U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

MSc
2.º
CICLO
FCUP
ANO

U.PORTO

NRfinder: A pipeline for the characterization of the number and type of nuclear receptors in a genomic sequence

Marcos António Pereira Domingues

FC

U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

NRfinder: A pipeline for the characterization of the number and type of nuclear receptors in a genomic sequence

Marcos António Pereira Domingues

Dissertação de Mestrado apresentada à

Faculdade de Ciências da Universidade do Porto em

Bioinformatica e Biologia Computacional

2021

MSc

NRfinder: A pipeline for the characterization of the number and type of nuclear receptors in a genomic sequence

Marcos António Pereira Domingues

Mestrado em Bioinformática e Biologia Computacional

2021

Orientador

Luís Filipe Costa de Castro, Professor Auxiliar, Faculdade de Ciências da Universidade do Porto

Coorientador

Pedro Gabriel Dias Ferreira, Professor Auxiliar, Faculdade de Ciências da Universidade do Porto

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, _____/_____/_____

Agradecimentos

Gostaria de agradecer ao Professor Doutor Filipe Castro pela ajuda, paciência e dedicação durante todo este percurso. Em segundo lugar gostaria também de agradecer ao meu coorientador Professor Doutor Pedro pela sua disponibilidade e auxílio. De seguida gostaria de agradecer também aos membros do grupo AGE do CIIMAR. Em particular gostaria de agradecer á Doutora Raquel Ruivo, pela paciência e pela ajuda que me de deu para melhorar a minha tese, á Doutora Elza Fonseca pelo conhecimento transmitido, ao Mestre André Fonseca pelas ideias e disponibilidade. Também gostaria de estender os meus agradecimentos ao Luís Alves pelo auxílio e ideias que me deu. Gostaria também de agradecer aos meus amigos pela ajuda e suporte que me deram para conseguir finalizar este percurso. Finalmente, e não menos importante, gostaria de agradecer á minha família por todo o apoio e conselhos que me deram ao longo destes anos.

Resumo

A superfamília de recetores nucleares consiste num grupo diversificado de fatores de transcrição exclusivo dos metazoários. Estes recetores desempenham um papel central no controlo homeostático de múltiplos processos fisiológicos dos metazoários. A sua estrutura é tipicamente modular composta por 5-6 domínios de conservação variadas. Os dois domínios mais conservados correspondem ao domínio de ligação ao DNA e ao domínio de ligação ao ligando. Através de métodos filogenéticos estes recetores podem ser classificados em nove subfamílias. Esta diversidade resultou de eventos de duplicação genética ao longo da evolução. A caracterização do repertório de recetores nucleares é essencial para o entendimento da evolução do sistema endócrino no conjunto completo de linhagens animais. No entanto, ainda não foi desenvolvido nenhum método automático para efetuar esta análise, que é particularmente relevante na altura da *"big data"* e "*big genomics"*. Deste modo, o objetivo desta tese é desenvolver uma pipeline, o *NRfinder*, para automatizar esta análise e facilitar a caracterização dos recetores nucleares nos genomas dos animais. Para além disso, esta pipeline foi integrada numa interface web, de forma a ficar disponível online. Assim, esta pipeline foi testada usando os genomas de *Mus musculus*, *Danio rerio* e *Drosophila melanogaster.* De seguida foi feita uma comparação dos resultados obtidos com os programas *Augustus* e *Exonerate*. Apesar de ter tido uma performance ligeiramente inferior às outras duas ferramentas testadas, foi possível identificar e classificar a maior parte dos recetores nucleares encontrados.

Palavras-Chave: Recetores nucleares; Metazoários, Bioinformática, Previsão de Genes

Abstract

The nuclear receptor superfamily consists of a highly diverse group of transcription factors exclusive to metazoans. These receptors perform a central role in the regulation of most physiological processes in metazoans. Nuclear receptors (NRs) share a modular structure composed of 5-6 domains of varied sequence conservation. The two most conserved are the DNA binding domain (DBD) and the ligand-binding domain (LBD), which are functionally distinct. The classification into nine subfamilies is the result of the phylogenetic approaches. This ample diversity was the result of gene duplication events throughout evolution. The characterization of the precise gene repertoire of NRs in different metazoan lineages is fundamental to understanding the evolution of endocrine systems. However, there is still a lack of an automated method to perform this analysis, particularly relevant in the age of big data and big genomics. Therefore, the objective of this thesis is the development of a comprehensive pipeline, the *NRfinder*, to automate these analyses and facilitate the characterization of nuclear receptors in animal genomes. The *NRfinder* was developed using homology-based methods. Furthermore, this pipeline was integrated with a web interface, to become publicly available online. We tested our pipeline with three established animal models: *Mus musculus*, *Danio rerio* and *Drosophila melanogaster*. After analysing the results, we compared them with the *Augustus* and *Exonerate* programs. Although slightly outperformed, *NRfinder* was able to correctly identify and classify the majority of the nuclear receptors present in the tested species. Future steps will include the development of a database of nuclear receptors to further improve our pipeline.

Contents

Agradecimentos i
Resumo ii
Abstract iii
Contents iv
List of Figures v
List of Tables v
Acronyms vi
Abbreviations vi
1-Introduction 1
1.1-Genome Sequencing 1
1.2-Genome assembly and Annotation 4
1.3-Sequence analysis 6
1.4-Metazoa 8
1.5-Nuclear Receptors 11
1.6-Gene Duplication and Loss 15
1.6-Objectives 18
2-Methods 20
2.1-Brief overview of the *NRfinder* 20
2.2-Detailed description of each step of the *NRfinder* 20
2.5-Evaluation 21
2.6-Benchmarking 22
3-Results 23
3.1-Web Service 23
3.2-*NRfinder* Performance 25
3.3-Benchmarking 28
4-Discussion 32
4.1-Performance Analysis 32
4.2-Limitations 32
4.3-Benchmarking 33
5-Conclusion and Future Perspectives 35
6-References 36
Appendix 1 46

List of Figures

Figure 1.Illumina sequencing workflow 2
Figure 2. Principle of single-molecule, real-time (SMRT) DNA sequencing 3
Figure 3. Schematic representation of the nanopore sequencing 4
Figure 4.Scheme of the BLAST algorithm 7
Figure 5.Maximum likelihood tree topology of the Metazoa 11
Figure 6.Modular structure and biding of the nuclear receptors. 13
Figure 7. The three potential fates of the duplicated genes 16
Figure 8. Phylogenetic tree of the Metazoa and Choanoflagellates 18
Figure 9. Schematic representation of the NRfinder 23
Figure 10.Overview of the NRfinder homepage 24
Figure 11.Example of the output of the NRfinder. 25
Figure 12.Output with the results of the number of receptors of each subfamily in each tested species 26
Figure 13.Results of the performance of the *NRfinder* 28
Figure 14. Results of the *Augustus*, *Exonerate* and *NRfinder* in *M. musculus* 29
Figure 15. Results of the *Augustus*, *Exonerate* and *NRfinder* in *D. rerio* 30
Figure 16. Results of the Augustus, *Exonerate* and *NRfinder* in *D. melanogaster* 31

List of Tables

Table 1. Gene prediction algorithms. 5
Table 2. BLAST-based program 7
Table 3. Nuclear receptors repertoire in various species of animals 14

Acronyms

BLAST -Basic Local Alignment Tool

COOH -carboxyl group

CSS - Cascading Style Sheets

DAX-1 - dosage-sensitive sex reversa

DBD - DNA binding domain

HMMs - Hidden Markov models

HRES - Hormone Response Elements

HSP -High Scoring Pairs

HTML - HyperText Markup Language

LBD - Ligand-Binding Domain

NGS - Next Generation Sequencing

PCR - Polymerase Chain Reaction

RXR - Retinoid X receptor

SHP - Small Heterodimer Partner

SMRT - Single-molecule real-time

SVMs – Support Vector Machines

WGD – Whole Genomic Duplication

ZMW - Zero-mode waveguide

CIIMAR- interdisciplinary Centre of Marine and Environmental Research

Abbreviations

DNA - Deoxyribonucleic acid

mRNA – Messenger RNA

cDNA -Complementary DNA

PHP - Hypertext Preprocessor

1-Introduction

1.1-Genome Sequencing

The first sequencing techniques emerged in the mid-1970s and were developed by Frederick Sanger and colleagues, who developed the Polymerase-Chain Reaction (PCR)-based chain termination method, and by Maxam and Gilbert, who created the chemical sequencing [1,2,3]. Due to its greater efficiency and use of fluorescently labelled compounds, the chain-termination method, or Sanger sequencing, became the most popular first-generation sequencing technique [1,4]. In the following years, several improvements were made to this technique, such as automation and simultaneous sequencing of distinct fragments [1,4]. As a result, the Sanger technique was used to sequence the first human genome, which started in 1993 [1,4,5]. However, it took about a decade and a lot of resources to complete the first draft genome, and the need for faster, cheaper, and high thought techniques led to the emergence of next-generation techniques (NGS) [1].

Since the sequencing of the first human genome, several new technologies have been developed. Second-generation sequencing technologies arose applying the same general sequencing by synthesis approach but using different strategies: including a library preparation step, hybridization of fragments onto a surface and sequencing using fluorescently labelled nucleotides [6]. The first step involves the fragmentation of the DNA sample, binding of distinct adapters to the ends of each fragment (Figure 1A) [6,7]. The specialized adapters allow the hybridization of the DNA fragments onto a surface. Among the various second-generation platforms, the Illumina platforms are the most currently used. In this platform, the amplification and sequencing take place in a glass surface (flow cell), in which two types of adapters, complementary to the ones added to the DNA fragments, are fixed [8,9]. The prepared DNA fragments are loaded onto the flow cell and bind to the first type of adaptors (Figure 1B) [8,9]. Then the DNA polymerases synthesize a complementary strand to the hybridised fragments [8,9]. The double-stranded fragments are denatured, and the original strands are removed [8,9]. After this, the single strand bends over and binds to the second type of adaptor [8,9]. The polymerases will then create another complementary strand, forming a double-strand bridge (bridge amplification) [8,9]. This bridge is denatured, the novel strand forms an additional bridge, and the process is repeated generating several fragment clusters [8]. This elicits a signal, during sequencing, intense enough to be differentiated from the background noise [6] (Figure 1C). After the amplification, the fragments will be denatured, and reverse strands are removed [9]. In the sequencing step, the polymerases synthesize a new

strand in each fragment by incorporating modified nucleotides [8,9]. These nucleotides carry a fluorophore and have their 3' end blocked which prevents the insertion of the adjacent base [8,9]. The incorporation of these nucleotides leads to the emission of light in each cluster corresponding to the introduced base [8,9]. After this, the fluorophore and the 3' blocker are removed, and the next nucleotide is incorporated [8,9].

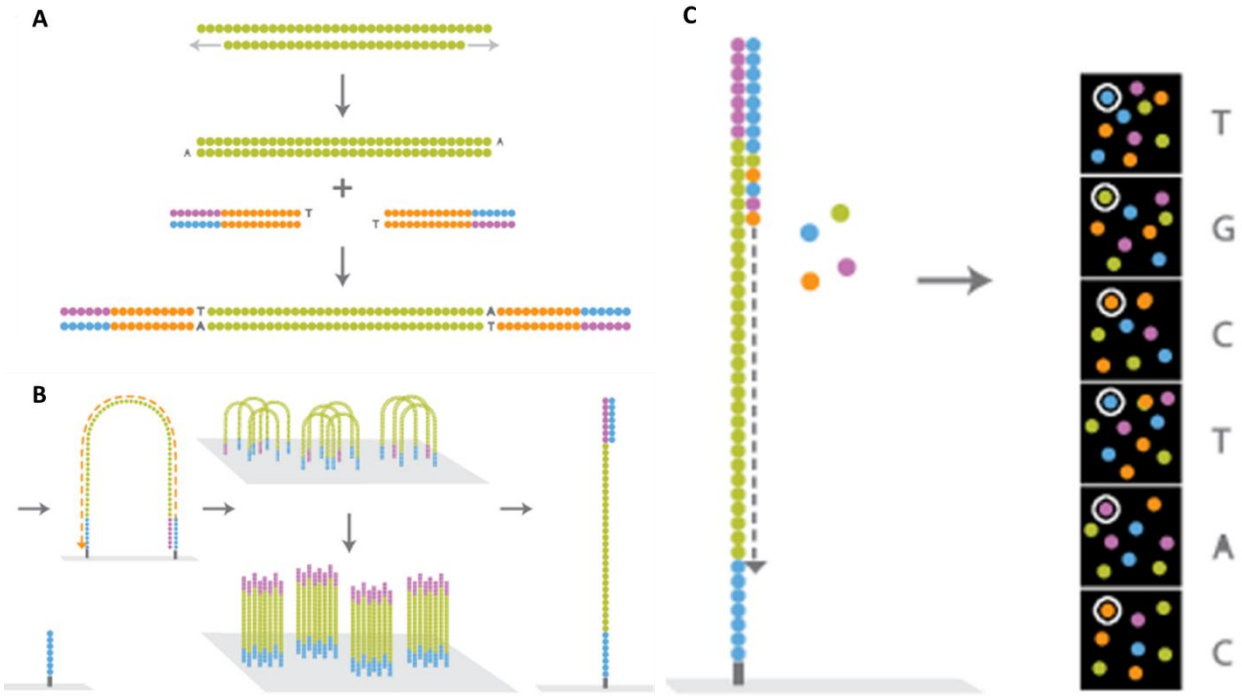


Figure 1.Illumina sequencing workflow. (A) Library Preparation (B) Bridge Amplification (C) Cyclic reversible terminator sequencing. Adapted from [10].

Although second-generation sequencing enabled the reduction of costs and the increase of the sequence data these methods have some limitations [1,11]. The main drawback is the reduced size of the produced reads, which makes the assembly *de novo* of genomes difficult [11]. As a result, several genes, and regions of interest are not properly assembled [11]. Because of this, new methods for sequencing long DNA molecules, known as "third-generation sequencing", were developed [9]. The main difference between third generation and second-generation technologies is that the former does not have an amplification step for cluster generation [9]. The two main methods of third-generation sequencing are Single-Molecule Real-Time sequencing (SMRT), created by PacBio in 2010, and Nanopore sequencing, developed by Oxford Nanopore [11]. The Single Molecule Real-Time approach is based on the real-time monitoring of the continuous incorporation of fluorescently labelled nucleotides by the DNA polymerase [8,9]. This process is carried out in zero-mode waveguide (ZMW) microwells and at the bottom of each microwell is fixed a polymerase [12]. In the library preparation, the DNA is sheared, and the fragments are bound with hairpin-adaptors [8,12]. These adapters bind the two strands of each fragment forming a circular DNA molecule, SMRTbell [8,12]. After preparing the DNA libraries, the SMRTbells are

loaded to the ZMW microwells [8,12]. In each microwell, the DNA polymerase binds to one of the adapters in SMRTbell and starts incorporating fluorescent labelled nucleotides [12]. Upon incorporation, the nucleotides emit a light signal that is recorded by a detector [8]. After this, the fluorophore is removed, and the next nucleotide is added [8]. Due to the circular shape of the fragments, each can be sequenced multiple times [12]. This generates long circular molecules that by cutting into regions originate subreads [12]. Through the alignment of these subreads a consensus sequence can be obtained which allows increasing the accuracy of this method [12].

