877	0,408	3,705	1,042
svrag	4,985	2,084	3,535	2,721	1,523	0,228	3,45	1,413	2,398	1,052	6,631	2,894	1,476	1,879	14,445	6,285	7,411	2,961	2,832	0,304
shisa4	6,709	2,905	1,4	1,002	0,746	0,26	0,393	0,292	1,154	0,655	0,483	0,546	0,756	0,734	1	2,019	47,73	23,611	1,232	0,767
hnmph1l	183,721	134,817	203,925	178,592	83,244	39,373	185,92	148,841	91,342	45,599	66,308	64,14	102,89	92,595	322,902	303,825	416,954	401,974	83,087	56,843
nipsnap2	99,859	57,852	56,034	38,087	231,648	77,574	28,103	20,762	31,78	12,904	17,257	16,72	115,724	78,227	38,055	38,315	70,602	66,884	149,462	70,086
atp6v0cb	1309,238	614,493	351,199	211,861	164,772	53,231	295,832	158,502	319,933	105,384	131,543	81,715	372,964	233,367	239,482	149,525	197,573	123,743	153,283	66
LOC103179186	5,286	10,105	3,049	8,981	2,856	2,853	0,932	1,447	3,301	5,814	3,434	5,598	1,373	17,937	7,522	14,79	5,252	11,937	4,966	7,508
LOC103177298	10,398	7,76	18,458	15,161	5,496	7,4	21,172	16,106	5,959	2,612	8,834	8,024	5,968	1,788	21,229	19,53	36,233	32,57	4,996	3,108
trnam-cau	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
srrm4	13,75	2,703	0	0,012	0	0,009	0,063	0,041	0	0,021	0,092	0,046	0,5	0,516	0	0,084	0,261	0,133	0,429	0,147
pgam5	37,662	49,919	45,457	70,411	27,402	20,809	59,092	74,136	48,906	37,97	35,252	52,262	31,205	45,619	60,623	88,514	50,669	83,9	26,461	25,632
LOC121852047	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
nsdhl	49,761	74,819	19,49	33,334	2,576	2,394	9,31	15,141	9,167	9,386	16,524	29,424	1,709	3,261	6,571	11,479	70,083	131,408	0,543	0,643
ndufa10	449,361	102,593	142,602	40,869	262,925	36,63	207,586	51,143	266,06	40,673	194,878	52,633	133,872	37,96	122,072	42,51	284,085	86,768	89,349	16,598
LOC103188886	0,277	0	0	0	0,023	0	0,085	0	0	0	0,032	0	0,042	0	0,04	0	0	0	0,043	0
cep170aa	10,793	5,273	2,027	1,242	10,473	2,91	0,898	0,395	0,921	0,357	0,987	0,497	0,826	0,881	6,827	3,56	18,341	8,908	7,892	2,454
nxn	23,573	6,361	45,943	14,601	31,697	5,291	9,92	2,88	36,398	6,746	16,737	5,678	17,619	5,98	11,701	4,153	35,765	12,554	19,454	4,096
sbno1	37,855	33,577	50,571	53,929	19,259	10,907	41,395	39,482	23,911	14,597	19,663	21,348	23,218	26,488	114,456	123,606	81,794	92,043	15,732	11,925
LOC103174317	0,138	0,159	0	0	0	0	0	0,043	2,485	1,976	0,022	0,041	0	0	0,405	0,652	0	0,188	0	0
LOC121850765	2,045	12,228	0,598	2,954	0,18	0,859	0	0	0,443	1,803	0,31	2,098	0,198	1,535	0	0	1,043	9,607	0	0
LOC103177159	32,46	83,586	224,418	815,894	9,751	15,568	110,915	325,015	19,631	36,678	63,128	188,955	22,42	113,895	79,477	259,446	50,474	171,432	8,299	30,102
