Synteny guided fusion of Anopheles gene families

by

Mark Forteza

Undergraduate Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Bachelor of Science

in the
Department of Mathematics
Faculty of Science

© Mark Forteza 2017
SIMON FRASER UNIVERSITY
Fall 2017

All rights reserved.
However, in accordance with the *Copyright Act of Canada*, this work may be reproduced without authorization under the conditions for "Fair Dealing." Therefore, limited reproduction of this work for the purposes of private study, research, education, satire, parody, criticism, review and news reporting is likely to be in accordance with the law, particularly if cited appropriately.

Approval

Name: **Mark Forteza**

Degree: **Bachelor of Science (Mathematics)**

Title: ***Synteny guided fusion of Anopheles gene families***

Examining Committee:

Cedric Chauve
Supervisor
Professor _____

Marni Mishna
Professor _____

Date Defended: 15 December 2017

Abstract

Identifying gene families is an important step in comparative genomics, since genes from the same family usually perform similar or related biological functions. The process of grouping genes into families is automated, and thus often contains some errors, one particular case is a family that has been split into two clusters. Commonly used clustering techniques are reasonably accurate and corrections are generally time consuming, so a partitioning is usually accepted even with errors. In this current study, we designed a pipeline to improve the clustering of genes into families by combining synteny conservation and gene sequence similarity. We examined the cluster quality of a given clustering based on nucleotide sequence similarity and found many hints suggesting a significant number of clusters contain errors, we then identified and joined pairs of clusters showing syntenic signals and evaluated any improvement in cluster quality using statistical tests. Our method was applied to a clustering of genes belonging to a set of Malaria vectors but the pipeline can be applied on any gene clustering to improve the partitioning.

Keywords: Clustering genes; homology inference; gene families; synteny

Acknowledgements

A great many thanks to my supervisor, Dr. Cedric Chauve, for his guidance during this past eight months of research and writing. Also thanks for convincing me to write this thesis; it really was a great learning experience. Thanks to Dr. Marni Mishna for running an awesome thesis writing course. Thanks to my fellow MATH 498 students for their positive comments. Special thanks to my family and friends. They actually don't know this thesis exists but they are always supportive.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In a living organism, the biological functions of each cell are largely determined by proteins. The synthesis of these proteins is controlled by one or more specific genes present in the genome contained in the cell nucleus [4]. A **gene** can be described by a sequence of letters (nucleotides) that acts like a code which the cell reads as instructions to synthesize proteins. **Genomes** can have thousands of genes, and each gene sequence can be hundreds to thousands of nucleotides long. In general, the entire genome of an organism is passed on to its offspring. Some evolutionary events (mutations) may cause a descendant's genes to differ from its ancestor. Some events modify parts of the nucleotide sequence, while other events may create two copies of a gene in the same genome or even delete the gene entirely. Over time, evolution can cause descendants of the same ancestor to have significantly different genetic make-up, eventually creating two different species. The genomes of the new species essentially contains copies of the ancestor's genes that have evolved through such mutations. This on-going evolution of genes and genomes through generations, created the current enormous diversity of living organisms that forms a major part of our environment.

The importance of genome evolution in shaping species and ecosystems has motivated the emergence of the field of computational evolutionary biology over the last thirty years. Computational biology aims to understand genome evolution from the expanding datasets of genome sequences assembled in public databases. This thesis focuses on a particular problem of this field, the clustering of genes into gene families. In purely evolutionary terms, a **gene family** is a set of genes descending from the same ancestral gene, a definition often extended to require they have maintained a recognizable nucleotide sequence similarity. Two genes belonging to the same family are said to be **homologous**. It follows that the identification of gene families is an important problem in computational biology. In particular, homology is not only useful in studying evolution of species but also helps in studying the function of genes [25]. Even if the sequence of a gene has been obtained through a genome assembly, it is difficult to determine experimentally which trait the gene influences. However, since homologous genes originate from the same gene and have maintained similar sequences,

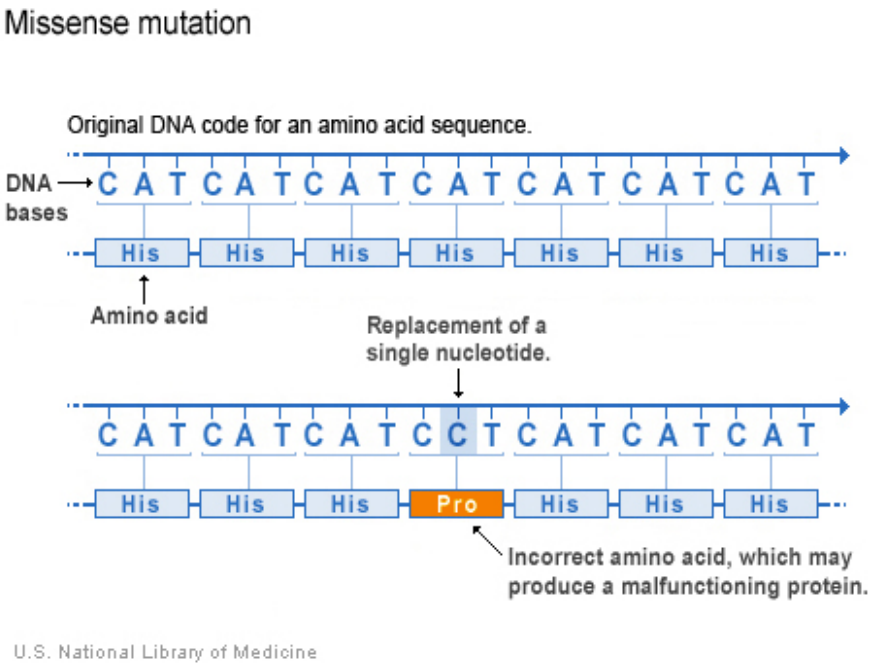

Figure 1.1: A short segment of nucleotides (DNA bases) are translated into amino acids inside the cell, amino acids then forms the protein. A small mutation in the nucleotide may affect the protein created [1]

while biological function is mostly driven by the nucleotide sequence of a gene, they are usually expected to perform similar or related biological functions as the ancestor gene. The concept of homology is thus often used in functional genomics to predict the function of genes from the characterized function of other members of the same gene family. Accurate identification of gene families is actually a key early step in many comparative genomics projects, which can involve up to tens of genomes and tens to hundreds of thousand of genes.

Identifying gene families among hundreds of thousands of genes comes down to a clustering problem. Cluster analysis is one of the major problems in data science, studied from many view points including mathematics, statistics, computer science, among others [3, 15] Clustering refers to collecting data points into groups (clusters) that are similar to each other according to some distance or similarity measure and also dissimilar to elements of any another cluster. The similarity measure between two data points is defined specifically for the application. For example, clustering points that are close together in a 2-dimensional plot will typically rely on Euclidean distance as the similarity measure (fig 1.2). In non high-dimensional applications, graph clustering is often used: data points are modeled as the vertices of a graph and each pair of vertices will have an edge whose weight is the similarity measure between its ends (Figure 1.3). Clustering is then the same as par-

titioning the vertices of the graph such that the weights are maximized inside each cluster while also minimized between different clusters.

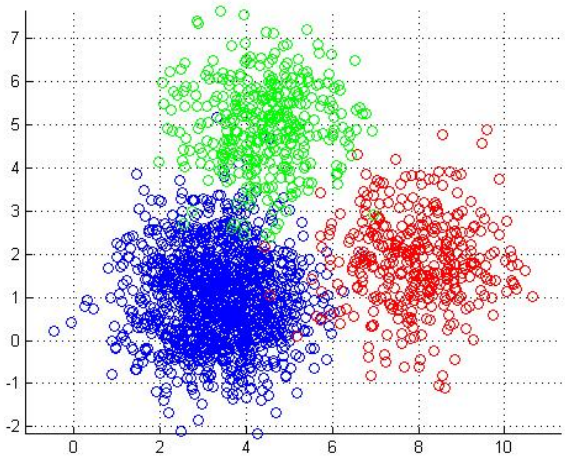

Figure 1.2: An example clustering of points in a 2 dimensional plane. Clusters are defined as points that are near each other (low Euclidean distance) and shown in the same color. Image reprinted from [7].

For clustering of genes into gene families, the genes will be the vertices. The edge weights between two genes will be a similarity measure that is high for homologous genes. In most gene family clustering methods, this similarity measure will be based on the nucleotide sequence similarity between the two genes. This approach was introduced recently, following the completion of the first sequencing projects of large and complex eukaryotic genomes; it was first seen in homology finding methods such as OrthoMCL (2003) and Inparanoid (2001) but is widely used today. Sequence comparison algorithms are used to quantify sequence similarity into a single score. Of such algorithms, the Basic Local Alignment Tool (BLAST) by Altschul [6] is the most popular and many gene family clustering methods base their similarity measures on BLAST scores. More recently, other measures of similarity have been combined with sequence similarity, in particular synteny, that considers the co-localization along chromosomes of genes (e.g. GenFamClust by Ali [5] and MSOAR by Fu [12]).