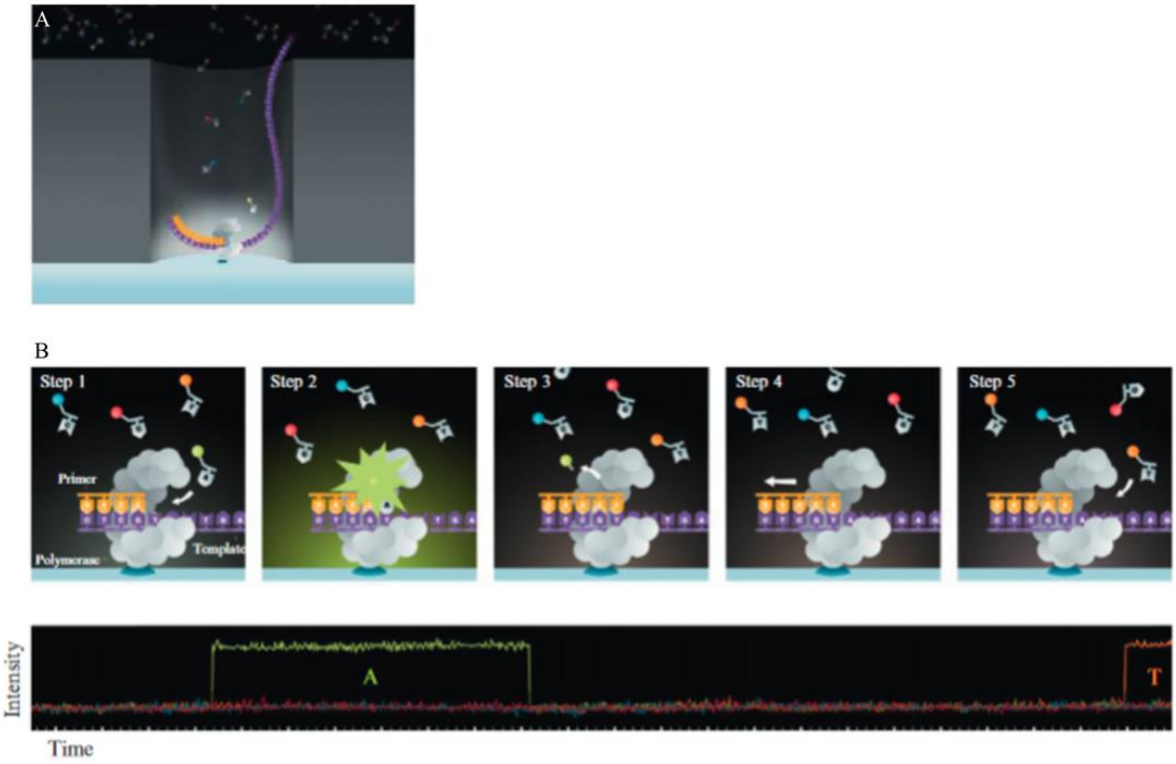


Figure 2. Principle of single-molecule, real-time (SMRT) DNA sequencing. (A) DNA polymerase is fixed at the bottom of the zero-mode waveguide (ZMW) and binds to the circular DNA template. (B) Representation of the Single Molecule Real Time (SMRT) sequencing steps (top), and emission spectrums corresponding to the incorporated fluorescence labelled nucleotides (bottom). Adapted from [13].

Regarding Nanopore sequencing (Figure 3), this technique uses nanopores embedded in an electrically resistant membrane [14]. This membrane is immersed in an electrolytic solution and the application of a potential creates an ionic flux through the nanopores. The transition of molecules through the nanopore leads to characteristic disruptions of the flux [14]. In the library preparation, after the fragmentation of the DNA, a hairpin adapter attaches to one end of each fragment and a motor protein attaches to the other [8]. In the sequencing, the DNA fragments are unfolded by the motor protein so that only a single strand of DNA passes through the nanopore [8]. Thus, during the sequencing of each fragment, one of the DNA strands, the adapter, and the other strand of DNA pass through the

pore [8]. By sequencing the two complementary strands consensus sequences are generated, called "2D reads" [8].

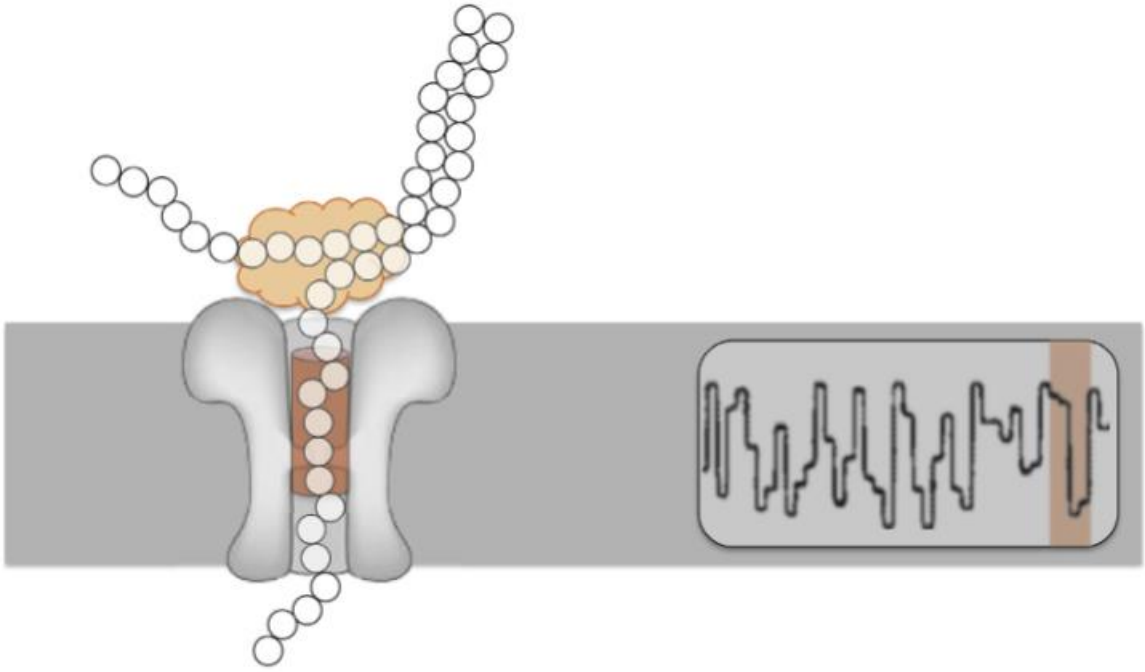


Figure 3. Schematic representation of the nanopore sequencing. A nanopore is inserted on a membrane electric resistance membrane. This membrane is flooded with an electrolyte solution and ions pass through the nanopores. The application of a potential generates an ion current through the nanopore. Then a “motor protein” bonded to the DNA template, in the proximity of the nanopore, ratchets the strands. The disruptions of the current, caused by the passage of the strand through the nanopore, are measured. Adapted from [14].

1.2-Genome assembly and Annotation

After the sequencing, it is necessary to assemble the reads to reconstruct genomic sequence [15]. There are two types of assembly: mapping and assembly, and *de novo* assembly [16]. In the mapping and assembly approach, the reads are aligned against a reference genome prior to assembly [16]. When no genomic reference is available, the de novo assembly is performed [16]. This approach consists in joining overlapping reads to form “contigs”, then the contigs are joined forming scaffolds and subsequently these form chromosomes [17]. After genome assembly, its quality is evaluated. For this, several statistics can be used, being N50 the most common one [16]. This metric is calculated by adding the lengths of the contigs, sorted in descending order, until this is greater than half the total length of all the contigs [11]. Thus, the N50 assesses the contiguity of the assembly and the higher its value the less fragmented the assembly is [11]. After this, if the quality of the assembly is within the minimum required levels, the annotation is performed. Genome annotation involves the identification of the elements present in the raw genomic sequence. Among these it highlights the genes, since they code the proteins, which are the main functional and structural

units of the cells [18]. Several methods have already been developed to determine the location and structure of the genes [11]. These methods can be classified as extrinsic, ab-initio or combiners [11] (Table 1). The extrinsic methods are based on the homology with transcript or protein sequences, from the same or closely related species [11]. Despite their usefulness for gene identification, these methods have limited performance when these types of sequences are not available [19]. Furthermore, they can provide inaccurate information about the structure of the gene [11]. The *ab initio* methods use the information obtained from content sensors and signal sensors to train statistical models such as hidden Markov models (HMMs) and support vector machines (SVM) [11]. Signal sensors identify functional sites, like start and stop codons, polyadenylation sites, splice sites [20,21]. Content sensors differentiate coding and non-coding regions through statistical properties such as nucleotide composition and codon usage [20,21]. Some of these predictors can use external information to increase their accuracy [20]. Combiners use external information with a set of predictors to either select the best gene model of the possible gene locus or can integrate the external information to modify the gene prediction [11,20]. Although several methods have already been created, they still present some difficulties in the prediction of genes in eukaryotic genomes [22]. This is due to the high complexity of these genomes that, unlike prokaryotes, are mostly composed of non-coding regions, their genes are very distant, and coding regions (exons) are interspaced with non-coding (introns) [22]. In addition to the complexity of the genomes, the annotation of fragmented genomes is difficult and not very accurate, which can result in the propagation of errors [19,22]. Thus, despite the large increase in the number of sequenced genomes since the development of NGS methods, the process of annotation has not increased in the same way [22].

Table 1. Tools and pipelines used for gene predictions tasks.

Extrinsic	BLAST	Set of local alignment tools based on the Karlin-Altschul statistics to perform databases searches [23].
	BLAT	Alignment algorithm that builds and index with the target sequence to search for homologous regions [24].
Ab-initio	Augusuts	Eukaryotic gene predictor based on hidden Markov chain models. It can also accept EST and protein data [25,26].
	GeneMark	Self-training algorithm for eukaryotic gene prediction. [27,28]
Combiners	MAKER	Pipeline that combines the information obtained from the EST and protein alignments (BLAST, exonerate) with ab-initio gene predictions (Augustus, Snap GeneMark-EP) [29,30].
	BRAKER	Pipeline that integrates protein or transcriptomic data in the training and prediction of Augustus and GeneMark [31,32].

1.3-Sequence analysis

Sequence comparison is one of main techniques used in Bioinformatics for the analysis of newly sequenced genes [33]. These methods infer homology based on the similarity between the sequences [34]. The sequence alignment is the most commonly used method to perform this comparison [33]. This process consists in arranging two sequences to compare each of their residues in the same position. During evolution, sequences from the same ancestor diverge from each other due to the accumulation of mutations (e.g., insertions, substitutions, deletions) over time [16]. However, some regions in these sequences may remain preserved, due to possessing a functional or structural key role [33]. Thus, through the alignment of the sequences it is possible to infer their degree of conservation and to establish an evolutionary relation between them [16,33]. If the alignment of two sequences presents significant similarity, it is most likely that they are homologues and thus have the same function [16]. There are two types of alignment: global and local. Global alignment consists in aligning two sequences from beginning to end, while local alignment consists in aligning the most similar regions between two sequences [16,34]. The first algorithms created to perform this type of alignments were the Needleman-Wunch algorithm (global alignment) [35], in 1970, and the Smith-Waterman algorithm [36], in 1981. Both algorithms use the dynamic programming approach to perform the alignments. Through this method, they can obtain the best alignment between two sequences [33]. However, they perform an exhaustive search, which makes them impractical to use when searching in a database composed of many sequences. Thus, it was necessary to develop faster algorithms to perform these searches, known as heuristic algorithms. Yet, although they can perform the searches faster than the dynamic computation-

based algorithms, heuristic algorithms are less accurate [33,34]. The most famous heuristic algorithm is the Basic Local Alignment Tool (*BLAST*) [33].

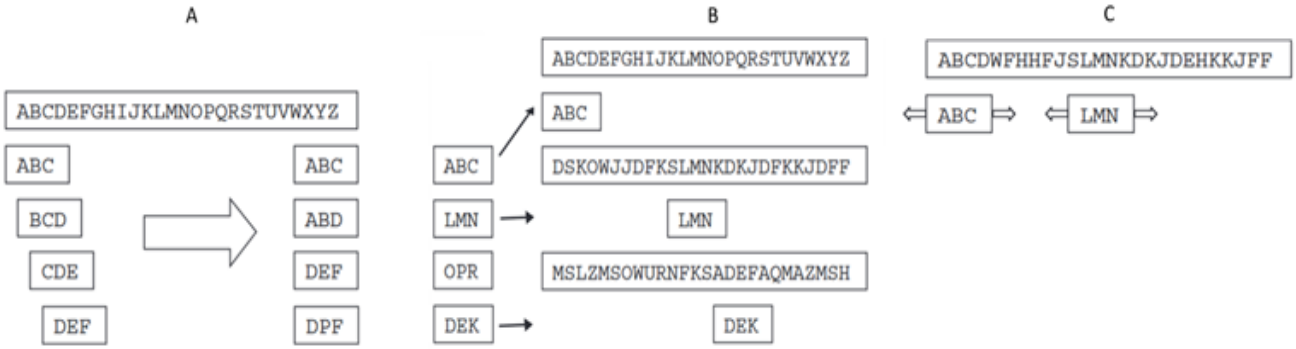


Figure 4.Scheme of the *BLAST* algorithm [37]. (A) Generation of a list of words with length equal to three from the query. The list is then expanded with all high-scoring matching words, keeping only those that score more than the threshold T. B) The word list is compared to the sequence database to find exact matches. (C) For each word match, the alignment is extended in both directions to generate alignments that score higher than the score threshold S.

BLAST was developed by Stephen Altschul and colleagues in 1990 [23], and it performs the alignment through three steps (Figure 4). In the first step, it retrieves all the words or sub-sequences of size w (parameter of the algorithm) from the query sequence [34,37]. For each of the obtained words, it finds all the matching-words and assigns them a score [34,37]. In the second step the database is searched for exact matches of the words with a score higher than a threshold value T [33,34,37]. Then in the third step, the found matches are then extended in both directions until an alignment score is lower than a threshold [33,34,37]. The output generated by *BLAST* consists in a list with the alignments between the query and database sequences and the corresponding *percentage identity*, *E-value*, *bit score*, *raw score* values [37]. The two measures that provide a better indication for the inference of homology are the *E-value* and *bit-score* [38]. The *E-value* corresponds to the number of alignments with a score greater or equal to the observed one, that could be found in a search against a random database with the same composition [37]. However, one downside of this measure is that it depends on the size of the database [33,37]. The *bit-score* is obtained through the normalization of the *raw score*, and the higher this value the more significant it is [33,37]. Furthermore, this measure is independent of the sequence length and of database size, which allows to compare two alignments obtained from databases with different sizes [33,37]. There is an ample selection of blast programs that can be differentiated by both: the type of sequence that is used as a query and the type of databases where they search (Table 2). Despite its popularity, *BLAST* does not use any splice site models, thus the edges of the obtained alignments are not very precise [20]. To overcome this several splice aware algorithms were developed, such as *Exonerate* [39]. This program uses dynamic programming and heuristics

methods with splice-site models, which allow it to improve the prediction of the splice sites and exon boundaries [20,39].

Table 2. BLAST-based program. Adapted from [37].

BLAST Programs	
Program	Description
BLASTN	Uses nucleotide query to search nucleotide databases
BLASTP	Uses protein query to search protein databases
TBLASTN	Uses a protein query to search translated nucleotide databases
BLASTX	Uses translated nucleotide query to search protein databases
TBLASTX	Uses translated nucleotide query to search translated nucleotide databases databases

All the sequence alignment methods described above are based on the sequence comparison. However, this analysis can be also performed by using probabilistic based methods such as Hidden Markov Models (HMMs) [34, 40]. The HMMs involve two types of information, the observable symbols of the sequence, and the hidden states to which each of the sequence symbols are assigned to [34, 40]. When analysing a symbol within a determined state, a probability will be emitted, which determines if the next symbol corresponds to the same stat or not (transition probability) [34,40]. These methods have also been used in other sequence analysis problems, such as gene prediction [34,40]. One example of this is *Augustus*, which is a gene prediction program of eukaryotes based on HMMs [25]. Originally it only used genomic sequence information to make the prediction. However, since its development, some extensions enabled it to use external information, such as ESTs, protein, and nucleotide sequences, to improve its predictions [26,41,42]. The most recent expansion, the *AUGUSTUS-PPX*, allowed it to integrate protein family signatures to perform the identification of members of the protein family of interest in a genomic sequence [42]. These signatures correspond to block profiles that are generated from multiple alignments. These profiles consist in a set of position-specific frequency matrices, that describe the amino acidic distribution in an ungapped and highly conserved section of a MSA (block) [42].

1.4-Metazoa

The current classification of life consists into two super kingdoms Prokaryotic and Eukaryotic, which are divided into 7 kingdoms [43]. The Prokaryotic super kingdom contains the Archaebacteria and Bacteria kingdoms, while the Eukaryotic super kingdom contains the Metazoa, Plantae, Fungi, Protozoa and Chromista kingdoms [43]. The Metazoa, or animal

kingdom, will be the focus of the present thesis. The organisms that form this kingdom are multicellular, which means that their body is composed of multiple and specialized cells [44,45]. This trait differentiates them from the prokaryotes and protozoa, which contain unicellular life forms [44,46]. However, plants and fungi are also multicellular organisms, thus there are other traits that need to be considered to categorize an organism as animal [46]. One difference between plants and fungi and animals is how they obtain their energy. Plants obtain it through photosynthesis and fungi by decomposition of the organic matter [44,46]. Animals, on the other hand, consume other organisms, either dead or alive, or parts of them [44,46]. Furthermore, animals are also characterized for being able to move [44,46], at least during larval stages, like sponges, corals, and bivalves, that become sessile in their adult form [44,46]. Regarding reproduction, animals, plants, and fungi can generate descendants through sexual reproduction, although the production of sperm and egg cells is exclusive to animals [44,46]. Therefore, it is necessary to consider one last factor to categorize animals, which is that all of them have the same common ancestor [40,42]. Thus, an organism lacking a common animal trait can still be considered one if it shares a common ancestor with another animal [46].