LOC103175117	3,611	2,949	29,041	23,823	7,51	1,858	16,498	8,875	2,012	0,824	5,96	3,727	4,443	3,655	65,353	49,237	1,102	1,275	7,274	3,132
trnaa-ugc_71	0,392	0	0,3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,485	0
mfsd9	5,099	3,573	25,078	19,481	1,633	0,613	8,746	6,054	12,545	6,288	4,601	3,244	1,487	1,374	7,056	5,524	18,396	14,242	1,354	0,966
LOC103172208	3,825	3,65	2,322	9,728	0,528	0,161	5,615	1,775	89,418	26,086	211,614	133,816	1,024	0,554	4,344	7,422	8,864	12,244	0,168	0,192
gnaz	21,117	0	0,171	0	0	0	0,279	0	0,44	0	0,044	0	0,138	0	0,026	0	5,979	0	0,162	0
ankrd11	72,548	69,554	115,553	130,823	29,21	18,073	48,424	50,584	52,668	34,219	15,219	18,246	58,985	71,479	90,047	103,449	41,834	50,258	41,122	31,843
LOC103178856	0	0,047	0	0,068	0,028	0,02	0	0,113	1,256	1,388	1,323	1,687	0	0,071	0,736	0,989	0,112	0,166	0	0,03
nfkbie	7,362	8,093	14,03	26,162	2,592	2,203	12,576	14,388	6,245	5,77	4,082	6,087	11,522	17,592	52,763	64,345	6,364	10,06	2,056	5,286
trnae-cuc_98	0	0	0	0	0	0	0	0	0	0	0	0	0,244	4,946	0	0	0	0	0	0
trnae-cuc_98	0	0	0	0	0	0	0	0	0	0	0	0	0,244	4,946	0	0	0	0	0	0
mcm9	3,436	2,799	4,431	4,824	2,124	1,296	1,719	1,732	1,079	0,831	1,065	0,981	1,101	1,279	4,091	3,755	8,709	8,361	2,369	1,545
runx2a	2,116	0,628	15,291	4,105	2,42	0,18	7,991	1,454	25,181	2,76	0,664	0,153	12,336	3,014	9,087	2,55	24,273	5,446	1,737	0,309
abhd13	6,729	12,39	9,278	19,709	5,909	6,928	6,644	12,848	9,956	11,538	8,001	17,636	3,037	7,88	10,772	24,071	12,062	25,673	8,035	10,406
tomm70a	73,733	39,253	47,666	29,748	29,304	9,682	68,273	38,363	48,696	17,401	32,899	21,01	27,185	18,222	35,41	24,6	81,276	54,236	31,764	14,638
rpl10a	296,474	168,643	686,407	442,036	420,649	145,454	478,662	281,08	387,304	139,82	556,938	360,865	594,923	399,01	703,784	473,854	622,795	451,246	330,196	139,667
LOC103187850	0,556	0,118	5,209	1,468	0	0,01	0	0	0	0	0,313	0,109	0	0,044	0	0,032	0,222	0,083	0	0,015
ltv1	63,062	45,746	56,897	47,951	52,315	22,807	58,053	44,818	40,329	19,789	24,799	20,745	55,547	49,235	69,422	61,384	87,811	80,6	29,093	17,136
pkd1b	1,926	1,909	0,051	0,07	0,528	0,308	0,168	0,179	0,829	0,582	0,176	0,219	2,017	2,622	1,306	1,695	20,117	25,196	0,018	0,026
LOC103182193	193,273	29,173	208,55	34,759	55,914	5,085	214,254	31,978	104,461	12,545	47,11	8,544	122,935	23,398	234,603	40,203	366,014	68,411	15,655	2,188
LOC103177247	3,808	9,162	2,937	6,846	0,376	0,613	1,608	4,181	0,902	0,95	0,277	0,221	0,923	2,426	1,107	3,512	5,932	17,46	0,616	0,659
LOC103173597	9,645	0,668	11,191	0,779	0,293	0,028	1,94	0,174	0,664	0,08	1,119	0,18	1,652	0,209	13,88	1,126	3,954	0,658	0,984	0,223