In its initial stages, when available genome sequences were few, comparative genomics projects could afford a step of manual curation, or at least inspection, of gene families obtained from unsupervised computational methods. However, following the exponential increase in sequenced genomes, it is now difficult to process in this way. As in many big data fields, clustering results can be evaluated through statistical measures, but large-scale manual analysis is out of reach. In the mean time, the increasing size of the datasets that are processed by unsupervised gene family detection methods makes it likely that these methods result in a significant number of errors, despite maintaining in general a high level of accuracy [11]. A typical reason for this phenomenon is the fact that methods parameters

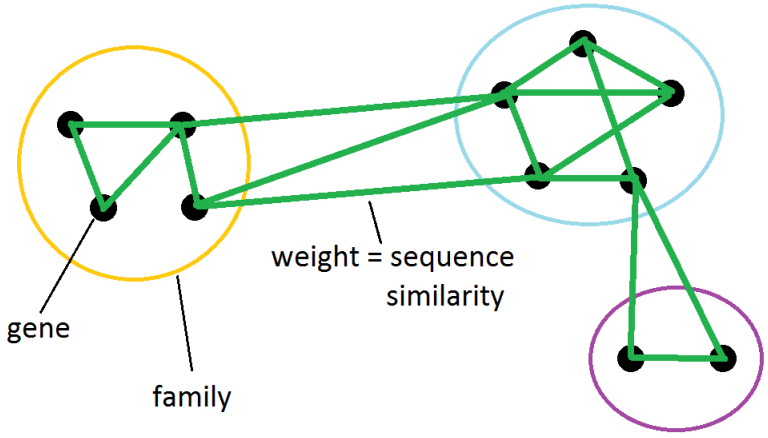

Figure 1.3: Graph model for clustering genes into families. Vertices (genes) are partitioned into disjoint subsets (gene families) such that vertices in the same partition have high edge weights (sequence similarity) between them.

are in general fixed for the whole data-set thus ignoring specificities of some gene families in terms of sequence evolution for example. The project described in this thesis aims to address this issue, at least partially. Our aim is to describe a method that corrects a given clustering of genes into gene families and to illustrate the results of this method on a specific mosquito genomes dataset.

Our motivation for studying gene families methods comes indeed from the study of malaria carrying mosquitoes. The malaria parasite and the disease it causes have been the subject of many studies. Previous research has identified that malaria is transmitted only by the *Anophiles* genus of mosquitoes. This suggests that some genes found only in the *Anopheles* genus may be an important factor in enabling the transmission of malaria. Such genes are likely to belong to the same family or families. Investigating which gene families are present in vectors but not in non-vectors can aid in identifying the genes that contribute to the likelihood of a mosquitoes to transmit the malaria parasite.

Identifying homologous genes in real life applications usually involve a large amount of data. In the case of the *Anopheles* genomes project [19], 18 species were studied with each species having around $10,000$ genes. Such a large number of genes requires automated methods for partitioning into families. In the study, *Anopheles* genes were clustered using orthoDB [27]. In addition to the partitioning method, the genome themselves were assembled using automated methods, thus giving access to more information regarding the co-localization of genes along chromosomes. Moreover, an initial exploration of the clustering of *Anopheles* genes into gene families has led us to believe that, as expected for such a large-scale project, some results were erroneous. In particular, we could observe the well known problem of having a family that has been incorrectly split into two clusters.

The work presented in this thesis is the design of a method for refining a given clustering of genes into families by identifying cases of two clusters whose union likely forms a true gene family. The method will take advantage of sequence similarity and gene order conservation. The combination of sequence similarity and synteny in clustering gene families has been used before, among others in the program GenFamClust [5]. GenFamClust quantifies synteny and combines it with sequence to similarity to define a new similarity measure for clustering. Our approach, aside from starting with pre-clustered genes, will differ by using a qualitative signal from synteny conservation to identify pairs of clusters that are likely to contain homologous genes. Two types of candidates were found using different syntenic signals: clusters with genes sharing a neighbour, and clusters that are likely tandem duplicates. We then decide whether to accept candidates by evaluating the cluster quality (sequence similarity of genes within a cluster) using silhouette analysis. Finally, false discovery rate (FDR) analysis is performed to decide whether joining a candidate pair will improve the cluster quality significantly enough. Applying this method on the previously clustered set of *Anopheles* genes at a 1% FDR revealed around 411 pairs of clusters that should be fused into a single family, thus resulting in an improvement of roughly 3% of the considered gene families.

In this thesis we will first introduce biological concepts regarding gene families, their formation and how to identify them. We then discuss clustering techniques for general graphs and also homology specific methods. We will investigate the concepts of cluster quality evaluation and multiple testing correction. Finally, we will present our methods for identifying pairs of clusters that likely forms a gene family and the pairs found after applying the method onto the given clustering of *Anopheles* mosquitoes genes. The data set used and the scripts for running our method can be found in the `Fuse_Anopheles_Families` repository [10].

Chapter 2

Background

2.1 Genome Evolution

We first introduce some relevant background regarding genome evolution. Much of the information presented here can be found in the book "Fundamentals of Molecular Evolution" by Graur and Li [14]. A genome is the complete DNA (nucleotide) information of an organism. A gene is a segment of the genome that affects a specific trait in the organism. New genomes and genes are a product of evolution. As an illustration, consider only organisms which reproduce asexually (e.g. bacteria) since sexual reproduction introduces another layer of complexity. An organism's genome is essentially a copy of the organism's parent's. However the genome replication mechanism is not perfect; among the millions of nucleotide making up the genomes, some blocks of letters may be incorrectly copied. This phenomenon is referred to as **mutation**. Depending on how much of the genome is mutated, traits expressed by the organism may be affected. This works similarly in sexually reproducing organisms, although other factors might be involved. Over many generations, the descendant's genome and traits may be quite dissimilar from its ancestor's. Mutation is one of the main mechanisms of evolution. Some mutations may alter parts of a gene (nucleotide-level mutations) while some events may involve an entire gene such as a duplication of an entire gene in the genome (gene-level mutations). Nucleotide-level mutations may also accumulate to result in a gene-level mutation, such as gene loss for example.

Gene level mutations contribute to the evolution of gene families (creation of new families, expansion or contraction of existing families). Given a gene family, its content, defined as the number of copies of the gene in each considered species, is the result of an evolutionary process whose main events are **speciation**, **gene duplication**, **gene loss**, **gene transfer**. These evolutionary events are illustrated in Figure 2.1. Speciation events occur when a species experiences enough evolution to result in two different species. Genes in the new species are considered to be distinct even if the genes originated from the same gene in the ancestral species, and as a consequence the two copies of the original gene evolve inde-

pendently. Duplication occurs when a new copy of an existing gene appears in the genome of a species. The two copies are considered to be distinct genes and again to evolve independently. A gene loss is the deletion of a whole gene from the genome, either as the result of accumulating mutations or of the deletion of a chromosomal segment. A gene transfer occurs when a distinct species "gives" part of its genetic material to another species. A combination of these gene-level evolutionary events can occur in a single generation or over many generations, resulting in the, sometimes complex, structure of gene families observed in a set of species.

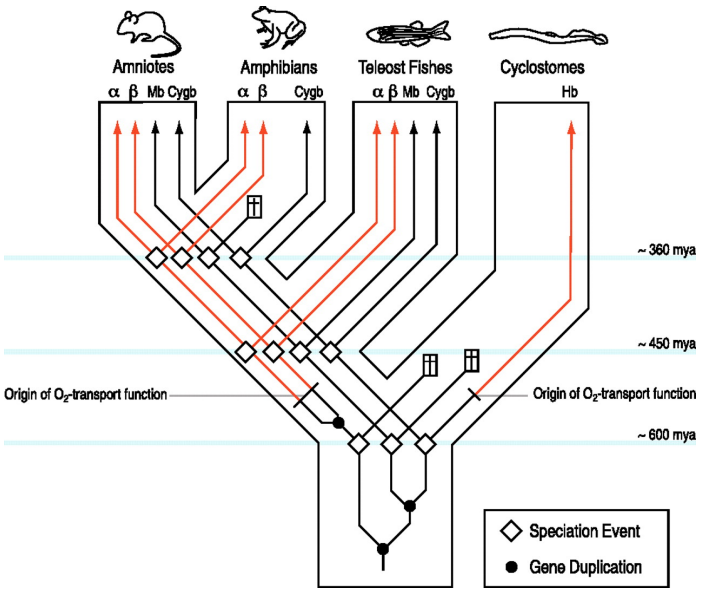

Figure 2.1: An evolution tree showing gene-level evolutionary events. The ancestor species and and one of its genes are at the bottom of tree. Two duplications created three copies of the gene in the genome from 6$\tilde{0}$0 mya (million years ago). A speciation event at 400 mya created two branching species and two copies of every gene. Speciation events at 600 mya and 300 mya did not produce copies of all genes because of gene deletions (denoted by a cross inside a rectangle). Image reprinted from [16].