The organisms to which metazoans share the closest common ancestor are the choanoflagellates [44-46]. This group consists of unicellular organisms that inhabit marine and freshwater environments and can form colonies or remain solitary [44-46]. They possess a collar, composed by a ring of tentacles with a flagellum in the middle, that resembles the choanocytes, characteristic cells of sponges [44]. The sister-group relationship between metazoans and choanoflagellates is supported by morphological and molecular analysis [45-47]. Because of this, it is believed that the ancestor of the metazoans was a cell colony that developed close contacts between its cells enabling the exchange of nutrients between them and, subsequently, the differentiation of cells and cellular functions [45].

Metazoans comprehend two groups of animals, the bilaterians and non-bilaterians [46]. The non-bilaterians correspond to animals that do not possess a symmetry plane dividing their body [44,46]. They are composed of the Porifera, Placozoa, Ctenophora and Cnidaria phyla [45,46,48]. The Porifera is the group of sponges, which are the most primitive animals [44-46]. They do not possess any organs, nervous systems, and muscles [44-46]. Instead, their body is composed of thousands of pores that are part of a complex network of tunnels and chambers, which contain [44-46]. These structures contain cells, designated choanocytes, which through the reduction of the water flow, allow the cells to capture the food and oxygen transported by the water [44]. The Placozoa is composed of only three marine species, the *Trichoplax adhaerens*, *Hoilungia hongkongensis* [49], and *Polyplacotoma*

mediterranea [50]. These have a disc-shaped body, with two germ layers (diploblastic) and six types of cells [46,48]. Like sponges, they lack nerve cells, mouth, muscles, and gut [44,46,48]. The organisms of the remaining two phyla, the cnidarians, and ctenophores, are also diploblastic, although contrary to the previous ones, they have a radial symmetry, nerve cells around their body, muscles, gut, mouth and, some species in the adult form, tentacles [46,48].

The bilaterian body is divided by a single line that splits it into symmetrically opposed images: the right and left sides [44,46,51]. Furthermore, they have a third germ layer, the mesoderm, during their embryological development, which enabled the formation of specialized organ systems and condensed nervous systems [51]. They are divided into two groups, Nephrozoa (deuterostomes, protostomes) and Xenacoelomorpha [52]. The Xenacoelomorpha are proposed to be the earliest bilaterians and are characterized for having an incomplete gut with a midventral mouth, direct development (no larval form) and lack of excretory organs [51,52]. The position of this group is not very well defined, with several conflicting hypotheses that put them as the sister group of the Deuterostomia, or within it, or even as the sister group of Nephrozoa [46,51,52]. Bilateria are divided into deuterostomes and protostomes. These two groups were defined based on the order in which the mouth and anus are formed during the embryonic development [46,51]. In deuterostomes, the blastopore originates anus, and the mouth is formed secondarily, while in protostomes it was thought that the blastopore formed the mouth, and the anus was developed afterwards [46,51]. However, there are some reported cases in which the blastopore does give origin to the mouth and cases where the blastopore originates the anus [46,51]. Protostomes are divided in two clades, the Ecdysozoa (e.g., arthropods and nematodes) and the Spirallia (e.g., molluscs and annelids), and the Deuterostomia are divided in Ambulacraria (e.g., echinoderms) and Chordata (e.g., vertebrates) [46, 52].

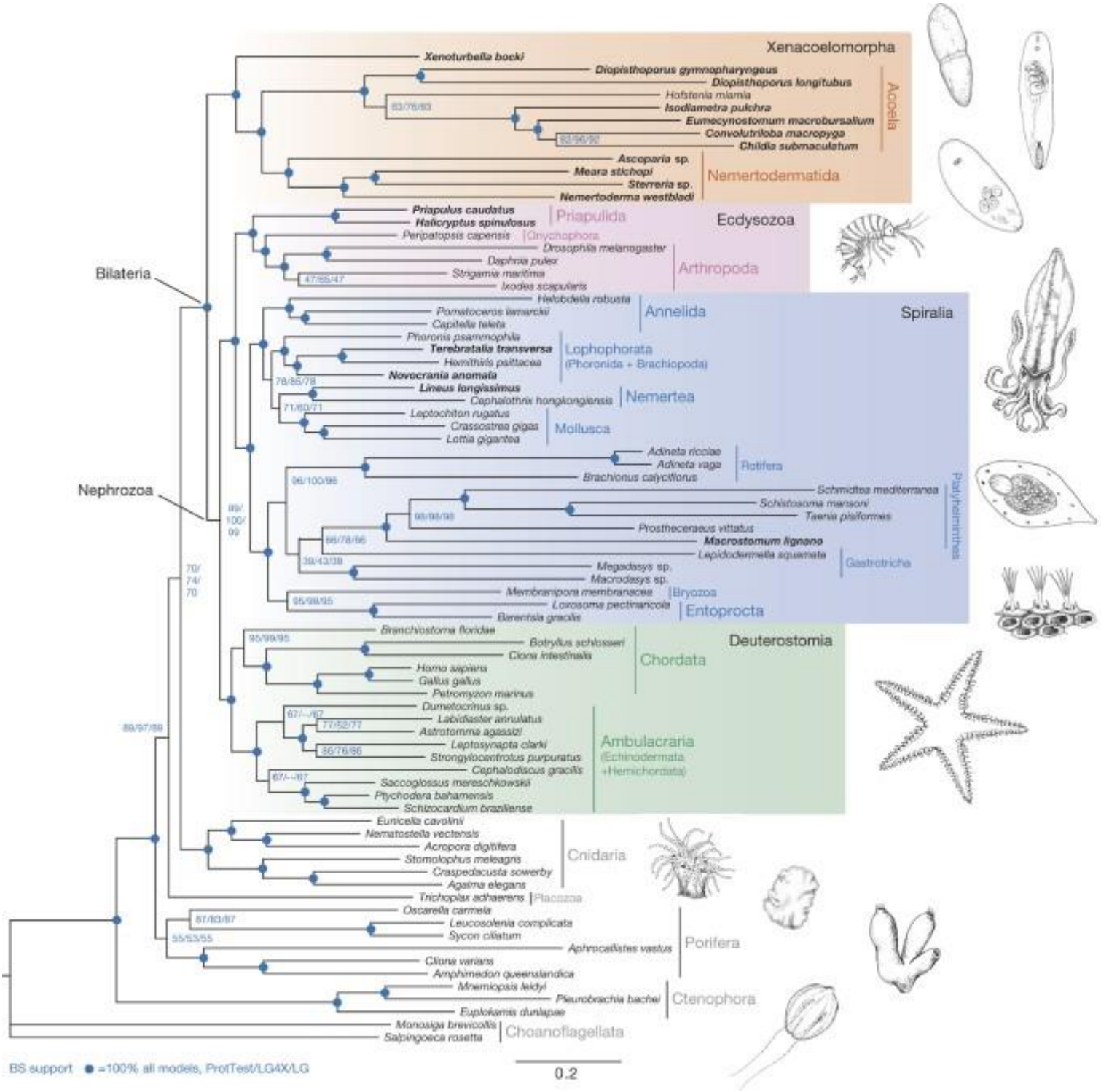


Figure 5.Maximum likelihood tree topology of the Metazoa [52].

1.5-Nuclear Receptors

Nuclear receptors constitute a superfamily of transcription factors that are responsible for regulating the expression of genes essential for most of the physiological processes of animals: such as development, metabolism, and reproduction [53-55]. The first nuclear

receptors to be cloned were the glucocorticoid receptor and the steroid receptor by Ron Evans and his colleagues and by Geoffrey and Pierre Chambon in 1985, respectively [53,56,57]. Nuclear receptor activity is mostly modulated by small hydrophobic molecules that diffuse through the cell and interact with the ligand-binding cavity [53,54]. Thus, nuclear receptors function as a direct link between the signalling processes and the regulation of gene expression [55,58]. Additionally, there are also some receptors, designated as orphan receptors, that do not have ligands, or their ligands are yet to be discovered [53-55]. Nuclear receptors may act as monomers, homodimers, or heterodimers, being the Retinoid-X-Receptor (RXR) the most common heterodimeric partner [54,59,60]. Due to their central role, nuclear receptors are also associated with several diseases like cancer, infertility, obesity because of dysregulation of their activity [54,55,61].

Nuclear receptors share a common modular structure usually composed of 5 to 6 domains [46,53,54,62] (Figure 6A). Among these domains, the two most important are the DNA-binding domain (DBD), absent from the dosage-sensitive sex reversal receptor (DAX-1), and the ligand-binding domain (LBD), absent from the small heterodimer partner receptor (SHP) [53,60,63]. The DBD, the most conserved domain, binds to specific regions in the DNA designated hormone response elements (HRES) [46,53,64]. This domain is composed of cysteine-rich zinc finger motifs, which are characteristic of nuclear receptors, two α helices, and a carboxylic acid (COOH) extension [46,53,54,65]. Additionally, there sequence elements, called P, D, T and A boxes, are also present [46,53,54]. These elements influence the specificity towards the HRES (P box), are involved in the formation of a dimerization interface between the DBDs of the two nuclear receptors and in the interaction between the DNA backbone and the residues flanking the DNA recognition sequence [46,53,54]. The LBD is in the C-terminal region and its structure is composed of the following components: a dimerization interface which is involved in the formation of homodimers or heterodimers with other receptors [46,53]; a ligand-binding pocket responsible for interacting with ligands [46,53,54]; a co-regulator binding interface, which associates with co-regulators to control gene expression [46,53]; and, an activation function (AF-2) that mediates the transactivation in a ligand-dependent manner [46,53]. Between the DBD and LBD is the D domain. This domain has a low degree of conservation and functions as a hinge, enabling the DBD and LBD to have different conformations [46,53,54]. In the N-terminal is located the A/B domain, which is the less conserved domain [46,53,54]. This domain is often the target of posttranslational modifications and alternative splicing events [46,53]. Additionally, it contains an activation function (AF-1) responsible for interacting with coregulators; the AF-1 was also

suggested to be able to activate constitutive transcription in the absence of a ligand in some nuclear receptors [46,53,54,66].

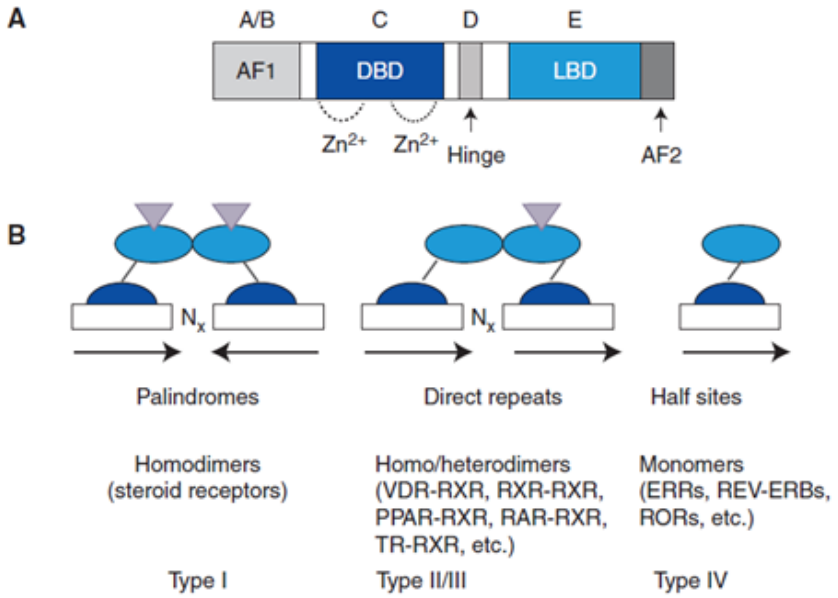


Figure 6.Modular structure and biding of the nuclear receptors. A) Structural organization of the nuclear receptors in five domains. B) Three types of HREs configurations and dimerization patterns of the nuclear receptors [65].

Although nuclear receptors share a common structure, their mode of action is very diverse. Unliganded nuclear receptors are generally in the nucleus bond to HREs, repressing the expression of their target genes in interaction with corepressors [46,53]. The binding of the ligand induces a conformational change in their ligand-binding pocket, which results in the dissociation of the corepressors, the recruitment of coregulators and the initiation or promotion of the expression of their gene targets [59,63,65]. However, not all unliganded receptors follow this mode of action. The steroid receptors, for instance, in the absence of a ligand, are in the cytoplasm, and the interaction with their ligands results in a conformational change followed by the migration to the nucleus to start the transactivation [46,59,63]. In addition to direct ligand activation, some nuclear receptors may be regulated by post-translational modifications, like phosphorylation; this activation mechanism is common, but not exclusive to, orphan receptors that do not accommodate ligands in the binding cavity [68]. Regardless of the type of nuclear receptor, an essential step for their action is the recognition and binding to the HREs. These sequences consist of two hexametric core half-site motifs separated by a variable number of nucleotides [53,60]. These motifs originated from the same DNA sequence, RGGTCA (R = A/G) [46,60]. However, the occurrence of mutations, extensions, duplications, and different orientations of this motif led to the formation of specific response elements for each nuclear receptor [46,60]. Before binding to the HREs, they can form homodimers, heterodimers (usually with the RXR) or monomers [53,60,63] (Figure 6B). Through these features, nuclear receptors can be classified in four classes [46,65]. The first class is composed of receptors

located outside the nucleus that, after binding with the ligand, form homodimers and bind to inverted HREs [46,65]. The receptors of the second class are in the nucleus and recognize either direct or indirect HREs [46,65]. They can bind to these by forming heterodimers with the RXR without the presence of a ligand [46,65]. The receptors of the third class are similar to the ones in the first class, although they bind to direct HREs [46,65]. Finally, the receptors of the fourth-class form monomers and bind to half-sites of the HREs [46,65].

Table 3. Nuclear receptor repertoire in various species of animals. Adapted from [46].

Chordata	*Branchiostoma floridae*	33
	Ciona intestinalis	17
	Danio rerio	73
	Xenopus tropicalis	52
	Gallus gallus	44
	Anas platyrhynchos	42
	Tursiops truncatus	47
	Mus musculus	49
	Homo sapiens	48
Echinodermata	*Strongylocentrotus purpuratus*	33
Mollusca	*Biomphalaria glabrata*	39
	Lottia gigantea	33
	Crassostrea gigas	43
Annelida	*Capitella teleta*	27
Rotifera	*Brachionus koreanus*	32
	Brachionus plicatilis	29
	Brachionus rotundiformis	32
	Brachionus calyciflorus	40
Arthropoda	*Drosophila melanogaster*	21
	Daphnia pulex	25
	Daphnia magna	26
Nematoda	*Caenorhabditis elegans*	284
	Caenorhabditis briggsae	268
Cnidaria	*Nematostella vectensis*	17
Placozoa	*Trichoplax adhaerens*	4
Ctenophora	*Mnemiopsis leidy*	2
Porifera	*Amphimedon queenslandica*	2

The analysis of various genomes of plants, fungi and choanoflagellate (the proposed phylogenetic sister clade to metazoans) did not identify any nuclear receptors, which means they are exclusive to metazoans. Within the metazoan, the number and type of nuclear receptors varies between lineages (Table 3) [46,59]. Regarding the origin of the nuclear receptors, it was previously thought they evolved from an orphan receptor, due to the phylogenetic analysis of this superfamily [55,69,70]. This analysis grouped nuclear receptors

into six subfamilies, in which orphan receptors were scattered throughout and the position of nuclear receptors was not associated with the nature of their ligand, since there were receptors with similar ligands in different subfamilies [46,70]. However, the evolution from an orphan receptor implies that the ability to regulate the gene expression by binding to a ligand was independently acquired multiple times, which is not parsimonious [46,71]. The acquisition of new data and development of better algorithms allowed us to overcome some barriers, like the lack of knowledge about the nuclear receptors of basal metazoans and the uncertainty about the root of the nuclear receptors tree [71]. The identification of two receptors in the Porifera *Amphimedon queenslandica* genome implies that the common ancestor functioned as a ligand receptor, and, during evolution, some receptors lost the ability to bind to ligands [46,71].