LOC103189266	1,107	3,204	1,164	10,908	0,181	1,682	0,466	3,246	1,064	4,835	0,235	2,1	0,137	4,59	1,04	7,663	2,101	8,442	0,358	2,524
LOC121850069	57,941	29,38	55,954	36,887	18,804	6,943	65,287	40,442	17,888	6,415	43,792	32,604	18,393	33,747	106,326	67,591	103,204	71,929	19,839	9,758
LOC121852683	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
hacl1	17,213	4,662	8,189	2,472	4,39	0,608	12,527	3,28	18,28	2,961	57,064	15,445	9,592	2,925	5,688	2,455	25,278	8,853	8,549	1,879
nuf2	17,436	7,407	27,395	12,558	9,668	2,583	36,563	16,53	6,832	2,044	4,228	2,16	19,692	12,302	44,181	21,4	224,054	118,696	33,229	11,512
LOC121848044	3,139	10,411	11,04	18,549	5,99	5,343	8,015	21,573	3,313	9,886	3,836	3,575	1,306	33,49	73,763	108,316	9,352	10,962	10,649	10,833
LOC103179346	4,879	10,62	1,132	3,039	4,379	5,312	0,933	1,369	0,42	0,592	0,131	0,322	1,001	2,688	0,276	0,795	0	0,159	0,922	1,671
LOC103183506	2,399	3,051	0,925	1,859	1,352	1,078	0,832	1,34	1,511	1,251	0,448	0,901	0,064	0,557	1,969	5,32	4,179	8,248	2,859	3,05
phf10	8,301	12,334	26,006	22,165	10,421	7,219	24,827	19,774	13,576	7,195	13,658	13,887	5,7	7,575	31,075	32,655	57,899	64,699	8,999	12,959
vps53	5,823	1,733	4,213	2,063	5,773	0,439	11,052	1,565	2,618	0,54	8,21	1,356	0,262	0,794	18,994	3,716	19,995	3,653	9,622	1,032
pfde4	81,077	33,327	63,729	30,262	86,533	22,871	47,939	20,768	48,331	13,268	18,771	9,811	64,607	32,641	60,114	30,449	182,104	94,033	94,07	30,887
LOC103189818	0,988	2,613	0,501	1,876	0,277	0,207	0,681	1,504	0,671	1,2	0,96	1,644	0,572	3,191	12,809	11,135	2,482	2,9	1,389	0,856
LOC103183537	13,352	13,279	86,853	101,62	15,599	10,034	19,052	20,946	61,861	42,268	3,678	4,379	12,391	15,81	33,229	41,279	18,647	22,891	8,304	6,757
tmem51a	7,484	4,937	13,382	10,954	3,183	1,181	17,039	11,005	7,378	3,902	8,153	6,406	1,397	1,472	5,543	6,764	14,213	12,575	0,918	0,799
LOC103189301	1,018	0,54	0	0,03	0,059	0,096	0	0,011	0	0,018	0	0,203	0	0,094	0	0,113	0,288	0,318	0	0,081
LOC103187138	0,765	1,354	2,165	3,532	0,765	0,685	0,41	0,765	1,114	0,989	0	0,523	4,172	6,883	0,889	1,93	6,56	7,898	1,134	1,183
LOC103189082	7,671	2,764	34,689	8,024	0,97	0,163	15,217	3,224	22,86	3,907	6,2	1,81	3,824	1,335	8,302	1,729	6,942	2,318	0,14	0,049
trnae-uac_42	0,283	0	0	0	0	0	0	0	0	0	0,087	0	0	0	0	0	0,393	0	0	0
ube4a	7,449	6,488	14,677	14,622	4,115	2,185	19,942	17,058	8,109	4,671	10,859	10,323	9,018	9,745	19,321	18,889	13,418	13,945	10,517	7,197