Evolution affects not only specific genes but also entire genomes. A genome can be visualized as a sequence of genes, often divided into segments called chromosomes. Mutations, on large nucleotide segments or accumulated over time, can affect multiple genes. One such event which will be of interest to us later is **tandem duplication**. When a gene g_0 undergoes duplication (creating genes g_1 and g_2) and is later followed by speciation, copies of g_1 and g_2 will appear together (in tandem) in the observed genomes (Figure 2.2). Some mutations may also alter the order of genes on the genome. Duplications and deletions are obvious examples, adding or deleting genes from the gene sequence. Genes may also be moved to another location on the genome. Another important mutation is an inversion of a gene's orientation. The orientation is the direction in which a gene is read by protein building mechanisms in the cell. Figure 2.3 shows gene order in closely related genomes of

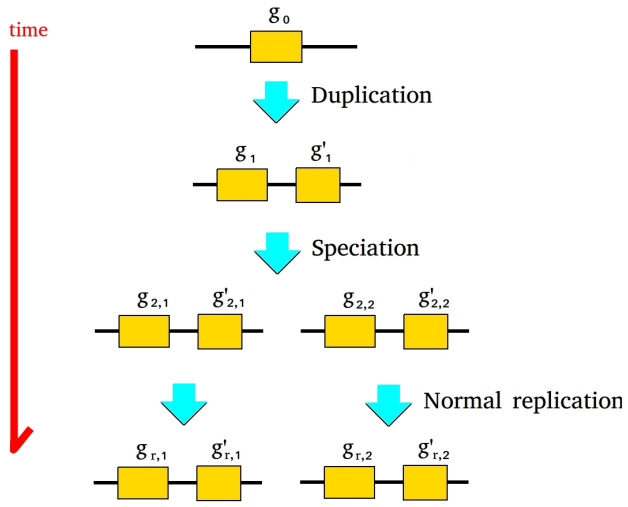

Figure 2.2: Evolution of tandem duplicates

bacteria. In a later section, we will show how gene order (and orientation) provides a signal for identifying homologous genes.

2.2 Inferring Gene Families

After this background in genome evolution, we may now examine how to identify genes belonging to the same family. Since homologous genes descend from a shared ancestor, they likely have very similar nucleotide sequences. This idea is supported by the fact that the process of clustering genes into families is usually applied on genes belonging to closely related genomes, and thus, we can expect that evolution has not yet drastically altered their gene structures. Comparing two nucleotide sequences is formally called sequence alignment, and can also be applied on protein sequences. Algorithms like BLAST (Best Local Alignment Search Tool) by Altschul et. al. [6] finds an optimal alignment of two sequences and computes the sequence similarity. In practice the algorithm actually implements a heuristic approach for the sake of time complexity. We present a sample alignment using the web based BLAST program [2] in Figure 2.4. The outputted **bit scores** are used to define sequence similarity between the input nucleotide sequences. To make sense of the bit score, one can look at the **expect value** which is the expected number of sequences in the database that will achieve a score greater or equal to the current alignment score. An expect value close to zero means that the input sequences have high similarity.

Sequence comparison is a widely accepted method for identifying homologs, but it is not perfect as high sequence similarity between genes does not necessarily imply a shared ancestry. For instance, two genes with similar nucleotide sequences may have evolved from two unrelated ancestor genes [13]. False negatives are also possible if two duplicated genes evolve in drastically different ways over many generations.

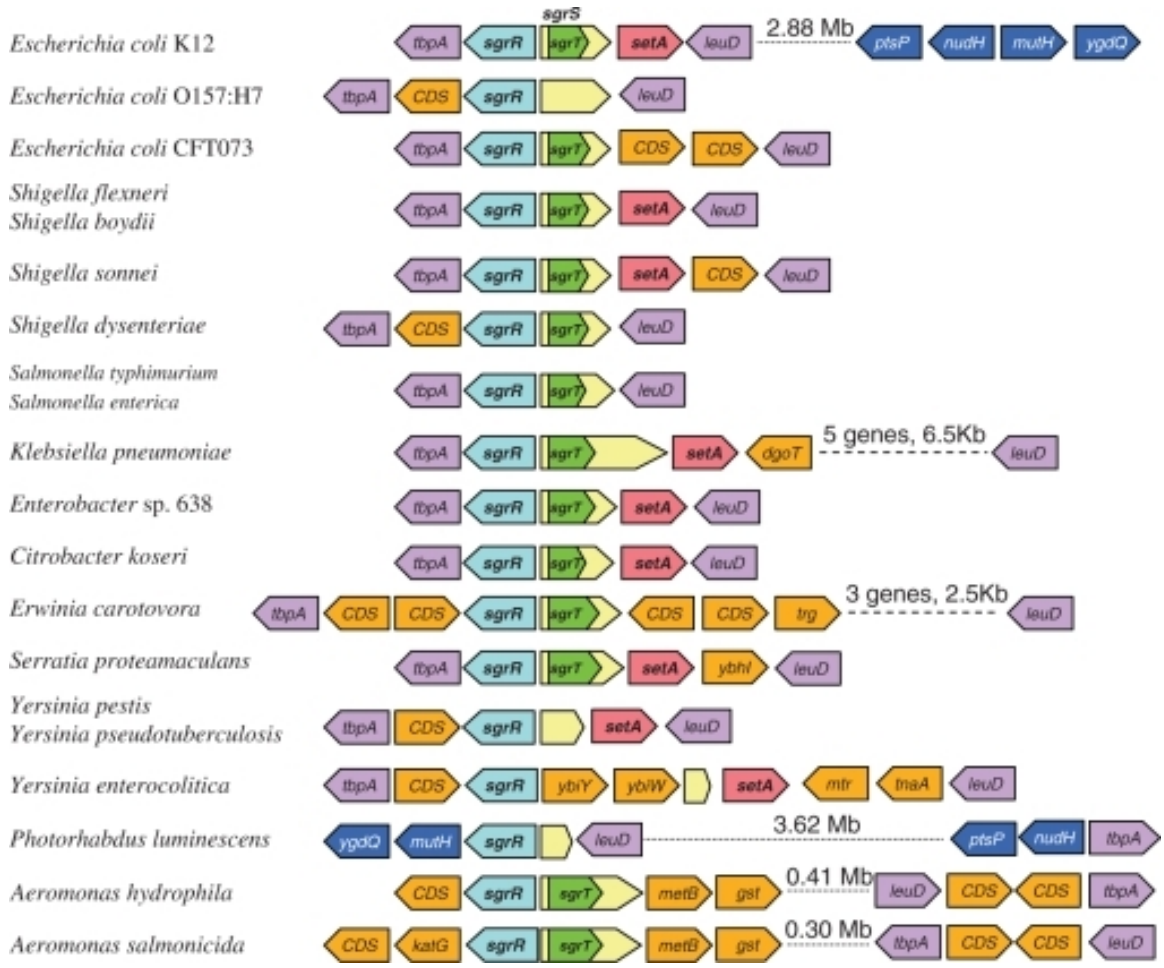

Figure 2.3: Gene order illustrated using genomes of closely related species of bacteria. Arrows indicate gene orientation. Genes of the same color are homologous. We observe that gene order tends to be conserved although some alterations (e.g. insertions or deletions of some genes) occur. Image reprinted from [17].

In order to improve the accuracy of gene family inference methods, we can benefit from more information to supplement sequence similarity. Of particular interest for us is the notion of **synteny conservation**. In genomics, synteny refers to gene order and localization along chromosomes. A genome can be visualized as a sequence of genes, often divided into multiple segments called chromosomes. As the genome evolves, the genes tend to stay in the same relative order since evolutionary events modifying gene content or order are quite rare events compared to nucleotide mutations. So when comparing genomes that diverged recently (via speciation), we can expect to find a conserved gene order across the genomes. For example, consider two genomes containing genes A_1 and A_2 which were grouped into different families by a sequence alignment. If A_1 has a neighbors B_1 and C_1, A_2 has a neighbors B_2 and C_2, and $\{B_1, B_2\}$ and $\{C_1, C_2\}$ are homologous, then we may suspect that A_1 and A_2 also belong to the same gene family. Over many generations, gene

```
> Input:
>
> Sequence 1:
> CCGTGCCTGATGAACGATAGCCAGCTAGGGCCCTATAAGCCTAGGATAGCAGTCGG
>
> Sequence 2:
> CCGTGCCTTGAACGATTGCCCTGTAGGGCATAAGCCTAACTAGGATAGCAGTCGG
```

> Output

Sequence ID: Query_238207 **Length:** 55 **Number of Matches:** 1

Range 1: 1 to 55 Graphics ▼ Next Match ▲ Previo

Score	Expect	Identities	Gaps	Strand
44.6 bits(48)	9e-11	47/60(78%)	9/60(15%)	Plus/Plus

```
Query  1   CCGTGCCTGATGAACGATAGCCAGCTAGGGCCCTATAAGCCTA----GGATAGCAGTCGG  56
           ||||||||  ||||||||  |||   ||||||   |||||||||    ||||||||||||
Sbjct  1   CCGTGCCT--TGAACGATTGCCCTGTAGGGC---ATAAGCCTAACTAGGATAGCAGTCGG  55
```

Figure 2.4: Sample BLAST alignment with a graphic output, produced using a sequence vs. sequence, `blastn` query of the NCBI web based BLAST service [2]. Two nucleotide sequences are accepted as input. In the graphical output, vertical lines are drawn where the sequences match. `Score` is the bit score used as sequence similarity. `Expect` is the expected number of sequences found in a database that will produce a greater or equal alignment score. `Identity` is the percentage similarity between sequences. For this dataset, an Expect value of $9e^{-11}$ means that it is highly unlikely that such a high sequence similarity happens by chance.

level mutations can accumulate to significantly alter gene order. As such, synteny is not enough to find gene families, but gene order can reinforce sequence similarity signals to identify homologous genes.