1.6-Gene Duplication and Loss

Gene duplication is considered one of the main mechanisms underlying the evolution of organisms. This process can occur through unequal crossing over, retrotransposition, and whole-genome duplication (WGD) events [16,72]. Unequal crossing over originates tandem repeated sequences, and depending on its position, it can involve part or the entire gene, or several genes [16,72]. Retrotransposition consists in the reverse transcription of a messenger RNA (mRNA) to complementary DNA (cDNA) which is then inserted in the genome [16,72]. WGD may occur through the lack of disjunction between the daughter chromosomes after mitosis or meiosis [16,72]. This type of duplication was discussed by Susumu Ohno in the book "*Evolution by Gene Duplication*" in 1970 [73]. Ohno, considered WGD to have a more important role in the evolution of organisms than individual gene duplications [16]. He proposed that two rounds of WGS had taken place during vertebrate evolution [16,46,73]. Currently, this theory is known as the 2R hypothesis, and is supported by the comparisons between the number of genes in vertebrates and invertebrates, which reveal to be more in the genomes of vertebrates [16,46,74]. Despite the precise time of these duplications is still unknown, it is estimated that the second event of WGD occurred before the cyclostomes/gnathostome separation [46,75]. After these two rounds of WGD, a third duplication occurred in the Teleostei and a fourth in the Salmonids [46,76,77].

After duplication, the most common fate for the gene copies is nonfunctionalization or pseudogenization (Figure 7a), which consists in the accumulation of degenerative mutations by one of the copies, resulting in the formation of a pseudogene (unexpressed of functionless gene) [72,78]. After some time, the pseudogenes can either be removed from the genome or become so diverged from the parental genes that they are no longer identifiable [72,78]. The functions of the parental genes are assured by the other copies. Alternatively, one of the

copies may undergo mutation giving rise to a new function, which if beneficial is fixed (neofunctionalization) (Figure 7b) [72,78]. This process is one of the most important outcomes of gene duplications since it leads to the generation of gene novelties [78]. Finally, it can also occur subfunctionalization (Figure 7c), in which both copies are maintained after duplication, and accumulate complementary mutations in their coding and/or regulatory regions [78].

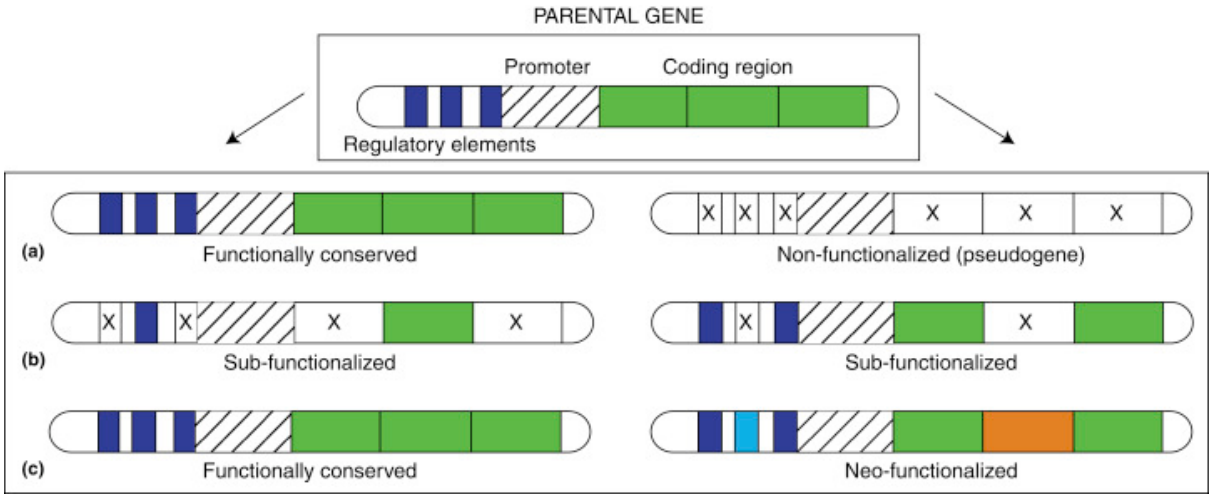


Figure 7. The three potential fates of the duplicated genes. a) One of the copies that accumulates deleterious mutations and becomes a non-functional pseudogene, while the other copy retains the ancestral functions. (b) Both genes acquire deleterious mutations in different regulatory and/or coding regions, thus the ancestral function is partitioned ancestral between the two copies. (c) One of the copies acquires an advantageous mutation (orange) that can lead to evolutionary innovation, while the other paralog retains the ancestral functions [74].

The loss of a gene was only regarded as one of the possible outcomes of gene duplication, thus it was assumed to produce no functional alterations in the organism [79]. However, this vision changed with the increase of the genomic data, because of the emergence of next-generation sequencing technologies [79]. The analysis of various animal genomes revealed that several genes and gene families in different groups were lost during evolution, and that gene loss events could underpin adaptive processes [79]. This was also observed in fungi, plants, protists, and prokaryotes [79,80]. Thus, this evidence contradicted the idea that the number of genes within a genome is associated with the complexity of an organism [79].

Gene loss can be a result of the loss of function of a gene due to the occurrence of a pseudogenization or can involve the gene physical removal through an abrupt mechanism, like unequal crossing over or retrotransposition [79,80]. The loss of a gene is associated with its degree of dispensability, which is influenced by two factors, the mutation robustness, and the environmental variability [79]. The mutational robustness results from the presence of duplicated genes or alternative pathways in regulatory networks, which ensure that if a part of the network fails, the biological functions can be redirected to these alternative pathways

[79,80-83]. Thus, in a system with a high mutational robustness, the effects of mutations are mitigated, which in turn increases the degree of gene dispensability and promotes the occurrence of gene loss [79]. Environmental variability can affect gene dispensability because genes do not have the same degree of importance in all environments, and there are genes that may only be essential in certain environmental conditions [79]. In addition to gene dispensability, the loss of a gene can also be adaptive. There is a growing number of examples of adaptive gene loss in both unicellular and multicellular organisms [79]. In bacteria the loss of genes has been shown to confer adaptive advantages during infections [84-87]. In the case of multicellular organisms, the loss of genes has been associated with pollination in plants [88,89]. The loss of genes has also been extensively associated with eye regression and depigmentation in species that adapted to dark environments like cavefish, mole rats, bats, myriapods and subterranean diving beetles [90-94]; as well as, with adaptations to novel environmental niches such as the case of the colonization of aquatic environments by marine mammals [80,95,96].

Regarding the nuclear receptor superfamily, their diversity is caused by two waves of duplications [46,51,65] (figure 6). The first occurred before the split between the Cnidaria and Bilateria and originated most of the subfamilies and their respective groups [46,55,69]. The second wave corresponded to the WGD that occurred before and after the split of the cyclostomes/gnathostome lineages and led to the formation of the paralogue forms in each subfamily [46,55,69,75]. These events originated receptors that could bind to new ligands. The expansion of the molecules that could serve as ligands allowed an increased complexity in the regulation of physiological processes [97]. Furthermore, some lineage experienced specific gene loss events during their evolution [46,61,98]. Through this process, receptors whose function became less essential were removed from genomes [99]. Therefore, both events were responsible for shaping the repertoire of nuclear receptors in the various Metazoa lineages.

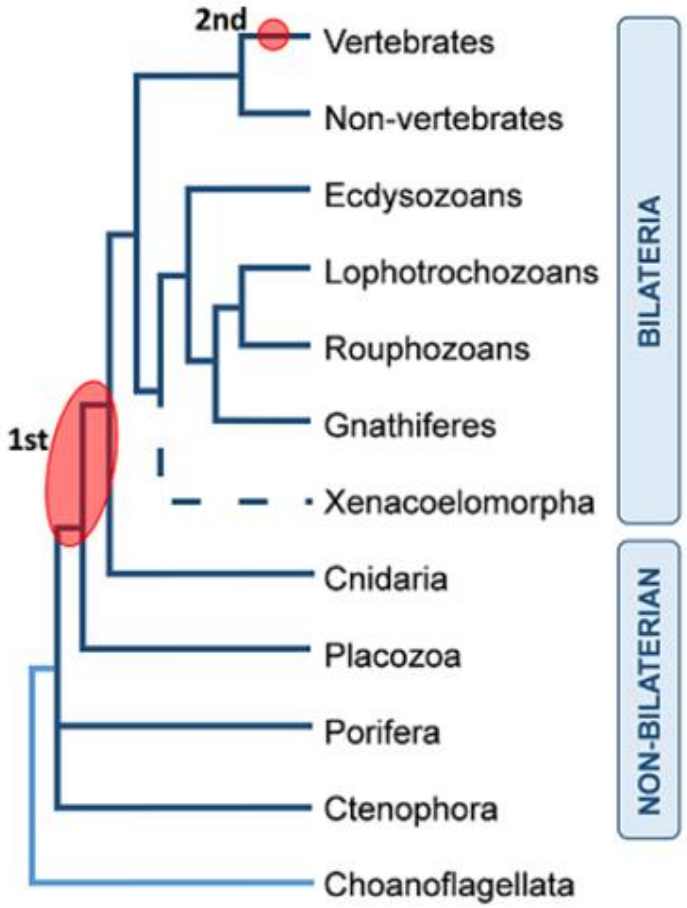


Figure 8. Phylogenetic tree of the Metazoa and Choanoflagellates. The two waves of nuclear receptor gene duplication are represented as circles. Taken from [46].

1.6-Objectives

The continuous technological development of NGS methods has caused a substantial decrease in cost and an increase in the amount of data generated by sequencing [100]. This contributed with advances in the *omics* fields, like genomics, transcriptomics, and proteomics [101,102]. Among these, genomics is the most studied field and involves the analysis of the entirety or parts of genome sequences [103]. An essential step of these analyses is the annotation of the genomic sequences, which consists in the identification of the functional elements present within those sequences [103]. To accomplish this and to give response to the high quantity of biological data that is being continuously generated, several automated annotation pipelines have been developed. However, the process of annotation still presents some major challenges, such as the annotation of fragmented genomes [22]. In addition, the complexity of the eukaryotic genomes also complicates the process of annotation in this group of organisms [22]. In this context, most of the available eukaryotic genomes are not annotated. Moreover, many genomes are released with

no annotation files. Thus, the characterization of gene families such as the nuclear receptor superfamily in metazoan lineages is difficult. Therefore, we aimed to develop a pipeline, the *NRfinder*, to automate these analyses and facilitate the characterization of nuclear receptors in animal genomes. This pipeline will be based on homology searches to identify and classify the nuclear receptors present in a genome without an annotation file. The proposed tool and analysis will allow a better understanding of the nuclear receptor origin and the events responsible for their diversity (gene loss and gene duplication). In addition, a user-friendly web interface will also be developed to make it accessible for users that are not experienced with bioinformatics.

2-Methods

2.1-Brief overview of the *NRfinder*

The *NRfinder* was developed to analyse the repertoire of nuclear receptors of a genome in a faster and automatic way. For this, we can take advantage of the genes coding regions conservation across related species [104]. Thus, we developed a homology-based pipeline that uses the sequences of the well-conserved DBD to infer about the nuclear receptors in a species of interest. For this, two inputs are needed:

1- A file with the highly conserved DBD protein sequences (Fasta format). In order to obtain good results, the user should select DBDs from closely-related species, when available.

2- A file with the genomic sequence (Fasta format), belonging to the species of interest.

Through these two inputs, a two-step process is carried out, consisting of: a) Identification of nuclear receptors present in the target species, by aligning the two inputs with the tBLASTn; b) Classification of the hits found and processed using the BLASTp algorithm against a database composed by nuclear receptors from a reference species. The code used to develop this process is available in Appendix 1.

2.2-Detailed description of each step of the *NRfinder*

1) Detection of nuclear receptors present in the target species

1.1) Alignment between genomic sequence and receptor protein sequences: Each of the protein sequences will be aligned against the target genomic sequence. The output of each alignment will be in a tabular format composed by the columns: Accession (target sequence accession); HSP (target high-scoring segment pair); bit score, identity (percentage of identical matches); e-value (expect value); q. start (start of alignment in the query); q. end (end of alignment in the query); s. start (start of alignment in the target sequence); s. end (end of alignment in the target); frame (target sequence frame), and strand (target strand, forward or reverse). The entries of each table represent the alignment regions between the query sequence and the target. Finally, all the tables are joined to form a single table.

1.2) Removal of overlapping hits: The table entries corresponding to overlapping hits are filtered according to their bit score value. This way, mapped regions correspond to the most significant alignment.

1.3) Concatenation of mapped regions: The mapped regions within a predetermined genomic distance (parameter) are concatenated, and the resulting sequences are translated to protein sequences.

2) Alignment classification

2.1) Reference Database: To perform the classification we created a reference database with sequences of nuclear receptors from previously studied species. These sequences were extracted from the **Refseq** database of the National Center for Biotechnology Information (NCBI) [105]. Before retrieving the sequences, the conserved domains search tool was used to analyse their conserved regions.

2.2) Alignment against reference nuclear receptors: The sequences obtained in the previous step are aligned against a database composed by protein sequences of nuclear receptors from different species. For each sequence in the input file, the subfamily of the receptor whose alignment had the highest bit score value is stored.

2.3) Output generation: After the alignment between the sequences in the input file and the database sequences a table is generated. This table is composed by the columns that correspond to the mapped regions obtained in the first step; and the nuclear receptor subfamily and its bit score value obtained in the second step.

2.5-Evaluation

To assess the performance of *NRfinder*, this pipeline was used to analyse the nuclear receptors present in the genomes of well-annotated species. In each analysis, the accuracy and recall of *NRfinder* were calculated. The recall measure consists in the ratio between the number of nuclear receptors found and the number of annotated receptors (1). The precision measure is the ratio between the number of well classified receivers and the number of receivers found (2).

(1) $$Recall = \frac{Number\ of\ nuclear\ receptors\ found}{Number\ of\ annotated\ nuclear\ receptors} \times 100$$

$$(2)\ Precision = \frac{Number\ of\ nuclear\ receptors\ correctly\ classified}{Number\ of\ annotated\ nuclear\ found} \times 100$$

2.6-Benchmarking

The *NRfinder* results were compared with those obtained by the *Augustus* and *Exonerate* programs. In *Augustus*, the *prints2prfl.pl* script was used to obtain the block profiles from the multiple alignments of nuclear receptor protein sequences. Next, the *fastblocksearch.pl* script was used with the obtained profiles and with the genomic sequence to identify the putative exonic regions of the nuclear receptor's genes. After this, the identified regions, and the block profiles, were used to make the structural predictions. As for the *Exonerate*, before using it, a blast was made between the genome and the protein sequences of the nuclear receptors, to obtain the genomic sub sequences that contain all the hits. This step served to reduce the size of the search space. Next, the DBD protein sequences were aligned with *Exonerate*, using the protein2genome model, against the genomic regions obtained in the previous step. The protein sequences, obtained from both programs, were then aligned with a reference database with nuclear receptors from multiple species like previously. For each protein sequence the nuclear receptor with the maximum bit score value was selected.

3-Results

3.1-Web Service

We developed a web interface for the *NRfidner*, to make it accessible to a wider range of users that are not experienced with bioinformatics and to avoid them to work with the command-line interface. This component consists of a main web page that receives the required inputs and contains the customizable parameters for the analysis, and the results page that is dynamically generated with the results of the analysis. The front-end system of the *NRfinder* was implemented in HyperText Markup Language (HTML) with Cascading Style Sheets (CSS), whereas the back end was developed in Python 3, using the Biopython modules [106] and the BLAST+ package [107] (Appendix 1). The connection between these two components is done with the Hypertext Preprocessor (PHP). Currently, the *NRfinde*r is not yet online, however it is fully functional within the machine used to develop it. In the near future, it will be located in a cluster server of the interdisciplinary Centre of Marine and Environmental Research (CIIMAR) and be freely available to the users. Therefore, the *NRfinder* constitutes a bioinformatic web service that uses simple inputs and a set of established parameters to perform the identification and classification of nuclear receptors present in the genomic sequence (Figure 9).