kxd1	102,548	85,121	64,61	62,016	30,706	16,266	46,538	42,716	49,517	27,653	74,526	73,82	41,105	42,677	69,89	71,837	116,988	117,208	34,258	23,249
ppp1r9a	14,292	3,666	6,238	2,618	2,03	0,63	12,67	3,141	5,526	1,1	1,813	0,529	2,071	1,309	24,615	7,113	6,654	2,188	3,092	0,562
LOC103186089	54,551	11,9	39,175	10,108	7,709	1,13	14,146	3,373	11,327	1,584	4,839	1,473	8,996	2,222	35,999	10,539	48,358	14,574	10,046	2,059
katnip	11,326	9,617	7,966	3,202	3,472	1,849	2,227	2,093	2,362	1,458	2,09	2,188	1,773	2,07	7,728	8,206	28,468	30,686	6,606	4,531
med6	39,304	29,528	39,2	34,765	6,85	3,196	18,593	15,197	13,086	6,71	10,529	9,97	14,768	14,147	36,329	34,342	41,624	41,511	16,323	15,643
trnat-cgu_58	0,262	0	0	0	0	0	0	0	0	0	0	0	0	0	0,825	0	0	0	0	0
otulina	23,098	73,564	30,671	103,941	7,367	12,788	18,699	58,363	17,013	32,467	9,126	30,243	43,308	145,218	35,059	128,969	16,389	70,867	33,58	77,146
LOC103179538	6,477	1,796	23,249	7,151	0,52	0,951	68,237	17,491	9,644	2,437	5,114	1,769	1,168	0,996	60,844	20,601	3,268	1,088	9,783	1,9
LOC103189297	0	0	0	0,226	0	0	0	0,05	0	0,193	0,2	0,369	0	0,023	2,627	1,439	0	0,147	0	0,06
LOC103180618	0,49	0,654	32,844	34,466	0,405	0,197	2,992	1,283	15,776	7,836	110,992	97,785	0	0,085	0,537	0,464	2,923	2,274	0	0,147
rgs4	18,382	12,02	7,41	5,68	7,76	2,767	14,244	9,728	13,271	5,086	30,749	21,966	6,131	4,792	34,031	33,758	51,657	43,625	15,002	7,871
LOC103185905	0	0	9,635	12,245	0,212	0,133	17,269	19,211	0,963	0,676	1,07	1,411	1,09	1,489	14,126	21,993	0,536	0,636	0,574	0,438
LOC121850385	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
vat1l	574,471	130,087	0,327	0,137	0	0,026	0,648	0,163	0,414	0,097	0	0,043	0,075	0,039	0,237	0,1	133,935	41,817	0	0,027
LOC103188581	3,762	1,567	10,927	4,144	1,827	0,401	8,186	2,523	3,034	0,78	4,656	2,018	1,563	0,823	23,691	10,07	19,326	6,646	4,209	1,248
LOC103186863	0,861	2,204	25,265	21,074	0	0,011	0,194	0,13	2,719	1,308	0,145	0,138	1,299	1,113	0	0,073	13,309	13,652	0	0,035
iba57	17,299	23,623	16,54	24,627	7,717	6,073	14,016	19,297	14,869	12,455	12,669	18,743	15,835	24,468	15,916	26,612	22,665	39,99	5,345	5,751
pde1a	4,924	0,963	2,356	0,579	1,788	0,133	1,119	0,148	1,381	0,184	0,264	0,052	0,275	0,139	1,239	0,218	6,834	1,272	1,14	0,195
trnaa-ugc_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LOC103190161	1,894	2,2	2,107	2,503	15,586	10,155	3,995	4,662	1,457	1,195	1,62	2,382	6,503	9,106	8,614	11,682	4,438	5,898	20,66	15,469
trnas-aga_2	0	0	0,432	0	0	0	0	0	0	0	0	0	0,114	0	0,001	0	0,074	0	0	0
LOC103187735	3,758	2,198	2,618	1,056	1,246	0,848	1,029	0,395	0,405	0,16	0,623	0,224	0,546	0,702	2,314	0,798	10,126	3,312	4,249	1,727