We described the signals that can be extracted from genomics data to tell if a pair of genes are homologous. However, real applications often involves multiple genomes each with thousands of genes; at such scale, these signals that can lead to detect homologous genes need to be abstracted into mathematical structures that can be processed by unsupervised methods. This is essentially a clustering problem, using mostly (but not uniquely depending on the methods) a sequence similarity based score as the similarity measure between two data points. We will discuss clustering in the following sections.

2.3 Similarity Measure

Before we describe how to perform clustering algorithms for clustering genes into families, we need to define a similarity measure. The BLAST algorithm is one of the most used methods for quantifying sequence similarity between pairs of genes. Compared to many sequence alignment algorithms before it, BLAST is accurate and efficient enough for practical applications. Due to this efficiency, many sequence similarity measures have been developed based on the BLAST algorithm. One problem with BLAST is that it can be misled by domain insertions, or the insertion of a large segment of nucleotides in two non-homologous genes. This issue is better explained in Figure 2.5. Significantly large domain insertions in two genes that do not belong in the same family can result in a large BLAST score between the two genes. Thus, a clustering method using BLAST score as similarity measure may incorrectly place the genes in the same cluster.

One way to improve on the alignments found by BLAST is to distinguish between sequence similarity caused by homology and similarity caused by domain insertions. **Neighborhood Correlation** by Song et. al. [23] achieves a refinement on BLAST scores between two genes by investigating the set of genes that are similar to both genes. As before, define a graph where genes are vertices and edges are weighted using the BLAST score between its ends. If we delete edges whose weights are below a certain threshold, then the neighbors of a gene will be those that aligns well with the gene. When two genes x and y are identified to have a large BLAST score, we can look at the intersection of the neighborhoods of x and y. If the shared neighbors have high similarities between them, then it is unlikely that the similarity between genes x and y was caused by a domain insertion. This idea is captured by a computation of correlation between BLAST scores of two genes:

$$NC(x,y) = \frac{\sum_{i \in N}(S(x,i) - \overline{S}(x))(S(y,i) - \overline{S}(y))}{\sum_{i \in N}(S(x,i) - \overline{S}(x))^2 \sum_{i \in N}(S(y,i) - \overline{S}(y))^2} \tag{2.1}$$

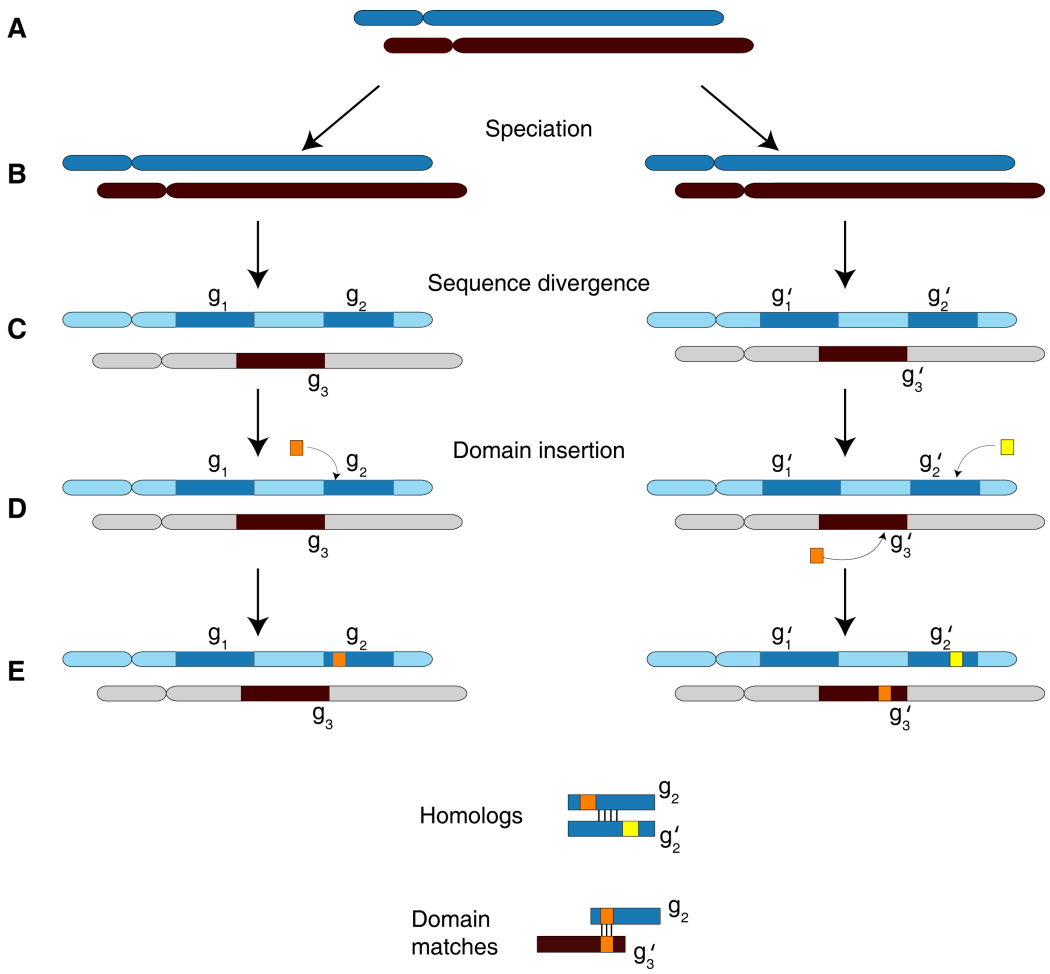

Figure 2.5: Sample evolution of genes to illustrate alignment problems when domain insertions are involved. (A) Two chromosomes in a genome. (B) A speciation event creates two copies of the genome. (C) Mutations alter parts of the nucleotide sequences except for the darker colored genes or regions. (D) Orange and yellow segments of nucleotides are inserted. (E) Current genomes to be examined by sequence alignment. Homologous genes g_2 and g_2' will be identified despite of the domain insertions since majority of their sequences still align well. Genes g_2 and g_3' will align at the orange segments so the resulting alignment score may be substantially high despite being non-homologs. Image reprinted from [23].

where N is the set of genes, $S(x, i)$ is the BLAST score between x and i, and $\overline{S}(x)$ is the mean BLAST score $S(x, i)$ over all elements of N. NC scores lie in the range $[0, 1]$ with a higher score indicating high similarity between two genes. This NC score will be used in our clustering methods as an alternative similarity measure to BLAST scores.

2.4 Clustering

Having defined the similarity between two points, we are now ready to partition the data set. Clustering is the process of partitioning data points into clusters such that points in a cluster are similar to each other and dissimilar to points in other clusters. The definition of similarity (similarity measure) depends on the particular application. For example if each data point has a list of quantifiable characteristics, we may provide these scores in a vector and define the similarity between two data points as the Euclidean distance of their respective vectors. For one dimensional applications, as in our case, we can apply graph clustering methods. We define a graph whose vertices are the data points and edges are weighted with the similarity measure.

In this section we will describe commonly used clustering methods; both general and homology specific. The general clustering techniques presented here can be applied on graphs as well as higher dimensional applications. Other graph specific clustering methods takes advantage of graph structures, like the Markov Clustering Algorithm [26]. Our goal of improving clustering of genes into families may not require clustering from scratch, but we feel that discussing clustering methods will give the reader a better insight into the clustering process. The main sources for this section are the books *Data Mining and Analysis* [28] and *Introduction to Data Mining* [24].

2.4.1 General clustering techniques

A natural way to perform clustering is to partition vertices into non-overlapping subsets. This general method is aptly called **partitional clustering**. The **k-means** algorithm is a commonly used example of this clustering pattern. The user chooses k to be the number of clusters. The algorithm then chooses k vertices randomly or semi-randomly as the **centroids** of each cluster. Non-centroid vertices are assigned to the cluster of the nearest or most similar centroid. We calculate the centers of the resulting k clusters and assign those as the new centroids. This process is repeated until assignments do not change. The k-means algorithm is also classified as a **representative based clustering** due to the use of one vertex to represent each cluster. The algorithm is widely used because of its speed. However, the number of clusters needs to be known a priori. This is rarely the case when clustering genes into families.