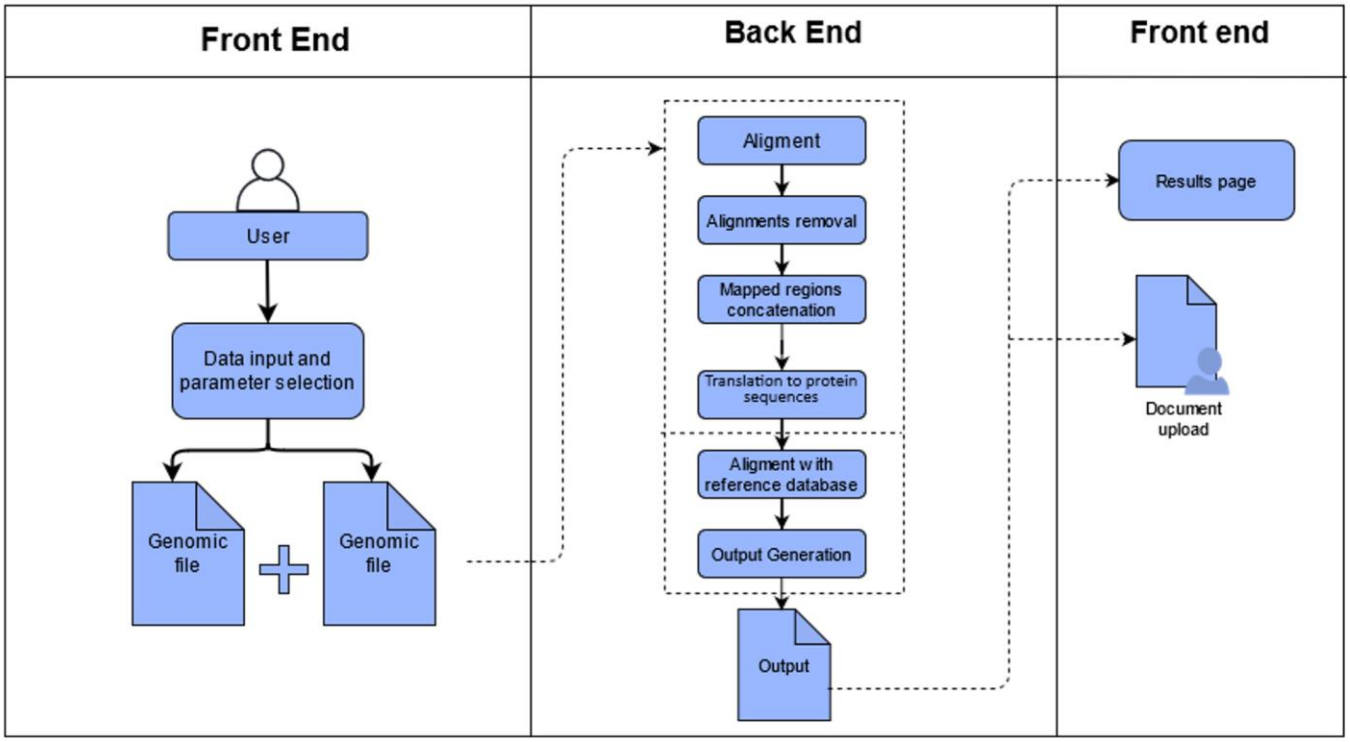


Figure 9. Schematic representation of the *NRfinder*. From left to right and top to bottom: The user gives the required inputs and selects the parameters underlying at the *NRfinder's* homepage to start the analysis. A connection is established with

the PHP which receives the input files and the parameters information and runs the back-end component. This component performs a two-step analysis which consists in 1) the identification of the nuclear receptors in the genomic sequence; 2) classification of the receptors found. The results are displayed in a webpage, and a file with these results is uploaded to the user.

The organization of the *NRfinder* main webpage is represented in figure 10. This page is divided into two sections:

- At the top of the page is the section where the user enters the biological data necessary to carry out the analysis. The DBD sequences of the reference species can be inserted directly in the grey area or by the Ref file button, which receives the file with these sequences. The button Genomic File receives the genomic sequence of the target species. Finally, the Genomic Distance button enables the user to change the distance in which two aligned regions are joined.

- In the middle of the page there is the parameters selection section. In this section the user can change the matrix, e-value, word-size, and gap penalties (open and extension) parameters relative to the alignment between the submitted data. There is also the Min Bit Score parameter, which serves as a filter to remove the alignments with a bit score below this value in the second step of the pipeline.

Figure 10.Overview of the *NRfinder* homepage. There are two sections in this page: a data input section (top of the page); and a parameter selection section (middle of the page).

To illustrate an example of the obtained output we analysed the nuclear receptors of the *Danio rerio*, using the DBDs of the Human nuclear receptors and the default parameters (**evalue**:0.05; **Word-size**:6; **Gap-cost-existence**: 11; **Gap-cost-expansion**: 1; **Matrix**:

Blosum62; **Min BitScore**:40). The first nineteenth rows of the output are illustrated in figure 10. This output consists in a table with the columns Score (maximum bit score of the aligned nuclear receptor); Accession (code correspondent to the genomic region of the hit); Proteins (protein code of the nuclear receptor in the reference database); NR (column with the nuclear receptor subfamily); Start (starting region of the alignment of the obtained hit); and Stop (end region of alignment of the obtained hit). At the top of the page is a button so that the user can download the output, along with the sequences of the obtained hits.

Download

	Score	Accession	Proteins	NR	Start	Stop
0	77.0	NC_007112.7	XP_011530277.1	NR3C2	37000369	37000470
2	146.0	NC_007113.7	NP_005225.2	NR2F6	24376497	24376776
3	144.0	NC_007113.7	NP_001351952.1	NR1F2	49431593	49432537
1	142.0	NC_007113.7	NP_002948.1	NR2B1	20765659	20765844
4	143.0	NC_007114.7	NP_068370.1	NR1D1	23492724	23493029
6	110.0	NC_007114.7	NP_003241.2	NR1A1	34713973	34714276
5	106.0	NC_007114.7	XP_024304880.1	NR1B3	33075038	33075196
7	82.8	NC_007114.7	NP_995582.1	NR5A2	53464201	53464474
9	133.0	NC_007115.7	NP_003288.2	NR2C1	25851152	25851337
8	94.7	NC_007115.7	NP_056953.2	NR1C3	18623618	18623785
12	151.0	NC_007116.7	NP_005645.1	NR2F1	49745706	49745906
13	144.0	NC_007116.7	NP_002948.1	NR2B1	64501616	64501801
10	103.0	NC_007116.7	NP_001351952.1	NR1F2	25376226	25377125
11	72.4	NC_007116.7	NP_000167.1	NR3C1	35607936	35608037
14	157.0	NC_007117.7	NP_775292.1	NR4A3	12463638	12464494
15	89.7	NC_007117.7	NP_001351014.1	NR1I1	38977028	38977371
17	133.0	NC_007118.7	XP_024304052.1	NR1H3	34428873	34429070
16	94.0	NC_007118.7	NP_599022.1	NR1F1	29556575	29558886
18	87.8	NC_007118.7	XP_005260464.1	NR2A1	69032096	69032209

Figure 11.Example of the output of the *NRfinder*.

3.2-*NRfinder* Performance

We tested our pipeline, using the default parameters, in the genomes of vertebrates *Mus musculus*, *D. rerio*, and the invertebrate *Drosophila melanogaster*. In the genomes of the vertebrate species, the human DBD sequences were used as a query. Initially, we tested our

pipeline with a reference database containing the receptors of the Human and later with the receptors of the Human, *Xenopus tropicalis* and *Gallus gallus*. In the case of *D. melanogaster*, the DBD protein sequences of *Daphnia magna* were used as a query. Next, we tested our pipeline with a reference database containing the receptors of D. *magna*, and later we added the receptors of *Bombyx mori* and *Apis mellifera*. In figure 12 are represented the results of the nuclear receptors found by the *NRfinder* and the ones present in the annotations of the tested genomes.

M. musculus

NR	Single Reference	Multiple Reference	Annotated
NR1A	2	2	2
NR1B	4	4	3
NR1C	4	4	3
NR1D	3	3	3
NR1F	4	4	3
NR1H	5	5	4
NR1I	4	4	3
NR2A	2	2	2
NR2B	3	3	3
NR2C	2	2	2
NR2E	3	3	3
NR2F	3	3	3
NR3A	3	3	2
NR3B	8	8	4
NR3C	7	7	4
NR4A	3	3	3
NR5A	4	4	4
NR6A	2	2	1

D. rerio

NR	Single Reference	Multiple Reference	Annotated
NR1A	3	3	3
NR1B	4	4	4
NR1C	7	7	5
NR1D	5	5	5
NR1F	6	6	6
NR1H	3	3	3
NR1I	3	3	3
NR2A	3	3	3
NR2B	6	6	6
NR2C	2	2	2
NR2E	2	2	2
NR2F	6	6	6
NR3A	3	3	3
NR3B	8	8	5
NR3C	5	5	4
NR4A	4	4	4
NR5A	4	4	4
NR6A	2	2	2

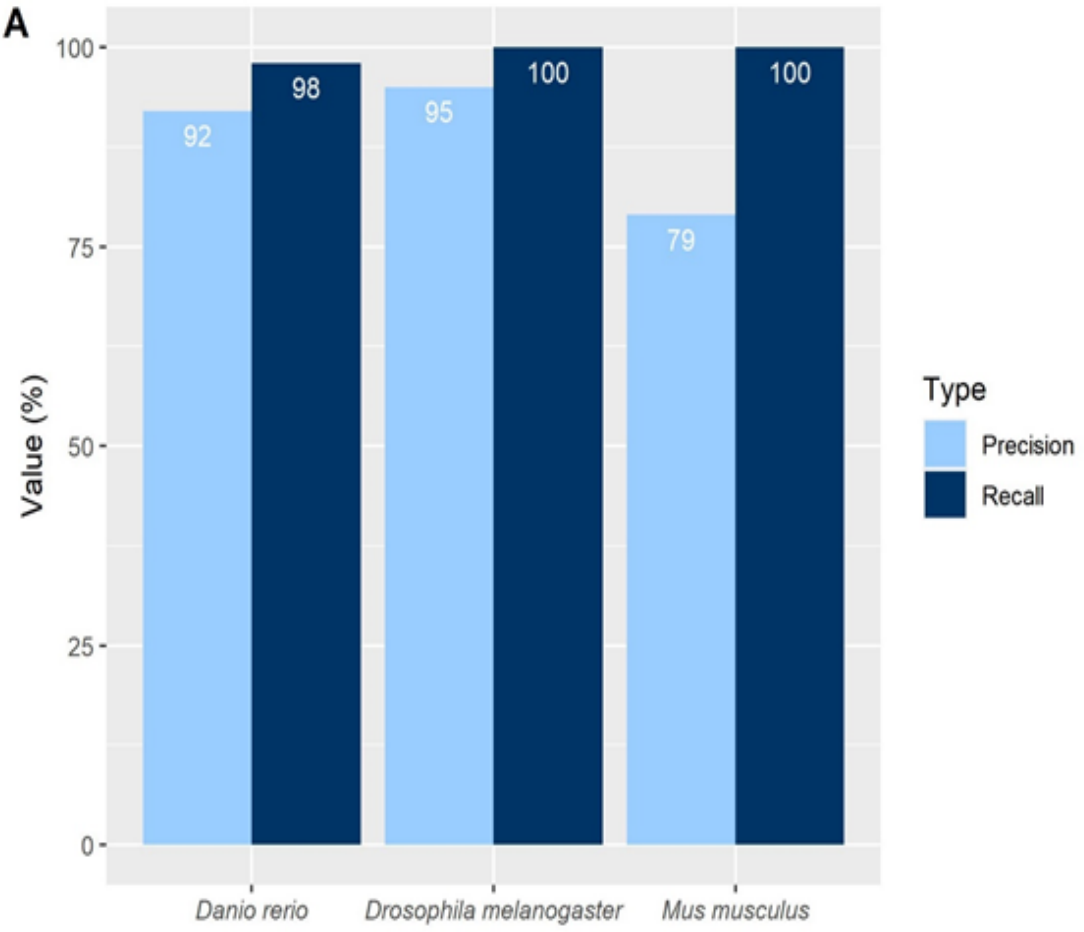
D. melanogaster

NR	Single Reference	Multiple Reference	Annotated
NR0A	3	3	3
NR1D	2	2	1
NR1E	1	1	1
NR1F	1	1	1
NR1H	1	1	1
NR1J	1	1	1
NR2A	1	1	1
NR2B	1	1	1
NR2D	1	1	1
NR2E	3	4	4
NR2F	2	1	1
NR3B	1	1	1
NR4A	1	1	1
NR5A	1	1	1
NR5B	1	1	1
NR6A	1	1	1

Figure 12.Output with the results of the number of receptors of each subfamily in each tested species. Output with the results of the number of receptors of each subfamily in each tested species. Each table possess a column with the subfamilies present in the species (**NR**), the results corresponding to columns to the analysis with the reference database containing nuclear receptors from only one specie (**Single Reference),** the results obtained with the reference database containing multiple species (**Multiple Reference)**, and the number of annotated receptors from each subfamily, respectively.

In a first analysis of the tables in figure 12, we can observe that *NRfinder* identified more receptors than the ones that are annotated. This overestimation results from hits of the same receptor that were classified as different receptors. These cases were redundant errors, and occur due to the genomic distance value, used as inputs. In *M. musculus*, the NR1B, NR1C, NR1F-I, NR3A and NR6A subfamilies had one repetition (**GeneID**: 218772, **GeneID**: 19016, **GeneID**: 19883, **GeneID**: 20186, **GeneID**: 18171, **GeneID**: 13982, and **GeneID**:14536, respectively), the NR3B had four repetitions (two of the gene with the **GeneID**: 26381, and two of the gene with the **GeneID**: 26380), and the NR3C subfamily had three repeated receptors (**GeneID**: 110784, **GeneID**: 18667 and **GeneID**: 11835). In *D. rerio*, the NR1C subfamily had two repeated receptors (of the gene with the **GeneID:** 557037), the NR3B subfamily had three repetitions (of the gene corresponding to the **GeneID**: 405890, and two of the gene with the **GeneID**: 407693), and the NR3C subfamily had one repetition (**GeneID**:

562171). Furthermore, one receptor belonging to the NR3B subfamily (**GeneID**: 110438504) was not found. In *D. melanogaster*, the NR1D had one repeated receptor (**GeneID**: 39999), and one receptor from the NR2E subfamily (**GeneID**: 36702) was classified as being part of the NR2F subfamily. The addition of nuclear receptors, from two different species to the reference database had no impact on the results of the vertebrate species. However, in *D. melanogaster*, this addition led to the correct classification of the receptor of the NR2E subfamily. As a result of this, we analysed the overall specificity and sensitivity of *NRfinder*, and its errors in the tested species, using the results obtained with a reference database with nuclear receptors of different species (Figure 13).