LOC103188985	0,6	0,65	0,134	0,157	0,226	0,114	0,285	0,146	0	0,048	0	0,028	0,231	0,326	0,126	0,148	0,389	0,447	1,133	0,526
LOC121849378	666,148	258,613	429,853	207,966	656,657	225,738	758,456	288,194	833,623	279,583	250,121	12,722	584,621	328,82	215,209	115,336	275,94	200,806	342,399	164,995
LOC103173097	20,286	8,421	86,439	35,044	61,317	12,352	40,277	14,747	36,429	8,062	19,124	7,789	69,172	27,747	63,273	28,498	227,898	102,521	35,754	16,234
tmc1	11,921	3,418	20,516	7,343	6,996	1,336	6,583	2,082	13,119	2,515	4,555	1,962	22,273	7,454	13,26	5,34	22,808	9,991	4,524	1,565
LOC103189538	1,877	1,682	12,398	14,354	4,028	2,087	1,986	2,163	1,679	1,116	27,247	42,325	1,764	2,569	5,16	9,043	2,052	3,524	1,038	1,543
LOC103176725	31,189	54,66	19,938	42,264	7,64	8,599	13,852	26,726	13,35	16,543	15,864	34,409	12,846	29,788	17,393	40,157	25,493	55,297	29,036	41,334
LOC103176641	0,454	2,084	7,797	17,288	147,115	131,42	0,513	2,115	1,282	2,858	0,239	0,992	168,106	311,162	1,076	4,616	1,335	5,525	1339,52	1318,626
LOC103179477	5,76	6,2	10,59	13,406	4,801	3,037	5,629	6,33	5,259	4,076	3,341	4,382	3,326	5,153	16,695	22,806	20,052	26,529	6,625	5,594
LOC121850542	1,769	0,216	16,023	0,888	5,455	0,233	11,877	0,44	3,419	0,181	9,441	0,468	1,078	0,362	22,01	3,554	2,156	0,481	0,836	0,202
syn2b	67,812	81,136	0,383	8,501	0,193	2,056	1,322	3,723	1,07	2,012	2,414	2,833	0,08	4,663	0,218	3,299	0,196	1,29	1,358	2,401
LOC121851695	6,013	0,18	23,773	0,223	2,464	0,006	8,857	0,218	3,54	0,069	2,679	0,126	1,619	0,07	8,704	0,68	7,331	0,13	3,036	0,093
LOC121850699	39,315	1,768	330,995	17,931	14,439	0,874	82,199	7,522	35,453	1,279	47,344	1,277	8,875	0,563	263,553	23,401	120,893	4,447	36,636	0,792
stx18	38,061	2,93	21,032	2,09	14,926	0,614	19,554	1,453	12,433	0,653	7,301	0,703	11,103	1,149	13,595	2	52,587	4,723	18,76	0,941
pomp	34,705	10,268	56,844	18,797	30,383	5,242	44,622	13,192	49,221	8,598	20,959	7,546	43,973	14,896	35,206	15,752	57,377	22,422	114,97	23,188
trnae-cuc_64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,082	4,136
adora2b	6,437	1,894	4,305	1,662	7,467	1,402	5,983	1,396	3,913	0,974	1,111	0,346	5,318	4,334	13,613	5,239	2,393	1,449	3,678	0,916

Appendix H

Full scope of the comparison between GeneAnalyst's *Gallus gallus* gene expression profiles and VastDB's gene expression profile using featureCounts.

Green color indicates the same expression pattern was found, red indicates different pattern and blue indicates newly found expression that wasn't present in VastDB's gene expression profile.