To avoid the requirement of defining a number of clusters, we may perform a **hierarchical clustering**. Clusters can be divided into subsets to create a nested clustering. The

clusters can be visualized in a rooted tree called a dendogram where the root is a cluster containing all data points and the leaves are singletons or clusters containing only one data point (Figure 2.6). Each level of the tree is a clustering which can be evaluated using cluster validation techniques (Section 2.5) in order to choose the appropriate number of clusters. The disadvantage is that hierarchical clustering may be computationally expensive and requires a lot of storage space since all levels of the hierarchy must be computed and stored.

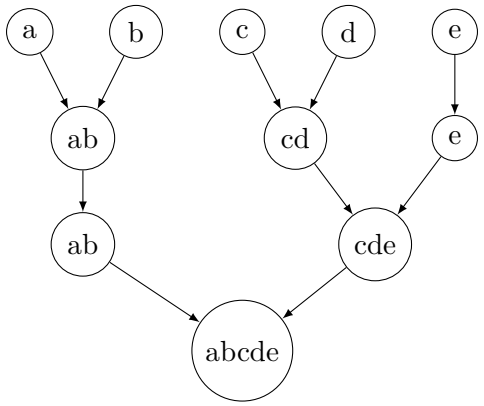

Figure 2.6: A heirarchical clustering of the data set $\{a, b, c, d, e\}$. Note that each level of the tree is a clustering, i.e. a disjoint partition of the data set.

Another limitation of representative based clustering like the k-means algorithm is that two very similar data points may be placed in different clusters because they are close to different centroids. **Density based clustering** fares better in these situations by using the density around a vertex instead of distance between vertices. **DBSCAN** is the classic example of density based methods. For each vertex x we define its neighborhood $N_r(x)$ as the set of vertices y such that $d(x, y) \geq r$, where $d(x, y)$ is the similarity between x and y. Given a user defined threshold density p, x is a *core point* if $N_r(x)$ contains at least p elements. If $|N_r(x)| < p$ but x belongs to the neighborhood of some core point z then x is a *border point*. Otherwise, x is considered an outlier. Two points x and y are *density reachable* if we can find a sequence of point $(x = x_0, x_1, x_2, \ldots, x_k = y)$ such that $x_i \in N_r(x_{i-1})$ and x_{i-1} is a core point. In other words x and y are density reachable if we can connect core points that are in each other's neighborhoods. Now x and y are *density connected* if there exists a core point z such that x and y are density reachable from z. A cluster is a maximal set of points that are density connected.

2.4.2 Homology-specific methods

While general graph clustering methods can be applied to any graph whose vertices represent genes and edges are weighted by a measure of similarity, in practice gene families are inferred by algorithms designed specially for this purpose. These homology specific methods take

advantage of structures found in genes, but also applies (or borrows techniques from) general clustering methods.

As discussed earlier, we are working on a data set of mosquito genes, for which an initial set of gene families was computed using **OrthoDB** [27]. OrthoDB may refer to the database of orthology related information and also the underlying algorithm to find orthologous genes (homologs created by speciation events). To describe the algorithm, we need to define the concept of **best reciprocal hits** (BRH). Using a sequence similarity score like the one obtained from BLAST, we get *hits* (scores) between each pair of genes. Consider two genomes A and B and two genes a and b, with $a \in A$ and $b \in B$. If the best hit with gene a, $hit(a, y)$, among all $y \in B$ is with $y = b$, and the best hit $hit(x, b)$, among $x \in A$ is with $x = a$, then genes a and b are each other's best reciprocal hits. Now we can describe the OrthoDB algorithm.

- First, it identifies the BRH of genes between all pairs of genomes of the given data set. The resulting gene pairs are assumed to be homologous genes resulting from speciation events and to belong to the same gene family.

- Next, it clusters genes by forming triangulating best reciprocal hits. Figure 2.7 explains this clustering procedure. We begin with an edge between genes a and b, which corresponds to the highest hit score amongst all BRH. By definition of BRH these genes are present in different species, $a \in A$ and $b \in B$. We then add the BRH of a and b, gene c, from a third species C. Once this triangle is formed, we can greedily add genes that are BRH with genes already in the cluster. Forming triangles with the newly added genes is not necessary.

- The last step is identifying hits between genes in the same genome that are better than the BRH between genomes (i.e. find genes a, a' in the same genome A such that the BRH is $hit(a, b)$ and $hit(a, a') > hit(a, b)$). The pairs found are likely homologous genes created by duplication after the most recent speciation event and also to belong to the same gene family. Suppose we have identified (a, a'), (b, b'), and (c, c') to be such high scoring pairs from species A, B and C, respectively. If a', b' and c' forms a triangulation, then we can add the three genes to the cluster containing a, b and c found in the previous step.

Some of the earliest clustering algorithms that are still used today are InParanoid by O'Brien [21], and OrthoMCL by Li [18]. Both algorithms follow a similar process as OrthoDB: identifying two types of homologs using best hits, then clustering. InParanoid (and OrthoDB) has a flaw, in that it can be misled by non-homologs with very similar sequences (leads to false positive best hits) and gene deletions (best hit will not be a paralog). OrthoMCL addresses this problem by including recent paralogs in the identified best hits. OrthoMCL then proceeds with a Markov clustering algorithm [9], which is based on random walks on the graph using a Markov matrix.

As mentioned in Section 2.2, synteny or gene order can be used to support sequence similarity signals to identify gene from the same family. **GenFamClust**[5] combines the two signals to cluster homologous genes.

2.5 Cluster Evaluation

To remind the reader, the goal of the current work is to design a method to improve a previously clustered set of genes. It is often difficult to produce a perfect clustering: in our case, a clustering in which every cluster contains all genes sharing one ancestor. For instance, two homologous genes may evolve, over many generations, to have fairly different nucleotide sequences. A clustering algorithm may then incorrectly place the two genes into two clusters. In more general examples using the distance concept, two nearby points may be placed in different clusters if they are closest to different centroids. There is also the problem of not knowing the optimal number of clusters, or in our application the true number of gene families. We now investigate the concept of evaluating a clustering. Clustering validation methods can be used to evaluate clustering results or a clustering algorithm itself. We may evaluate the "goodness" of a single cluster or all clusters in order to find the optimal parameters for clustering algorithms, like number of clusters or density thresholds.

Many validation measures exist, each with their own advantages and disadvantages. Cluster validation methods can be categorized into two main types: internal and external. Internal validation methods uses only information available with the data set, while external measures uses additional information such as previously known results. The choice of which evaluation tool to use not only depends on the availability of external data but also the clustering method applied. Some validation methods are designed for use with specific clustering methods. Results from cluster validation are not consistent across all methods so it is important to choose a validation measure that works best with the clustering method applied.

2.5.1 External Validation

External validation, also called supervised evaluation, compares a clustering to a known correct partitioning. Data points are often given class labels, where class refers to the true

partition the point belongs to. We will define some notation to be used in this section. Let S be a set of points and $|S| = N$. Let $P = \{P_1, P_2, \ldots, P_k\}$ be the partitioning found by a clustering algorithm. Let $C = \{C_1, C_2, \ldots, C_{k'}\}$ be the reference partitioning. We are often interested in the size of the overlap between the two partitions, so let $n_{ij} = |P_i \cap C_j|$, $n_i = |P_i|$ and $n_j = |C_j|$.

Purity and **entropy** both measures how much of a cluster belongs to a single class. Both methods are designed to evaluate k-means clustering. The formulas are as follows:

$$\text{purity} = \frac{1}{N} \sum_i \max_j n_{ij} \tag{2.2}$$

$$\text{entropy} = -\frac{1}{N} \sum_i \sum_j n_{ij} \log \frac{n_{ij}}{n_i}. \tag{2.3}$$

The computed purity lies in the range $(0, 1]$, while entropy lies between $[0, \log k']$, where k' is the number of classes in the reference partitioning. A high score indicates a good clustering in both measures.

The **Rand measure** [22] computes the number of correct classifications made by a clustering algorithm. The formula can be defined informally as $(a + b)/N$, where a (true positives) are the number of pairs in S that were correctly placed in the same cluster, and b (true negatives) are the number of pairs that were correctly placed in different clusters. More formally, we compute the Rand index as

$$RI = \frac{\binom{n}{2} - \sum_i \binom{n_i}{2} - \sum_j \binom{n_j}{2} + 2 \sum_{ij} \binom{n_{ij}}{2}}{\binom{n}{2}}. \tag{2.4}$$

Here $RI \in (0, 1]$ and a score of 1 means a perfect clustering.

The **F-measure** combines the concepts of precision and recall. Precision is defined as the number of true positives out of all positives found by an algorithm. Recall is the number of true positives out of all positive elements possible. We can compute the F-measure as follows

$$F = \sum_j p_j \max_i \left(\frac{2 \frac{p_{ij}}{p_i} \frac{p_{ij}}{p_j}}{\frac{p_{ij}}{p_i} + \frac{p_{ij}}{p_j}} \right). \tag{2.5}$$

The F-measure lies in the range $(0, 1]$ where a higher score indicates a good clustering.