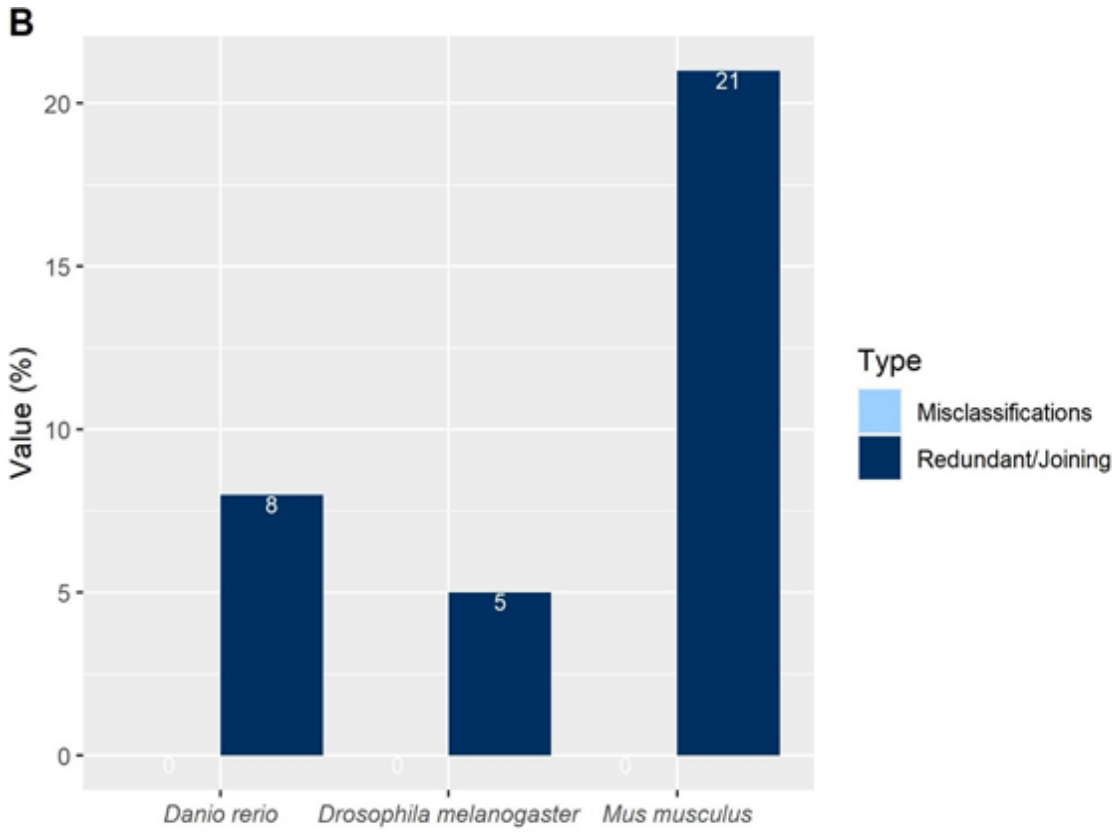


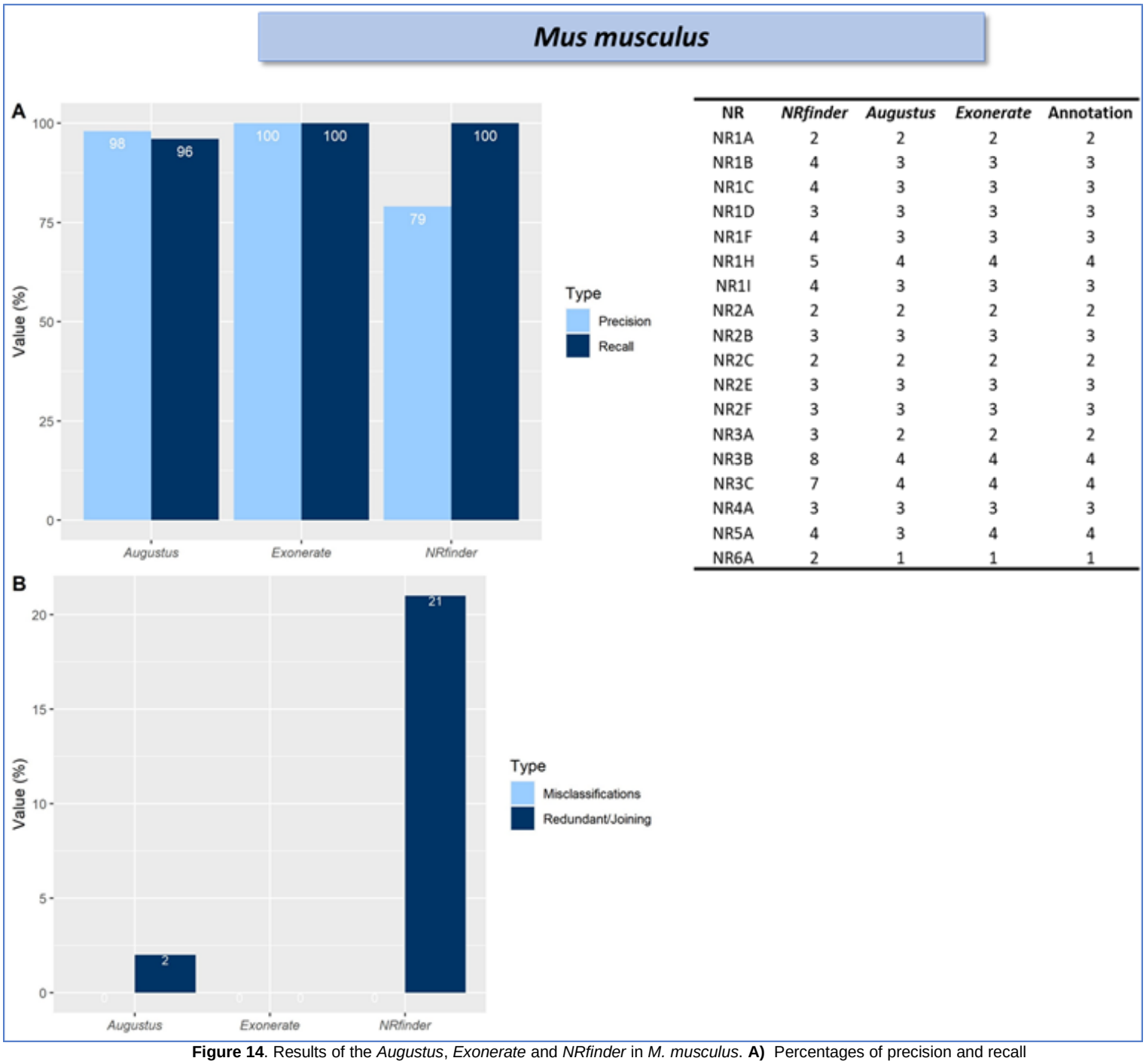
Figure 13.Results of the performance of the NRfinder. A) Percentages of precision and recall in the tested species. B) Percentage of errors in the tested species.

Regarding the results of accuracy and sensitivity (Figure 13A), in *D. rerio* we obtained a recall of 98% and precision of 92%; in *M. musculus* a recall of 100% and precision of 79%; and in *D. melanogaster* a recall of 100% and precision of 95%. The results of the precision can be explained with the errors found. Relatively to these, we found 21% for *M. musculus*, 8% for *D. rerio*, 5% for *D. melanogaster* of redundant/joining errors; no misclassification errors were found (Figure 13B).

3.3-Benchmarking

These results were compared with the ones obtained with the programs *Augustus* and *Exonerate* (Figure 14,15,16). Each figure contains the results regarding the overall accuracy and sensibility, the percentages of errors and the table with the number of receptors of each subfamily found for each program. In the case of *M. musculus*, in *Augustus* one receptor of the NR5A subfamily was repeated (**GeneID**: 26424), and two receptors from the same subfamily were not identified (**GeneID**:100418069, **GeneID**:383927). This program had an accuracy and recall of 98% and 96%, respectively (Figure 14). The repetition of one receptor

corresponded to a 2% of redundant/joining error. In the *Exonerate* all the receptors were correctly identified.



NR	*NRfinder*	*Augustus*	*Exonerate*	Annotation
NR1A	2	2	2	2
NR1B	4	3	3	3
NR1C	4	3	3	3
NR1D	3	3	3	3
NR1F	4	3	3	3
NR1H	5	4	4	4
NR1I	4	3	3	3
NR2A	2	2	2	2
NR2B	3	3	3	3
NR2C	2	2	2	2
NR2E	3	3	3	3
NR2F	3	3	3	3
NR3A	3	2	2	2
NR3B	8	4	4	4
NR3C	7	4	4	4
NR4A	3	3	3	3
NR5A	4	3	4	4
NR6A	2	1	1	1

Figure 14. Results of the *Augustus*, *Exonerate* and *NRfinder* in *M. musculus*. **A)** Percentages of precision and recall of the three programs. **B)** Percentages of errors found in the three programs. In the right-side of the figure there is a table with the number of receptors of each subfamily found in each program, and the number of annotated receptors from each subfamily in *M. musculus*.

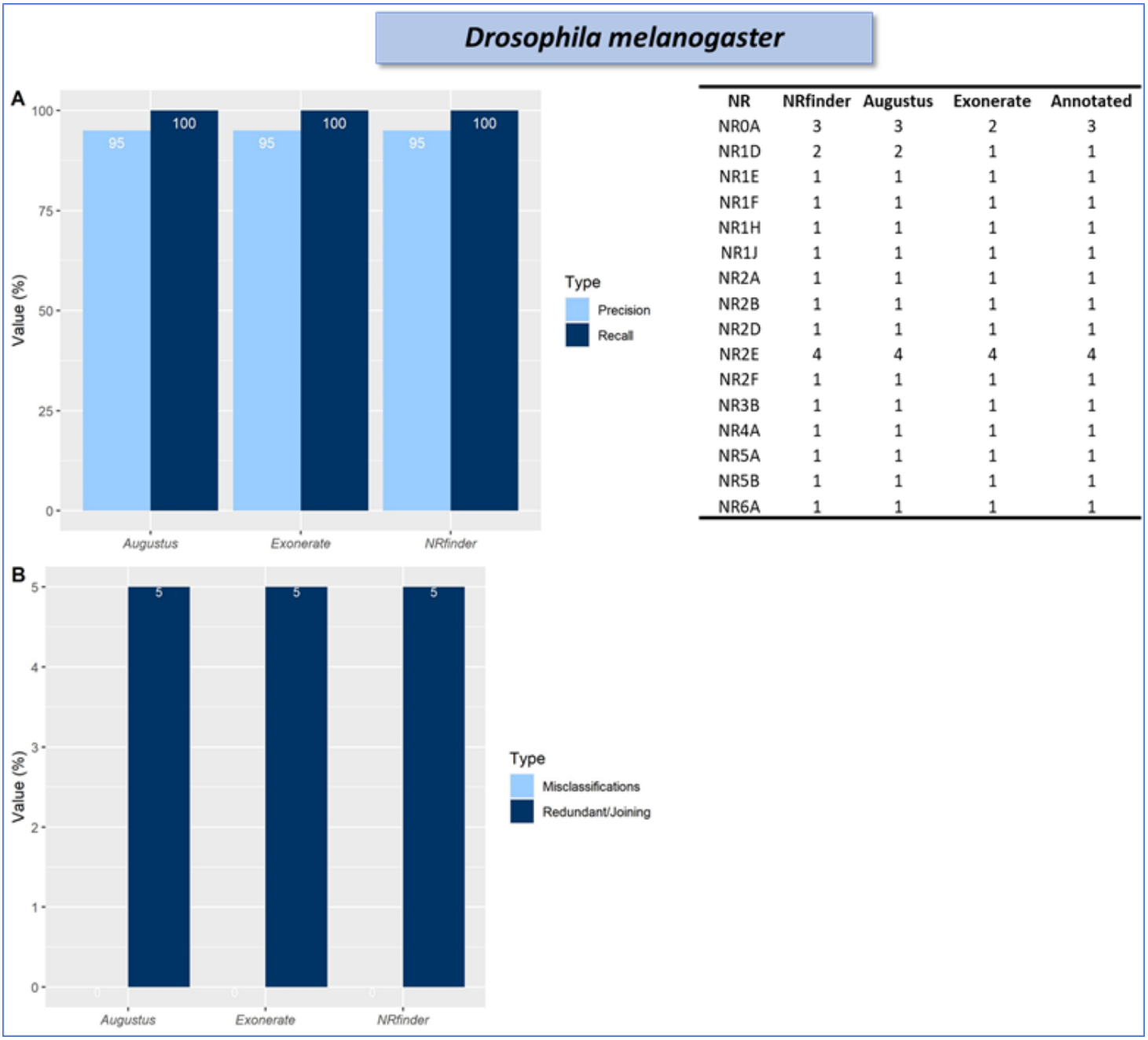
Regarding the *D. rerio* (Figure 15), in *Augustus* we found two repeated receptors that corresponded to the NR1F (**GeneID**: 100004847), NR1C (**GeneID**: 557037), NR2A (**GeneID**: 324010), and, like previously in the *NRfinder* the receptor of the NR3B subfamily (**GeneID**: 110438504) was not found. With these results, this program had an accuracy and recall of

96% and 98%, respectively. The three repetitions corresponded to 4% of redundant/joining errors. In the case of the *Exonerate* no repetitions of receptors were found and like the previous two programs, the receptor of the NR3B subfamily was not identified. Thus, this program had an accuracy and recall of 100% and 98%, respectively.

NR	NRfinder	Augustus	Exonerate	Annotation
NR1A	3	3	3	3
NR1B	4	4	4	4
NR1C	7	6	5	5
NR1D	5	5	5	5
NR1F	6	7	6	6
NR1H	3	3	3	3
NR1I	3	3	3	3
NR2A	3	4	3	3
NR2B	6	6	6	6
NR2C	2	2	2	2
NR2E	2	2	2	2
NR2F	6	6	6	6
NR3A	3	3	3	3
NR3B	8	5	5	5
NR3C	5	4	4	4
NR4A	4	4	4	4
NR5A	4	4	4	4
NR6A	2	2	2	2

Figure 15. Results of the *Augustus*, *Exonerate* and *NRfinder* in *D. rerio*. **A)** Percentages of precision and recall of the three programs. **B)** Percentages of errors found in the three programs. In the right-side of the figure there is a table with the number of receptors of each subfamily found in each program, and the number of annotated receptors from each subfamily in *D. rerio*.

Finally, in *D. melanogaster*, with *Augustus* we obtained one repeated nuclear receptor belonging to the NR1D subfamily (**GeneID**: 39999), which was the same found in *NRfinder* (Figure 16). Therefore, this program had an accuracy and recall of 95% and 100%, respectively, and 5% of redundant/joining error. Relatively to the *Exonerate*, in this program the hit corresponding to the nuclear receptor of the NR0A (**GeneID**: 40287) was joined with one belonging to the same subfamily (**GeneID**: 40285). For this reason, this program had a recall of 100%, a precision of 95%. The fusion of the hits of the two receptors corresponded to a 5% of redundant/joining error.



NR	NRfinder	Augustus	Exonerate	Annotated
NR0A	3	3	2	3
NR1D	2	2	1	1
NR1E	1	1	1	1
NR1F	1	1	1	1
NR1H	1	1	1	1
NR1J	1	1	1	1
NR2A	1	1	1	1
NR2B	1	1	1	1
NR2D	1	1	1	1
NR2E	4	4	4	4
NR2F	1	1	1	1
NR3B	1	1	1	1
NR4A	1	1	1	1
NR5A	1	1	1	1
NR5B	1	1	1	1
NR6A	1	1	1	1

Figure 16. Results of the *Augustus*, *Exonerate* and *NRfinder* in *D. melanogaster*. **A)** Percentages of precision and recall of the three programs. **B)** Percentages of errors found in the three programs. In the right-side of the figure there is a table with the number of receptors of each subfamily found in each program, and the number of annotated receptors from each subfamily in *D. melanogaster*.

4-Discussion

4.1-Performance Analysis

This work aimed to develop an automated pipeline to identify and categorize the nuclear receptors contained in a given animal genome. Regarding the obtained results through the implementation of the *NRfinder*, we did not detect any bias in the identification of receptors with respect to the subfamily to which they belong. In effect, it was possible to identify all the receptors present in the tested genomes, except for the receptor belonging to the NR1D subfamily (GeneID: **39999**) in *D. rerio*. The analysis of its conserved domains showed that it does not possess the DBD, which is the domain used to identify the receptors in the genomes. Regarding redundant/joining errors, these types of errors arise due to the genomic distance used. For example, if two alignments, corresponding to distinct exons of the same gene are at a genomic distance greater than the one defined by the user, then these regions will not be concatenated and will be classified as different receptors (redundant errors). Alternatively, the reverse can also occur, i.e., two alignments that map different genes are within the introduced genomic distance, they will be considered as the same gene (joining errors). In the tested species with our pipeline, there were only found redundant types of errors which resulted in repeated receptors. The highest percentage of these errors was obtained in *M. musculus*.

4.2-Limitations

The errors found are related to the hits obtained from the blast search that do not provide any information about the structure of the gene [108]. In addition, these errors may also be related to the complexity of eukaryotic genomes. Unlike prokaryotic organisms, in the genomes of eukaryotic organisms, gene regions are often long, and their coding regions (exons) are interspaced with non-coding regions (introns), which decreases the precision of gene prediction [19]. Furthermore, since our pipeline was developed using homology-based methods, its performance is associated with the diversity of receptors from different species present in our reference database. Likewise, the input sequences used to carry out the analysis should be from closely related species to the one that is being tested with, to obtain better results. The use of sequences from distantly related species could lead to wrong inferences.

4.3-Benchmarking

The *Augustus* and *Exonerate* programs are used to predict genes in eukaryotic genomic sequences. In this sense, their results were compared with the ones obtained with *NRfinder*. In Augustus two receptors from *M. musculus* belonging to the NR5A subfamily were not identified. These receptors corresponded to pseudogenes that “coded” short proteins. Although it correctly predicted other pseudogenes (GeneID: **545289**, GeneID: **100286842**), Augustus, as well as other ab initio methods are not very precise in predicting short proteins [19]. Regarding the percentage of errors, this program generated 2%, 4% and 5% of redundant errors in *M. musculus*, *D. rerio*, and *D. melanogaster* respectively. These errors can be due to the initial and termination exons not being predicted as accurately as the internal exons in ab initio programs such as *Augustus* [19]. Because of this, when analysing these genes, the inaccurate prediction of these exons led to the classification of this region as distinct genes. However, these results are comparatively better than the ones obtained in *NRfinder*, since this program generated fewer redundant errors, especially in *M. musculus*. The only case where the *NRfinder* is not outperformed *Augustus* was in the *D. melanogaster*, in which they had the same performance.