Link: https://figshare.com/s/77aec6d0cdd79ac9c61f

---

gene	heart	liver	lung
ADTRP	0.256	0.067	0.271
WDR59	1.626	1.293	7.282
ACBD6	1.235	0.763	3.46
RNF141	2.082	2.735	21.788
CACNG4	0.042	0.023	0.109
TRPC1	0.552	0.232	3.863
KCNQ5	0.02	0.029	0.608
MTIF2	2.359	2.04	8.331
RSL1D1	2.712	2.369	9.572
TLK2	2.624	2.33	18.205
VAV2	0.608	7.779	2.893
PARD6G	0.131	0.198	2.197
RPL18A	254.695	169.001	608.013
FHL3	2.896	0.468	5.559
RASSF2	5.333	7.682	53.128
TAB1	8.021	5.505	44.385
IGF2R	0.736	0.765	11.97
WHSC1	0.101	0.023	0.842
EXOSC2	7.08	6.525	22.792
MED13	1.498	4.136	8.505
DHX32	0.02	0.02	0.079
PKD2L1	0.002	0.024	0.3
IMMT	57.314	19.089	45.477
C7orf50	0.089	0.587	1.932
OXT	0.185	0.104	1.0
NDUFA9	4.478	1.574	4.28
ARL5A	2.696	1.295	11.283
NEK10	0.004	0.023	0.481
CPN1	0.946	55.425	4.893
NOX3	0.005	0.038	0.807
POLR3B	0.314	0.162	1.131
PPFIBP2	0.193	0.326	0.873
POPDC2	162.924	10.423	13.616
THEMIS2	1.774	25.449	17.053
GTF3C2	8.831	3.378	17.651
NDUFA9	4.478	1.574	4.28
C8orf37	0.64	0.417	3.418
PPIC	2.923	1.67	14.862
BAG1	0.335	0.194	1.411
RLTPR	0.69	0.888	4.686
SLC26A1	0.0	0.0	0.0
NDST1	5.72	2.557	16.791
EIF2B1	4.314	4.653	15.419
GPLD1	0.007	2.452	0.654
SLCO5A1	0.049	0.02	0.989
LTK	0.004	0.019	0.442
PALLD	2.224	0.195	5.457
FOXRED2	2.351	1.899	6.964
COLQ	0.154	0.274	0.663
PDCL2	0.042	0.025	0.327

gene	heart	liver	lung
SEC22A	0.185	0.366	2.125
MRPL50	32.435	23.285	27.175
ANKRD60	0.101	0.125	1.982
AVP	0.33	0.041	2.703
TXNDC15	1.494	0.955	5.715
OPRK1	0.0	0.016	0.412
PUDP	0.051	0.18	1.024
FLOT2	17.783	14.97	74.12
TMEM200C	0.537	0.153	5.647
CCZ1	0.417	0.612	2.506
NHP2	23.829	14.872	46.741
CEND1	0.002	0.003	0.0
FAM3D	0.019	1.266	9.006
SMPD2	5.484	14.044	21.477
SLC25A20	8.063	9.972	15.563
THADA	0.015	0.019	0.248
PAPSS1	0.724	0.408	4.046
CDK3	0.023	2.626	0.24
KLHL11	8.653	4.8	43.0
NECAB1	0.192	0.017	1.062
SASH1	0.189	0.072	1.707
PTPMT1	3.789	2.893	6.306
TRAPPC8	0.41	0.323	1.11
MSI1	0.429	0.338	1.452
CR2	1.708	1.057	17.356
TMEM45A	0.0	0.541	1.458
ADRB3	2.74	0.022	210.422
FBXO8	0.483	0.704	2.193
CTNND1	16.393	16.376	122.22
EML6	0.098	0.187	0.963
ALDH8A1	0.017	27.273	0.279
PCDHA1	0.0	0.0	0.0
ESRP1	0.001	0.024	0.63
KCNMA1	0.123	0.145	0.922
TNRC6C	4.068	1.397	18.373
PATZ1	0.446	0.175	1.725
MYL3	803.856	0.239	0.496
ERICH6B	0.008	0.02	0.608
RPGR	0.552	0.196	2.238
ANXA11	5.209	2.055	27.857
DSG2	5.927	0.731	3.994
LRRC72	0.047	0.041	0.117
ATP6V1H	0.296	0.297	2.006
JTB	59.14	16.483	23.325
SCAMP3	9.341	6.959	22.772
PRF1	3.347	3.405	38.31
ARRDC3	3.109	4.648	27.913
MSX2	0.176	0.034	1.052
SIPA1	18.31	15.731	119.812
FEZ1	15.808	0.673	9.771