External validation measure require previous knowledge or an expert to produce class labels. This is not applicable to our gene family clustering since we do not know the correct partitioning.

2.5.2 Internal Validation

Internal or unsupervised validation evaluates cluster quality without any external information. The only information used is the similarity measure applied on the initial clustering. From the definition of clustering, the goal is to partition a data set into clusters such that points in the same cluster are similar and points in different clusters are dissimilar. Internal validation techniques focus on how well a clustering satisfies the goal of compactness and separation. We define **compactness** to be the similarity of points inside a cluster, while **separation** is the similarity or dissimilarity of two clusters. Many internal validation methods are based on these two concepts.

For distance based clustering methods we can define some simple validation methods based on distance from clusters' centroids. The **root-mean-square standard deviation** (RMSSTD) measures the compactness of points inside a cluster by computing the distance of points from the cluster's centroid. We define

$$\text{RMSSTD} = \left(\frac{\sum_i \sum_{x \in C_i} ||x - c_i||^2}{|P \sum_i (n_i - 1)|} \right)^{\frac{1}{2}} \tag{2.6}$$

where C_i is the ith cluster, c_i is its centroid and $n_i = |C_i|$. A lower value of RMSSTD means a better clustering. **R-squared**, on the other hand, compares the distance within clusters to the distance across all data sets.

$$\text{R-squared} = \frac{\sum_{x \in S} ||x - c||^2 - \sum_i \sum_{x \in C_i} ||x - c_i||^2}{\sum_{x \in S} ||x - c||^2} \tag{2.7}$$

where S is the set of all data points and c is the center of S. The computed R-squared score lies in the range $[0, 1]$ where a score of 0 indicates a perfect clustering. In both RMSSTD and R-squared measures a lower score means that data points inside each cluster are nearer to each other, thus indicating a better clustering.

The **Dunn's index** compares the maximum distance between two points inside a cluster (compactness) to the minimum distance between two clusters (separation). For distance $d(x, y)$ (or another similarity measure) between points x and y, the formula is as follows:

$$\text{Dunn's index} = \min_i \left(\min_j \left(\frac{\min_{x \in C_i, y \in C_j} d(x, y)}{\max_k (\max_{x,y \in C_k} d(x, y))} \right) \right). \tag{2.8}$$

The higher the Dunn's index the better the clustering.

The **I-index** has the same compactness/separation scheme as the Dunn's index but combines it with the distance from centroid concept. The formula is given as

$$\text{I-index} = \left(\frac{1}{N} \cdot \frac{\sum_{x \in S} d(x, c)}{\sum_i \sum_{x \in C_i} d(x, c_i)} \cdot \max_{i,j} d(c_i, c_j) \right)^P \tag{2.9}$$

where N is the total number of data points. A higher score means a better clustering.

The **silhouette index** compares the similarity between all points in a cluster (compactness) against the inter-cluster similarity with all points of the closest cluster. For a point x, let $a(x)$ be the average distance from x to points in the same cluster, and $b(x)$ be the average distance from x to points in the closest cluster not containing x. Then the silhouette is simply

$$\text{Silhouette} = \frac{1}{N} \sum_i \left(\frac{1}{n_i} \sum_{x \in C_i} \frac{b(x) - a(x)}{\max(b(x), a(x))} \right). \tag{2.10}$$

Silhouette coefficients lie between $[-1, 1]$, where a score of 1 is a perfect clustering.

The formulas here are given for use with distance, but they can easily be manipulated to work with similarity measures where a high score means two points are similar. For all of the described cluster validation methods, we can find the optimal clustering by adjusting clustering parameters and maximizing the validation measure. This process requires multiple runs of the clustering algorithm and will take a lot of time regardless of the choice of clustering method and validation measure. A quicker solution is to use an cluster evaluation index to compare the quality of a cluster before and after a certain action on the cluster. Performing such an evaluation will require some sort of statistical test to determine if the cluster quality improves significantly enough.

2.6 Statistical Tests

Our clustered set of *Anopheles* genes does not come with a provided gold standard, in this case class labels indicating which family every gene belongs to, as we do not know the true evolutionary history of the genes. An internal validation measure is therefore required to evaluate our clustering. After we modify the clusters in hopes of improving the clustering, we need to determine whether an improvement in cluster quality corresponds to finding better gene families. In general, when trying to decide whether to accept a score (e.g. cluster quality) resulting from an experiment, we look at confidence values to measure the statistical significance of the score. These values describe the likelihood of a score to be found randomly, and scores that are unlikely (passing some threshold) to be found at random are deemed to be statistically significant. In order to use these confidence measures we need to define a null hypothesis. A null hypothesis is essentially a set of scores that can be found at random, but without violating the structure of the original data set. For example if we are interested in the weight of a mass produced object, then the weight likely falls within a certain range. Commonly used confidence values are the p-value and the false discovery rate. We will discuss how to use these measures.

Given a null hypothesis, we are usually interested in scores that passes a certain threshold. Let n_t be the number of scores in the null hypothesis that is greater or equal to a threshold t. Let n_N be the size of the null hypothesis. Then the **p-value** is simply the probability that an observed score (or the threshold) can be achieved in the null hypothesis, $p = n_t/n_N$. To decide whether to accept a computed p-value as statistically significant, we need to compare it to a confidence threshold α. We usually set $\alpha = 0.01$ or $\alpha = 0.05$. This becomes a problem when the data set becomes too large. If we are interested only in the top scores, the p-value becomes very small and does not provide much insight to the significance of the score. To solve this problem, we apply the **Bonferroni adjustment**. Suppose we computed n scores and chose a confidence threshold α. Then a score is statistically significant if the p-value is $\leq \alpha/n$. This simple adjustment is enough to guarantee that a passing score is unlikely to be found at random.

Unlike the p-value method where compute the likelihood of an observed score to be found randomly, the **false discovery rate** (FDR) determines a set of observed scores that is unlikely to be found at random. Suppose that the observed data set and the null hypothesis have equal sizes. For a threshold t, define s_b to be the number of observed data (candidate pairs) with a score above t, and s_n to be the number of null data with higher scores than t. Then we define FDR $= s_n/s_b$. In practice, we usually set an accepted FDR, in the same way we use the confidence threshold α. We then find the score threshold t that satisfies the given FDR.

The question now is when to use which statistical test. FDR analysis usually finds more significant scores than the p-value with Bonferroni adjustment. However, as these statistical test deal only with the likelihood of score appearing in random, there is no guarantee that all accepted scores correspond to correct results. Therefore FDR analysis may find more true positives but also find more false positives than the p-value method. The choice of confidence measure may depend on how follow-up validations will be conducted. If validations will be performed on accepted results as a group, the false negatives from FDR analysis may not be a problem. If we need to investigate results one by one, the p-value analysis is a better choice.

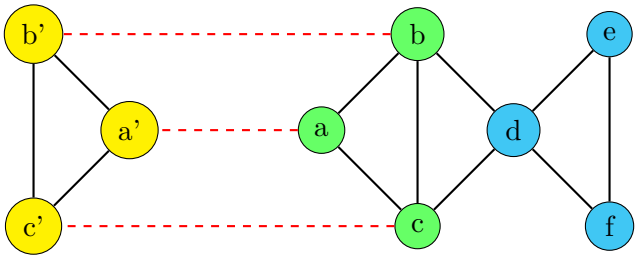

Figure 2.7: Sample cluster illustrating OrthoDB triangulation. Solid black edges correspond to best reciprocal hits (BRH) and dashed red edges indicates hits between genes in the same species that is higher than the BRH.

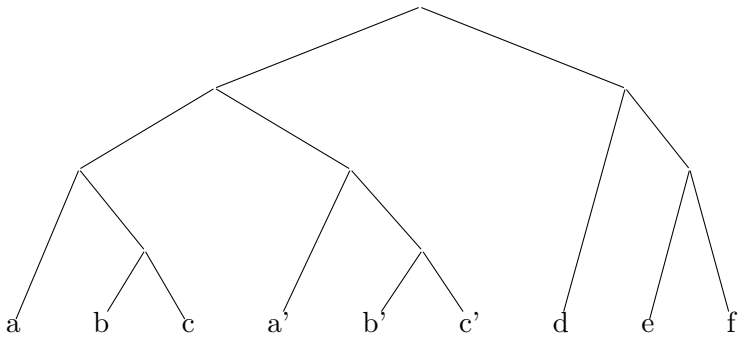

Figure 2.8: Inferred gene evolution tree from the clustering of genes in figure 2.7

Chapter 3

Methods

As mentioned previously, a graph is used to model the gene clustering (Figure 1.3). Vertices are the genes, and edge weights contain the sequence similarity between two genes. We compute sequence similarity using BLAST as a base hit scores and refine the scores using Neighbourhood Correlation. The resulting score, $NC(x, y)$ where x and y are genes, will lie in the range $[0, 1]$ where a higher score corresponds to a stronger sequence similarity between x and y. We compute NC score for every pair of genes, but to save computational resources we only draw edges between vertices with NC score of at least 0.3. On the resulting graph, non-adjacent vertices will be considered to have a similarity $NC = 0$.