Regarding the comparison with *Exonerate*, the initial analysis of the results led to the assumption that this program did not identify one nuclear receptor in *Drosophila melanogaster*. However, after examining the region of missing receptor, it was observed that this region was found by the program but was predicted as being part of the closest receptor. Therefore, this case was considered to be a joining error. With the exception of the *Drosophila melanogaster,* these two programs outperformed the *NRfinder* due to the less or non-occurrence of redundant/joining errors in the other tested species. This is because they are more advanced than the BLAST method used in the first part of our pipeline. *Augustus* uses pre-compiled parameters and homologous sequences to generate its models and make the gene prediction [39,42]. On the other hand, *Exonerate* is an alignment-based algorithm that uses splice site models that enables it to have better predictions of the exonic regions than BLAST [20]. In addition, between *Augustus* and *Exonerate*, the last one performed better to the non-occurrence of redundant errors. Despite being slightly outperformed, our pipeline represents a first attempt, to our knowledge, of an automatic process of both identification and classification of nuclear receptors. Since most genomes are not annotated, and some have erroneous annotations, we created a gene-family specific pipeline to analyse genomic sequences without the use of the underlying gene annotations. This is done by using the high degree of conservation of the DBD of the nuclear receptors to analyse the repertoire of nuclear receptors in a given genome. After identifying the nuclear receptors' possible coding regions,

we used a reference database with sequences of selected nuclear receptors from different species to classify the identified hits. Thus, through this process, the *NRfinder* was able, with a simple input, to automatically identify and classify the majority of the nuclear receptors in the tested genomes in a rapid manner, without the use of annotations.

5-Conclusion and Future Perspectives

The objective of this work was to develop a pipeline for characterization of the nuclear receptors present in a genomic sequence. To achieve this, homology techniques that use the DBD sequences, from closely-related species, have been implemented to identify the receptors in the target species. Furthermore, this pipeline was integrated into a web system to facilitate its use and make it publicly available to non-experienced users interested in studying the composition of the nuclear receptor family in a species.

The results obtained from the model species demonstrate that the *NRfinder* correctly identified and classified most of the nuclear receptors present in the genomes. Even so, the comparison with other gene prediction programs evidenced that they were slightly better. Thus, the integration of more advanced gene prediction programs can improve the performance of our pipeline, thus giving more accurate results to the users. Another aspect that will also improve our pipeline is the development of a specific database for nuclear receptors. Through these the *NRfinder* will be able to facilitate the study of nuclear receptors in neglected animal groups.

6-References

1- Van Dijk, E. L., Auger, H., Jaszczyszyn, Y., & Thermes, C. (2014). Ten years of next-generation sequencing technology. Trends in genetics, 30(9), 418-426.

2- Sanger, F., Nicklen, S., & Coulson, A. R. (1977). DNA sequencing with chain-terminating inhibitors. Proceedings of the national academy of sciences, 74(12), 5463-5467.

3- Maxam, A. M., & Gilbert, W. (1977). A new method for sequencing DNA. Proceedings of the National Academy of Sciences, 74(2), 560-564.

4- Heather, J. M., & Chain, B. (2016). The sequence of sequencers: The history of sequencing DNA. Genomics, 107(1), 1-8.

5- Sequencing, H. G. (2004). Finishing the euchromatic sequence of the human genome. Nature, 431(7011), 931-45.

6- Mardis, E. R. (2013). Next-generation sequencing platforms. Annual review of analytical chemistry, 6, 287-303.

7- Grada, A., & Weinbrecht, K. (2013). Next-generation sequencing: methodology and application. The Journal of investigative dermatology, 133(8), e11.

8- Bleidorn, C. (2017). Phylogenomics. *Cham: Springer International Publishing*.

9- Ari, Ş., & Arikan, M. (2016). Next-generation sequencing: advantages, disadvantages, and future. In Plant omics: Trends and applications (pp. 109-135). Springer, Cham.

10- Choucair, F. (2018). Unraveling the sperm transcriptome by next generation sequencing and the global epigenetic landscape in infertile men.

11- Del Angel, V. D., Hjerde, E., Sterck, L., Capella-Gutierrez, S., Notredame, C., Pettersson, O. V., ... & Lantz, H. (2018). Ten steps to get started in Genome Assembly and Annotation. F1000Research, 7.

12- Rhoads, A., & Au, K. F. (2015). PacBio sequencing and its applications. Genomics, proteomics & bioinformatics, 13(5), 278-289.

13- Korlach, J., Bjornson, K. P., Chaudhuri, B. P., Cicero, R. L., Flusberg, B. A., Gray, J. J., ... & Turner, S. W. (2010). Real-time DNA sequencing from single polymerase molecules. Methods in enzymology, 472, 431-455.

14- Vilgis, S., & Deigner, H. P. (2018). Sequencing in precision medicine. In Precision medicine (pp. 79-101). Academic Press.

15- Foxman, B. (2010). Molecular tools and infectious disease epidemiology. Academic Press,53-78.

16- Choudhuri, S. (2014). Bioinformatics for beginners: genes, genomes, molecular evolution, databases and analytical tools. Elsevier.

17- Baker, M. (2012). De novo genome assembly: what every biologist should know. *Nature methods*, *9*(4), 333-337.

18- Harrow, J., Nagy, A., Reymond, A., Alioto, T., Patthy, L., Antonarakis, S. E., & Guigó, R. (2009). Identifying protein-coding genes in genomic sequences. Genome Biology, 10(1), 1-8.

19- Scalzitti, N., Jeannin-Girardon, A., Collet, P., Poch, O., & Thompson, J. D. (2020). A benchmark study of ab initio gene prediction methods in diverse eukaryotic organisms. BMC genomics, 21(1), 1-20.

20- Yandell, M., & Ence, D. (2012). A beginner's guide to eukaryotic genome annotation. Nature Reviews Genetics, 13(5), 329-342.

21- Sleator, R. D. (2010). An overview of the current status of eukaryote gene prediction strategies. Gene, 461(1-2), 1-4.

22- Salzberg, S. L. (2019). Next-generation genome annotation: we still struggle to get it right.

23- Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. Journal of molecular biology, 215(3), 403-410.

24- Kent, W. J. (2002). BLAT—the BLAST-like alignment tool. Genome research, 12(4), 656-664.

25- Stanke, M., & Waack, S. (2003). Gene prediction with a hidden Markov model and a new intron submodel. Bioinformatics, 19(suppl_2), ii215-ii225.

26- Stanke, M., Schöffmann, O., Morgenstern, B., & Waack, S. (2006). Gene prediction in eukaryotes with a generalized hidden Markov model that uses hints from external sources. *BMC bioinformatics*, *7*(1), 1-11.

27- Ter-Hovhannisyan, V., Lomsadze, A., Chernoff, Y. O., & Borodovsky, M. (2008). Gene prediction in novel fungal genomes using an ab initio algorithm with unsupervised training. Genome research, 18(12), 1979-1990.

28- Zhu, W., Lomsadze, A., & Borodovsky, M. (2010). Ab initio gene identification in metagenomic sequences. Nucleic acids research, 38(12), e132-e132.

29- Cantarel, B. L., Korf, I., Robb, S. M., Parra, G., Ross, E., Moore, B., ... & Yandell, M. (2008). MAKER: an easy-to-use annotation pipeline designed for emerging model organism genomes. Genome research, 18(1), 188-196.

30- Holt, C., & Yandell, M. (2011). MAKER2: an annotation pipeline and genome-database management tool for second-generation genome projects. BMC bioinformatics, 12(1), 1-14.

31- Hoff, K. J., Lange, S., Lomsadze, A., Borodovsky, M., & Stanke, M. (2016). BRAKER1: unsupervised RNA-Seq-based genome annotation with GeneMark-ET and AUGUSTUS. Bioinformatics, 32(5), 767-769.

32- Brůna, T., Hoff, K. J., Lomsadze, A., Stanke, M., & Borodovsky, M. (2021). BRAKER2: Automatic eukaryotic genome annotation with GeneMark-EP+ and AUGUSTUS supported by a protein database. NAR genomics and bioinformatics, 3(1), lqaa108.

33- Xiong, J. (2006). Essential bioinformatics. Cambridge University Press.

34- Rocha, M., & Ferreira, P. G. (2018). Bioinformatics Algorithms: Design and Implementation in Python. Academic Press.

35- Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of molecular biology, 48(3), 443-453.

36- Smith, T. F., & Waterman, M. S. (1981). Identification of common molecular subsequences. Journal of molecular biology, 147(1), 195-197.

37- Pertsemlidis, A., & Fondon, J. W. (2001). Having a BLAST with bioinformatics (and avoiding BLASTphemy). Genome biology, 2(10), 1-10.

38- Pearson, W. R. (2013). An introduction to sequence similarity (“homology”) searching. Current protocols in bioinformatics, 42(1), 3-1.

39- Slater, G. S. C., & Birney, E. (2005). Automated generation of heuristics for biological sequence comparison. *BMC bioinformatics*, *6*(1), 1-11.

40- Yoon, B. J. (2009). Hidden Markov models and their applications in biological sequence analysis. *Current genomics*, *10*(6), 402-415.

41- Stanke, M., Tzvetkova, A., & Morgenstern, B. (2006). AUGUSTUS at EGASP: using EST, protein and genomic alignments for improved gene prediction in the human genome. *Genome biology*, *7*(1), 1-8.

42- Keller, O., Kollmar, M., Stanke, M., & Waack, S. (2011). A novel hybrid gene prediction method employing protein multiple sequence alignments. *Bioinformatics*, *27*(6), 757-763.

43- Ruggiero, M. A., Gordon, D. P., Orrell, T. M., Bailly, N., Bourgoin, T., Brusca, R. C., ... & Kirk, P. M. (2015). A higher-level classification of all living organisms. *PloS one*, *10*(4), e0119248.

44- Holland P. 2011. *The Animal Kingdom: A Very Short Introduction*. 1st ed. Oxford University Press: New York.

45- Nielsen C. 2012. *Animal Evolution Interrelationships of the Living Phyla*. 3rd ed. Oxford University Press: New York.

46- Da Fonseca, E. S. D. S. (2020). Nuclear receptors in metazoan lineages: The cross-talk between evolution and endocrine disruption. Dissertação conferente ao grau de doutor pela faculdade de ciências da universidade do Porto, ano 2020, Porto Portugal

47- King, N., Westbrook, M. J., Young, S. L., Kuo, A., Abedin, M., Chapman, J., ... & Rokhsar, D. (2008). The genome of the choanoflagellate Monosiga brevicollis and the origin of metazoans. *Nature*, *451*(7180), 783-788.

48- Lanna, E. (2015). Evo-devo of non-bilaterian animals. *Genetics and molecular biology*, *38*, 284-300.

49- Osigus, H. J., Rolfes, S., Herzog, R., Kamm, K., & Schierwater, B. (2019). Polyplacotoma mediterranea is a new ramified placozoan species. *Current Biology*, *29*(5), R148-R149.

50- Eitel, M., Francis, W. R., Varoqueaux, F., Daraspe, J., Osigus, H. J., Krebs, S., Vargas, S., Blum, H., Williams, G. A., Schierwater, B., & Wörheide, G. (2018). Comparative genomics and the nature of placozoan species. *PLoS biology*, *16*(7), e2005359.

51- Crustacea, S. (2016). Introduction to the Bilateria and the Phylum Xenacoelomorpha.

52- Cannon, J. T., Vellutini, B. C., Smith, J., Ronquist, F., Jondelius, U., & Hejnol, A. (2016). Xenacoelomorpha is the sister group to Nephrozoa. *Nature*, *530*(7588), 89-93.

53- Germain, P., Staels, B., Dacquet, C., Spedding, M., & Laudet, V. (2006). Overview of nomenclature of nuclear receptors. Pharmacological reviews, 58(4), 685-704.

54- Laudet, V., & Gronemeyer, H. (2002). *The nuclear receptor factsbook* (No. 964). Gulf Professional Publishing.

55- Escriva, H., Delaunay, F., & Laudet, V. (2000). Ligand binding and nuclear receptor evolution. *Bioessays*, *22*(8), 717-727.

56- Giguère, V., Hollenberg, S. M., Rosenfeld, M. G., & Evans, R. M. (1986). Functional domains of the human glucocorticoid receptor. *Cell*, *46*(5), 645-652.

57- Greene, G. L., Gilna, P., Waterfield, M., Baker, A., Hort, Y., & Shine, J. (1986). Sequence and expression of human estrogen receptor complementary DNA. *Science*, *231*(4742), 1150-1154.

58- Gronemeyer, H. (1995). Transcription factors 3: nuclear receptors. Protein profile, 2.

59- Beinsteiner, B., Markov, G. V., Erb, S., Chebaro, Y., McEwen, A. G., Cianférani, S., ... & Billas, I. M. (2021). A structural signature motif enlightens the origin and diversification of nuclear receptors. PLoS genetics, 17(4), e1009492.

60- Gronemeyer, H., Gustafsson, J. Å., & Laudet, V. (2004). Principles for modulation of the nuclear receptor superfamily. Nature reviews Drug discovery, 3(11), 950-964.

61- Bertrand, S., Belgacem, M. R., & Escriva, H. (2011). Nuclear hormone receptors in chordates. Molecular and cellular endocrinology, 334(1-2), 67-75.

62- Krust, A., Green, S., Argos, P., Kumar, V., Walter, P., Bornert, J. M., & Chambon, P. (1986). The chicken oestrogen receptor sequence: homology with v-erbA and the human oestrogen and glucocorticoid receptors. *The EMBO journal*, *5*(5), 891-897.

63-Holzer, G., Markov, G. V., & Laudet, V. (2017). Evolution of Nuclear Receptors and Ligand Signaling: Toward a Soft Key–Lock Model?. Current topics in developmental biology, 125, 1-38.

64- Kumar, V., Green, S., Staub, A., & Chambon, P. (1986). Localisation of the oestradiol-binding and putative DNA-binding domains of the human oestrogen receptor. *The EMBO journal*, *5*(9), 2231-2236.

65- Heldin, C. H., Lu, B., Evans, R., & Gutkind, J. S. (2016). Signals and receptors. *Cold Spring Harbor Perspectives in Biology*, *8*(4), a005900.

66- Wärnmark, A., Treuter, E., Wright, A. P., & Gustafsson, J. A. (2003). Activation functions 1 and 2 of nuclear receptors: molecular strategies for transcriptional activation. *Molecular endocrinology*, *17*(10), 1901-1909.

67- Sonoda, J., Pei, L., & Evans, R. M. (2008). Nuclear receptors: decoding metabolic disease. *FEBS letters*, *582*(1), 2-9.

68- Markov, G. V., & Laudet, V. (2011). Origin and evolution of the ligand-binding ability of nuclear receptors. *Molecular and cellular endocrinology*, *334*(1-2), 21-30.

69- Escriva, H., Safi, R., Hänni, C., Langlois, M. C., Saumitou-Laprade, P., Stehelin, D., ... & Laudet, V. (1997). Ligand binding was acquired during evolution of nuclear receptors. Proceedings of the National Academy of Sciences, 94(13), 6803-6808.

70- McEwan, I. J., Escriva, H., Bertrand, S., & Laudet, V. (2004). The evolution of the nuclear receptor superfamily. Essays in biochemistry, 40, 11-26

71- Bridgham, J. T., Eick, G. N., Larroux, C., Deshpande, K., Harms, M. J., Gauthier, M. E., ... & Thornton, J. W. (2010). Protein evolution by molecular tinkering: diversification of the nuclear receptor superfamily from a ligand-dependent ancestor. PLoS biology, 8(10), e1000497.

72- Zhang, J. (2003). Evolution by gene duplication: an update. *Trends in ecology & evolution*, *18*(6), 292-298.

73- Ohno, S. Evolution by Gene Duplication (Springer, Berlin, 1970).

74- Putnam, N. H., Butts, T., Ferrier, D. E., Furlong, R. F., Hellsten, U., Kawashima, T., ... & Rokhsar, D. S. (2008). The amphioxus genome and the evolution of the chordate karyotype. Nature, 453(7198), 1064-1071.

75- Nakatani, Y., Shingate, P., Ravi, V., Pillai, N. E., Prasad, A., McLysaght, A., & Venkatesh,B. (2021). Reconstruction of proto-vertebrate, proto-cyclostome and proto-gnathostome genomes provides new insights into early vertebrate evolution. Nature communications, 12(1), 1-14.

76- Kuraku, S., & Meyer, A. (2009). The evolution and maintenance of Hox gen in vertebrates and the teleost-specific genome duplication. International Journal of Developmental Biology, 53(5/6), 765-773.