We developed a method that will detect pairs of clusters (in the graph defined previously) whose union potentially forms a true gene family. The first step is to identify candidate clusters pairs (F_1, F_2), using syntenic features of genes in F_1 and F_2. Then we decide whether these candidates pairs of clusters are likely to form a true family if combined together. This is done by evaluating statistical properties of the resulting cluster using the concept of **silhouette**, and investigating whether improvements in the silhouette measure is likely to occur randomly under a fixed **false discovery rate**.

3.1 Identifying candidate pairs

A possible cause for a family to be split into two clusters (F_1, F_2) is that the sequences of the genes in these clusters were not deemed to be similar enough to each other to pass the similarity thresholds used to define families. To find these pairs we rely on signals given by a common order of genes in F_1 and F_2. Two types of candidate family pairs are examined. Both types satisfy a specific gene order conservation pattern, as a first requirement.

3.1.1 Shared neighbour and complementary species candidates

Since the considered genomes belong to the same genus, it is likely that a gene family is present in all species, although some genome assemblies are highly fragmented and subject

to the common issue of unassembled genes [8]. In a preliminary analysis of the dataset, we observed that many families are present in a few species only and many are also missing in a few species. This leads to a suspicion that families which appears in complementary species sets might "fit" together to form a family that is present in all species. Therefore, we looked for pairs of families (F_1, F_2), where F_1 appears in k genomes only, F_2 appears in all species except the k families in which F_1 is present. It is possible that a family is deleted from some species, mostly due to genome assembly issues, so the search was extended to families $F1$ in only k species, and families F_2 which are missing in k up to $k+2$ species. Complementary species spans is too broad of a criteria to suggest that a family split so we use synteny to narrow down the candidates. We look for a shared neighbor, i.e. a gene from each of F_1 and F_2 is adjacent to a member of some other family F_3 (see Figure 3.1).

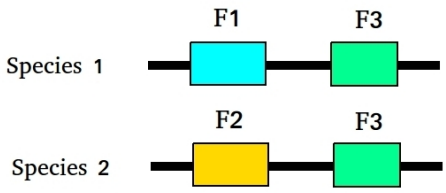

Figure 3.1: F_1 and F_2 share a common neighbor F_3

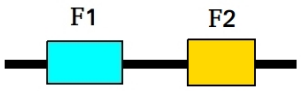

Figure 3.2: F_1 and F_2 are tandem duplicates

3.1.2 Tandem duplicates

We also noticed many occurrences of a pattern that suggests that families evolving through tandem duplication might be split into two families. These are families F_1 and F_2 originating from genes g and g', respectively, where g and g' are adjacent along a genome and are duplicates of one ancestral gene g_0. Figure 3.2 illustrates this type of evolution. Clustering methods might separates copies of g and copies of g' and create a new family with ancestral gene g' especially if gene sequences diverge significantly over time, while from a formal point of view the genes descending from g and g' belong to the same family. However, synteny is often conserved and we often find a copy of g and a copy of g' next to each other in the genome. Gene deletions, highly fragmented genome assemblies, and extreme nucleotide level mutations may prevent us from seeing synteny in some species so we considered pairs which are adjacent in almost all species where both families are present. In particular, candidates are defined as pairs (F_1, F_2) such that

$$m \leq |F_1| \leq |F_2| \quad \text{and} \quad \frac{\#\text{ of observed synteny}}{\text{size of } F_1} \geq r, \tag{3.1}$$

where minimum size m is an integer, r is a ratio, and $|F|$ is the number of genes in the family F.

3.2 Fusion Test

If two genes are homologous, then they are likely to display a syntenic order. The converse is not guaranteed to be true, however. An observed gene sequence conservation, even though it is a good starting point for our project, does not imply that genes belong to the same family. The next step in our method is performing a statistical test in order to decide whether candidate pairs of clusters are likely to form a true gene family. A union of two clusters is likely a true gene family if the cluster quality of the union is better than the initial clusters. In order to quantify the quality of a cluster, we use the textbfsilhouette measure [22, 24], an unsupervised cluster evaluation statistics described in Section 2.5. We compute the silhouette of each cluster. Then we fuse each pair of candidate families, and compare the silhouette of the union to a scaled average of the silhouette of the original clusters. Finally, we test for the statistical significance of any improvements in silhouette using the notion of **false discovery rate** (FDR) [20] defined in Section 2.6.

Let us recall the principles and definition of silhouette; it is a widely used cluster evaluation statistics which compares the similarity (or distance) between data points inside the same clusters to data points in different clusters [24]. The similarity measure is taken to be the edge weights in our graph. We compute the silhouette coefficient s_i of a vertex x_i as follows:

$$s_i = \frac{a_i - b_i}{\max(a_i, b_i)} \tag{3.2}$$

where a_i is the average similarity between x_i and vertices in the same cluster, and b_i is the maximum average similarity between x_i and vertices in other clusters not containing x_i. In other words, b_i is the similarity of x_i to the "closest" cluster that does not include x_i. Silhouette coefficients lie in the range $[-1, 1]$. A higher coefficient means that a vertex is likely in its proper cluster. A good clustering is expected to have silhouette coefficients close to one. The silhouette of a family is defined as the average silhouette of its members.

Next we need to decide whether silhouette improvements on candidate family pairs are significant enough to join them. First define the scaled average silhouette between F_1 and F_2 before joining:

$$s = \frac{\sum_{x \in F_1} sil(x) + \sum_{y \in F_2} sil(y)}{|F_1| + |F_2|}. \tag{3.3}$$

Then let s_{12} to be the silhouette of the joined cluster $(F_1 \cup F_2)$. We define silhouette change d to be $d = s_{12} - s$.

We compute the statistical significance of silhouette change using false discovery rate (FDR) analysis. FDR gives the probability that a score higher than some threshold can be found randomly under a provided null hypothesis distribution. For a threshold t, define s_b to be the number of observed data (silhouette change in candidate pairs) with a score above t, and s_n to be the number of null data with higher scores than t. Given that the observed and null datasets have the same size, we define FDR $= s_n/s_b$. If the data set sizes are not

equal, then we scale the FDR calculation:

$$FDR = \frac{s_n}{s_b} \cdot \frac{b}{n}$$

, where b and n are the sizes of the observed and null datasets, respectively. In our calculations, the FDR is given, and the threshold t will be computed such that any pair of clusters which displays an improvement $d \geq t$ will be accepted.

3.2.1 Null Hypothesis

In order to apply FDR analysis, we need to define a null hypothesis that will display a sense of randomness in a set of clustered genes. The null hypothesis for our application is a set of joined clusters which we can compare our joined candidate pairs to. A natural choice for null hypothesis, for example, is a set of random pairs of clusters. This choice does not help our fusion test, however, as we will show in a later chapter. In our data, there are approximatively $15,000^2$ possible pairs of clusters, and and we observed that it is nearly impossible to find pairs whose silhouette will improve after fusing. This means that any silhouette improvement in candidate pairs will be accepted, which is not a very insightful result. This is due to the fact that we rely then on a null hypothesis that has no inter-cluster structure, despite maintaining fully the intra-clusters structure. Our null hypothesis should ideally display a structure that bears similarity with our data set in order to capture the conservation of the general clustering signal. The design of a null hypothesis should revolve around the clustering signals used in our fusion method, in order to determine if these signals may skew the resulting silhouette change. A good null distribution of silhouette change should also exhibit a normal distribution, so that our FDR test will only accept candidate pairs with significantly high silhouette improvements after fusion. In this section we will describe the null distributions we have examined as possible null data sets.

We look back at the criteria for which we defined candidate pairs. One of these criteria is the syntenic signal coming from two clusters with elements that share a neighbor cluster. We argued earlier that signals from gene order does not imply genes are homologous, but it may be possible that synteny is strong enough in this particular data set. Thus, we investigate clusters with a shared neighbor as a potential null data set. Another criteria that defines our candidate pairs is that the clusters span complementary species sets. This defines our second set of null distribution.

The strongest signal that can affect silhouette change is the presence of edges between clusters. Remember that, to save computational resources, edge weights are only reported in our method if they are above a certain threshold. An edge between two clusters will therefore indicate a strong similarity between some of the contained genes. Sequence similarity is the only parameter for computing silhouette, so simply having this edge may improve silhouette. So we will consider a null distribution of cluster pairs with at least one edge in between.

This signal may be too strong, however. Since we assume that the initial clustering is reasonably accurate and clusters are cohesive, an edge between points in two clusters likely means that all points between the clusters are also similar. So cluster pairs in this null data set are likely to show positive silhouette change. We may need to break up some of this structure in order to see a more appropriate distribution. One way to do this is to randomize edge weights so that the clustering signal is not as strong between clusters with edges. We decided to randomize inter-cluster edges (i.e. some edge weights will be set to 0) to create a better separation between clusters. We also randomize intra cluster weights to preserve some of the cohesion from the original clustering, but disturb it enough to achieve a more random sampling.

The resulting distributions from these null hypotheses and our final choice to use in our fusion test will be discussed in Chapter 4.