77- Mehta, T. K., Ravi, V., Yamasaki, S., Lee, A. P., Lian, M. M., Tay, B. H., ... & Venkatesh, B. (2013). Evidence for at least six Hox clusters in the Japanese lamprey (Lethenteron japonicum). Proceedings of the National Academy of Sciences, 110(40), 16044-16049.

78- Moriyama, Y., & Koshiba-Takeuchi, K. (2018). Significance of whole-genome duplications on the emergence of evolutionary novelties. Briefings in functional genomics, 17(5), 329-338.

79- Albalat, R., & Cañestro, C. (2016). Evolution by gene loss. Nature Reviews Genetics, 17(7), 379-391.

80- Espregueira Themudo, G., Alves, L. Q., Machado, A. M., Lopes-Marques, M., da Fonseca, R. R., Fonseca, M., ... & Castro, L. F. C. (2020). Losing genes: the evolutionary remodeling of Cetacea skin. Frontiers in Marine Science, 7, 912.

81- Félix, M. A., & Barkoulas, M. (2015). Pervasive robustness in biological systems. *Nature Reviews Genetics*, *16*(8), 483-496.

82- Piškur, J., Sandrini, M. P., Knecht, W., & Munch-Petersen, B. (2004). Animal deoxyribonucleoside kinases: 'forward'and 'retrograde'evolution of their substrate specificity 1. *FEBS letters*, *560*(1-3), 3-6.

83- Wagner, A. (2005). Distributed robustness versus redundancy as causes of mutational robustness. *Bioessays*, *27*(2), 176-188.

84- Jain, N., Li, L., Hsueh, Y. P., Guerrero, A., Heitman, J., Goldman, D. L., & Fries, B. C. (2009). Loss of allergen 1 confers a hypervirulent phenotype that resembles mucoid switch variants of Cryptococcus neoformans. *Infection and immunity*, *77*(1), 128-140.

85- Maurelli, A. T., Fernández, R. E., Bloch, C. A., Rode, C. K., & Fasano, A. (1998). "Black holes" and bacterial pathogenicity: a large genomic deletion that enhances the virulence of

Shigella spp. and enter invasive Escherichia coli. *Proceedings of the National Academy of Sciences*, *95*(7), 3943-3948.

86- Moore, R. A., Reckseidler-Zenteno, S., Kim, H., Nierman, W., Yu, Y., Tuanyok, A., ... & Woods, D. E. (2004). Contribution of gene loss to the pathogenic evolution of Burkholderia pseudomallei and Burkholderia mallei. *Infection and immunity*, *72*(7), 4172-4187

87- Yu, H., Hanes, M., Chrisp, C. E., Boucher, J. C., & Deretic, V. (1998). Microbial pathogenesis in cystic fibrosis: pulmonary clearance of mucoid Pseudomonas aeruginosa and inflammation in a mouse model of repeated respiratory challenge. *Infection and immunity*, *66*(1), 280-288.

88- Hoballah, M. E., Gübitz, T., Stuurman, J., Broger, L., Barone, M., Mandel, T., ... & Kuhlemeier, C. (2007). Single gene–mediated shift in pollinator attraction in Petunia. *The Plant Cell*, *19*(3), 779-790.

89- Zufall, R. A., & Rausher, M. D. (2004). Genetic changes associated with floral adaptation restrict future evolutionary potential. *Nature*, *428*(6985), 847-850.

90- Leys, R., Cooper, S. J., Strecker, U., & Wilkens, H. (2005). Regressive evolution of an eye pigment gene in independently evolved eyeless subterranean diving beetles. Biology letters, 1(4), 496-499.

91- Chipman, A. D., Ferrier, D. E., Brena, C., Qu, J., Hughes, D. S., Schröder, R., ... & Richards, S. (2014). The first myriapod genome sequence reveals conservative arthropod gene content and genome organisation in the centipede Strigamia maritima. PLoS biology, 12(11), e1002005.

92- Zhao, H., Rossiter, S. J., Teeling, E. C., Li, C., Cotton, J. A., & Zhang, S. (2009). The evolution of color vision in nocturnal mammals. *Proceedings of the National Academy of Sciences*, *106*(22), 8980-8985.

93- Kim, E. B., Fang, X., Fushan, A. A., Huang, Z., Lobanov, A. V., Han, L., ... & Gladyshev, V. N. (2011). Genome sequencing reveals insights into physiology and longevity of the naked mole rat. Nature, 479(7372), 223-227.

94- Emerling, C. A., & Springer, M. S. (2014). Eyes underground: regression of visual protein networks in subterranean mammals. Molecular Phylogenetics and Evolution, 78, 260-270.

95- Lopes-Marques, M., Machado, A. M., Barbosa, S., Fonseca, M. M., Ruivo, R., & Castro, L. F. C. (2018). Cetacea are natural knockouts for IL20. *Immunogenetics*, *70*(10), 681-687.

96- Lopes-Marques, M., Alves, L. Q., Fonseca, M. M., Secci-Petretto, G., Machado, A. M., Ruivo, R., & Castro, L. F. C. (2019). Convergent inactivation of the skin-specific CC motif chemokine ligand 27 in mammalian evolution. *Immunogenetics*, *71*(5), 363-372.

97- Baker, M. E. (2019). Steroid receptors and vertebrate evolution. *Molecular and cellular endocrinology*, *496*, 110526.

98- Taubenheim, J., Kortmann, C., & Fraune, S. (2021). Function and Evolution of Nuclear Receptors in Environmental-Dependent Postembryonic Development. *Frontiers in Cell and Developmental Biology*, *9*.

99- Bertrand, S., Brunet, F. G., Escriva, H., Parmentier, G., Laudet, V., & Robinson-Rechavi, M. (2004). Evolutionary genomics of nuclear receptors: from twenty-five ancestral genes to derived endocrine systems. *Molecular biology and evolution*, *21*(10), 1923-1937.

100- Sboner, A., Mu, X. J., Greenbaum, D., Auerbach, R. K., & Gerstein, M. B. (2011). The real cost of sequencing: higher than you think!. Genome biology, 12(8), 1-10.

101- Hasin, Y., Seldin, M., & Lusis, A. (2017). Multi-omics approaches to disease. Genome biology, 18(1), 1-15

102- Ohashi, H., Hasegawa, M., Wakimoto, K., & Miyamoto-Sato, E. (2015). Next-generation technologies for multiomics approaches including interactome sequencing. *BioMed research international*, *2015*.

103- Abril, J. F., & Castellano Hereza, S. (2019). Genome annotation. Elsevier.

104- Sharma, V., Elghafari, A., & Hiller, M. (2016). Coding exon-structure aware realigner (CESAR) utilizes genome alignments for accurate comparative gene annotation. Nucleic acids research, 44(11), e103-e103.

105-O'Leary, N. A., Wright, M. W., Brister, J. R., Ciufo, S., Haddad, D., McVeigh, R., ... & Pruitt, K. D. (2016). Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. Nucleic acids research, 44(D1), D733-D745.

106- Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., ... & De Hoon, M. J. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, *25*(11), 1422-1423.

107- Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., & Madden, T. L. (2009). BLAST+: architecture and applications. *BMC bioinformatics*, *10*(1), 1-9.

108- She, R., Chu, J. S. C., Uyar, B., Wang, J., Wang, K., & Chen, N. (2011). genBlastG: using BLAST searches to build homologous gene models. *Bioinformatics*, *27*(15), 2141-2143.

Appendix 1

In the box below, I present the code used to develop the *NRfinder* back-end in Python 3. This code consists in functions, which have comment briefly, with a short description. The **Identification()** and **Classification ()** are the functions responsible for the two processes of the back-end component.

```
#Function used to create a database from a fasta file
def create_database(genome_file,db_type):
    from Bio.Blast.Applications import NcbimakeblastdbCommandline
    command_line = NcbimakeblastdbCommandline(dbtype=db_type,input_file=genome_file)
    command_line()
#Creates a dictionary from  a fasta file
def get_seqs(file):
    from Bio import SeqIO
    dic = {}
    for record in SeqIO.parse(file, "fasta"):
        dic[record.id] = str(record.seq)
    return dic
#Creates a fasta file with one sequence from the input
def create_tempfile(header,sequence,temp_file='Tem_seq_file.fasta'):
    with open(temp_file,'w') as txt:
        txt.write('>'+header+'\n'+sequence+'\n')
#Carries out the tBlastn algorithm
def perform_tblastn(seq_file,output,db,parameters):
    from Bio.Blast.Applications import NcbitblastnCommandline
    command_line = NcbitblastnCommandline(db =
db,query=seq_file,out=output,evalue=float(parameters[0]),word_size=int(parameters[1]),outfmt =
'6 std qcovs qcovhsp sframe',gapopen=int(parameters[2]),max_target_seqs = 5000,gapextend =
int(parameters[3]),window_size = 40,matrix = parameters[4])
    command_line()
```

```
#Formats the dataframe received
def formating_dataframe(temp_file):
    import pandas as pd
    import numpy as np
    df = pd.read_csv(temp_file,sep = '\t', header = None,names=['query acc.ver', 'Accession', 'identity',
        'alignment length', 'mismatches', 'gap opens', 'q. start', 'q. end', 's. start',
        's. end', 'evalue', 'bit score','Query Coverage','HSP','Frame'])
    df = df.loc[:,['Accession','HSP','bit score','identity','evalue','q. start', 'q. end','s. start', 's. end','Frame']]
    condition = df['s. start'] > df['s. end']
    df.loc[condition, ['s. start', 's. end']] = df.loc[condition, ['s. end', 's. start']].values
    df = df.sort_values(by = ['Accession','s. start','bit score'],ascending= False)
    df = df.reset_index(drop = True)
    df['strand'] = np.where(df['Frame'] > 0,'positive','negative')
    return df
#Removes the overlapping hits from the dataframe
def remove_overlapps(df):
    results = []
    for i in zip(df.index , df['s. start'] , df['s. end'], df['bit score'],df['Accession']):
        if i in results:
            continue
        for j in zip(df.index,df['s. start'],df['s. end'],df['bit score'],df['Accession']):
            if i[0] == j[0] or i[0] in results or j[0] in results or i[4]!=j[4]:
                continue
            if i[1] <= j[1] < i[2]:
                if i[3] > j[3]:
                    results.append(j[0])
                else:
                    results.append(i[0])
```

```
    res = df.drop(results)
    res = res.reset_index(drop=True)
    return res
#Processess the hits in the dataframe by merging them, translates the nucleotide sequences to aminoacidic sequences , and creates a fasta file with the obtained sequences.
def processing_hits(df,dic_genome,distance,file_name):
    clusters = create_clusters(df,distance)
    with open(file_name,'w') as outfile:
        for accession in clusters.keys():
            for i in zip(clusters[accession][0],clusters[accession][1]):
                seq=''
                if len(i[0]) > 2:
                    begin=str(i[0][0])
                    end=str(i[0][-1])
                    if i[1] != 'negative':
                        for j in range(0,len(i[0])-1,2):
                            seq+=translation(i[0][j],i[0][j+1],i[1],dic_genome[accession])
                    else:
                        i[0].reverse()
                        for j in range(0,len(i[0])-1,2):
                            seq+=translation(i[0][j+1],i[0][j],i[1],dic_genome[accession])
                    outfile.write('>'+accession+'|'+'['+begin+':'+end+']'+'\n'+seq+'\n')
                else:
                    begin=str(i[0][0])
                    end=str(i[0][-1])
                    seq+=translation( i[0][0], i[0][1],i[1],dic_genome[accession])
                    outfile.write('>'+accession+'|'+'['+begin+':'+end+']'+'\n'+seq+'\n')
    return clusters
```

```
#Function that joins the hits in the dataframe that are within a genomic distance
def create_clusters(dataframe,distance):
    import pandas as pd
    result_final = {}
    accessions = list(pd.unique(dataframe['Accession']))
    for accession in accessions:
        intervals = dataframe[dataframe['Accession'] == accession].values.tolist()
        result = []
        strand = []
        for interval in intervals:
            if result == [] or result[-1][-1] + distance < interval[7] or strand[-1] != interval[10]:
                result.append([interval[7],interval[8]])
                strand += [interval[10]]
            else:
                result[-1] += [interval[7],interval[8]]
        result_final[accession] = [result,strand]
    return result_final
#Function that translates the nucleotidic sequences to protein sequences
def translation(start,end,strand,seq):
    from Bio.Seq import Seq
    my_seq = Seq(seq[start-1:end])
    if strand == 'negative':
        my_seq = my_seq.reverse_complement()
    prot_seq = str(my_seq.translate())
    return prot_seq
#Combines all the previous functions to identify, process, and generate the output of the nuclear receptors' hits in the tested genome
def Identification(genome_file,ref_file,genomic_distance,parameters):
    import pandas as pd
    create_database(genome_file,'nucl')
    dic = get_seqs(ref_file)
```

```
    condition = False
    for i in dic.keys():
        create_tempfile(i,dic[i],'Tem_seq_file.fasta')
        perform_tblastn('Tem_seq_file.fasta','Temp_out',genome_file,parameters)
        df = formating_dataframe ('Temp_out')
        if condition == False:
            df_final = df
            condition = True
        else:
            df_final = pd.concat([df_final,df],ignore_index=True) # concantenation of the dataframes
    df_final = remove_overlapps(df_final)
    df_final=df_final.sort_values(by=['Accession','s. start'])
    dic_genome = get_seqs(genome_file)
    processing_hits(df_final,dic_genome,genomic_distance,'Hits.fasta')
#Carries out the Blastp algorithm
def blastp(query,database):
    from Bio import SeqIO
    from Bio.Blast.Applications import NcbiblastpCommandline
    fasta_sequences = SeqIO.parse(open(query),'fasta')
    for fasta in fasta_sequences:
        sequence = str(fasta.seq)
    if len(sequence) > 30:
        command_line = NcbiblastpCommandline(query=query, db=database,evalue=0.05,gapopen=11,
        gapextend= 1,threshold=21,window_size=40,word_size=6,outfmt='6',seg='no',out
='blastp_result')
    else:
        command_line = NcbiblastpCommandline(query = query, db = database,evalue =
200000,gapopen = 9,gapextend =  1,
        threshold = 11,window_size= 40 ,word_size = 2,comp_based_stats = '0',seg = 'no' ,matrix =
'PAM30',outfmt = '6',out  = 'blast_result')
    command_line()
```

```
#Obtains the hit with the max score, if there is more than one hit with the highest score, it is used the value of the identity
def retrieve_maxiden(file):
    import pandas as pd
    df = pd.read_csv(file,sep = '\t',header = None)
    max_score = [df.iloc[:,11].max()]
    rslt_df = df[df[11].isin(max_score)]
    if  rslt_df.shape[0] > 1:
        max_iden = [rslt_df.iloc[:,2].max()]
        rslt_df = rslt_df[rslt_df[2].isin(max_iden)]
    res =[]
    for i in range((rslt_df.shape[0])):
        res.append(list(rslt_df.iloc[i, :]))
    return res[0]
#Transforms the dataframe in the final output
def final_processing(df):
    import pandas as pd
    df[['Accession','Regions']]=df[0].str.split('|',expand=True)
    df[['Proteins','NR']]=df[1].str.split('|',expand=True)
    df=df.drop([0,1,2,3,4,5,6,7,8,9,10],axis=1)
    Regions=df['Regions'].str.split('(\d*):(\d*)',expand=True)
    Regions.rename(columns={1:'Start',2:'Stop'},inplace=True)
    frames=[df,Regions['Start'],Regions['Stop']]
    df_final = pd.concat([df,Regions['Start'],Regions['Stop']],axis=1)
    df_final=df_final.drop(['Regions'],axis=1)
    return df_final
#Classifies the sequences obtained from the Identification() function by using references sequences in a fasta file
def Classification(identification_result,db_file,treshold):
    import pandas as pd
    create_database(db_file,"prot")
    result_list = []
```

```
    dic = get_seqs(identification_result)
    for i in dic:
        create_tempfile(i,dic[i],'temp_file')
        try:
            blastp('temp_file',db_file)
            result_list+=[retrieve_maxiden('blastp_result')]
        except:
            nf+= [i]
            continue
    df=pd.DataFrame(result_list)
    print(len(df))
    df=df[df[11] >= treshold] #Column 11 corresponds to the bit_score column, with this all the sequences with a score below the threshold are removed
    df_final = final_processing(df)
    return df_final
#Run the two main functions to carry out the analysis
parameters = [evalue,wordsize,gap_open,gap_extend,matrix]#list with the tBlastn parameters
Identification(genome,ref_sequences,genomic_dist,parameters)
res=Classification('Hits.fasta',ref_db,treshold)
```