Chapter 4

Experimental Results

We applied the described method for improving gene families on a real data set of clustered genes. In this chapter we provide an small exploration of this data set to illustrate why we think the clustering may be erroneous. We describe the identified candidates clusters for fusing and the results of fusion tests.

4.1 Data

The data set used in our work contains 18 genomes from species of the *Anophelinae* subfamily. These genomes include the well studied Malaria vector, *Anopheles gambiae*, and closely related species. Each genome contains around 10,000 genes. These genes have been clustered initially into roughly 15,000 families using the OrthoDB algorithm by Waterhouse [27]. Figure 4.1 compares the number of clusters found on each genome using the initial clustering. We notice large discrepancies in these values, with some pairs of species having a difference of about 2,000 families present. This discrepancy is worrisome because gene deletions are quite rare and we expect that almost every gene from the subfamily's ancestor will have a copy in the examined genomes. So we expect that these genomes will have a roughly equal number of gene families. This is a sign that errors are present in our clustering. Another property we examined is the number of species in which a gene from a cluster is present (see Figure 4.2). As expected many families are present in all or almost all of the 18 species. What is interesting for us is the large number of clusters present inly a few species. These small clusters have possibly split from a bigger cluster, and are worth investigating.

Along with the nucleotide sequence of each gene, information on gene order are provided although it is often incomplete due to highly fragmented genome assemblies [19]. Some genomes, like *An. gambiae*, *An. dirus* and *An. funestus*, have been well assembled, while others, like *An. maculatus*, have very fragmented assemblies and thus, incomplete synteny

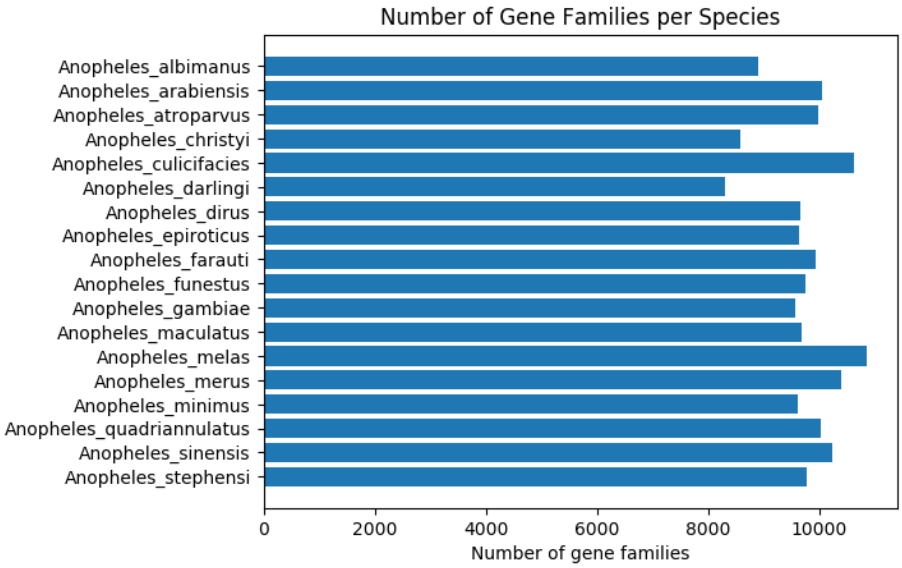

Figure 4.1: Number of clusters present in each genome

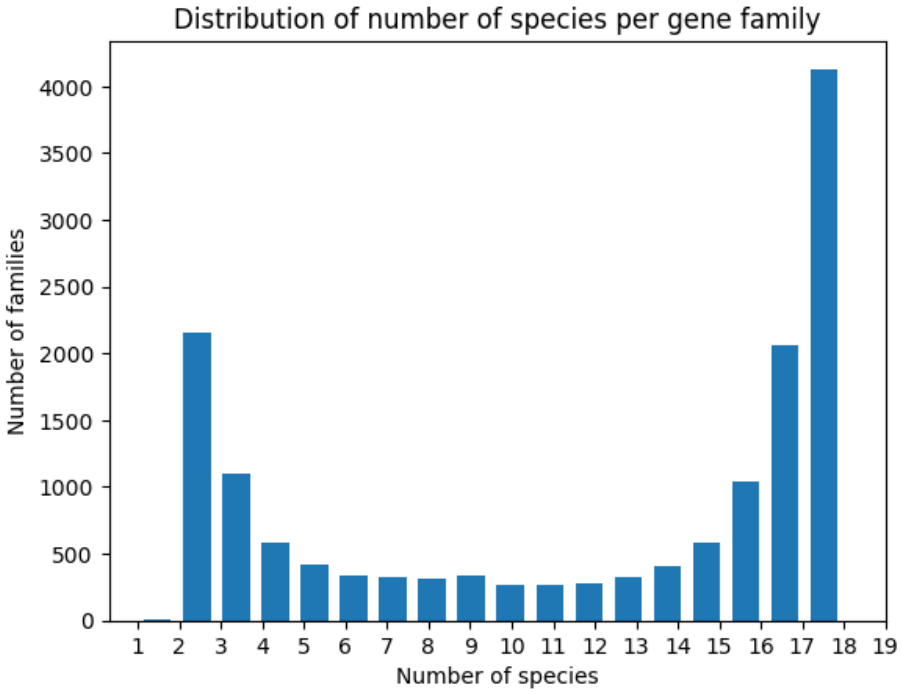

Figure 4.2: Distribution of number of species spanned by each cluster

information. Gene sequences in poorly assembled genomes also may not be entirely accurate, and some genes may be missing entirely.

4.2 Candidates

We searched for candidate pairs (F_1, F_2) among families present in complementary species where the smaller family is in 2 to 6 species. We found 133 pairs of families that share a common neighbor. An interesting result is that a significant number of pairs exhibited the candidacy conditions on multiple genomes. For example, the pair of families `MZ22528769`, `MZ22508297` (as labeled in the given data set) satisfied the synteny condition on 2 and 7 genes, respectively, with some syntenic family F_3. This suggests that the gene order was not found by chance and there is a strong signal that the family pair should be joined.

For finding tandem duplicates we limited the minimum size of the smaller family to 5 and set the ratio of observed synteny conservation to size of the smaller family at 0.8. We found $4,968$ tandem duplicates.

Some exploration of the graph model for our candidate pairs was performed to check for some hint that we did not just find random pairs [10, Examine_pairs.ipynb]. In particular, for a pair (F_1, F_2) we investigated the number of edges inside each cluster, the edges between the pairs (i.e. has and end in F_1 and another in F_2), and the edges coming out of the pair (i.e. has an end in F_1 or F_2 and the other end in a third cluster). In our graph, gene pairs with low sequence similarity does not have an edge between them, in order to save computational resources. Therefore, the number of edges between F_1 and F_2 could hint at the similarity of genes in the two clusters. In both candidate types, we found many pairs (F_1, F_2) with a high number of edges in between, relative to the size of the families. Of these pairs, some had many inside edges and only a few edges with an end not in F_1 or F_2. This suggests that the two clusters are very similar to each other but not to any other cluster. Such pairs may be good candidates for fusing.

4.3 Null Hypothesis

As described in section 3.2.1, we considered various null hypothesis for running the FDR test. We sampled $5,000$ pairs from each chosen null hypothesis and calculated silhouette change after joining each pair. The resulting distribution are plotted in figure 4.3. As expected, random pairs rarely sees any silhouette improvement after joining since the initial clustering is reasonably accurate. A similar result is observed from cluster pairs spanning complementary species since this is not a signal for homology. The null hypothesis of cluster pairs with shared neighbors shows a significant number of positive silhouette change, but the mean is still negative. This results helps show that gene order conservation is not enough to imply homology. This null hypothesis cannot be accepted since doing so would let us accept

FDR (%)	t	Number of accepted pairs		
		Shared neighbours	Tandem duplicates	Total
1	0.264	40	371	411
2	0.163	42	478	520
3	0.116	44	538	582
4	0.103	45	561	606
5	0.098	46	568	614
6	0.086	47	588	635
7	0.077	47	600	647
8	0.062	49	633	682
9	-0.022	95	2299	2394
10	-0.035	100	2432	2532

Table 4.1: Computed FDR thresholds and number of accepted candidates.

joining pairs with very little positive silhouette change, and we decided to be stringent in terms of modifications of the existing gene families, motivated among other reasons by the accepted accuracy of the OrthoDB algorithm. A small cluster quality improvement could be brought by two genes from different clusters that are similar, possibly due to domain insertions, even though the rest of the clusters have very low similarity. A good distribution is shown by pairs with inter-cluster edges. However, running an FDR analysis using this distribution outputs too few accepted results. It turns out that most of our candidate pairs belong to this null set, so we are not testing the candidates against random pairs, which is what a null distribution should provide. This leads us to the decision to randomize inter-cluster edge weights in order to break apart the similarity between clusters. We also randomized edges inside each cluster to create a more random data set while preserving the similarity signal within the clusters.

4.4 Fusion test

Figure 4.4 illustrates the distribution of silhouette change on candidate pairs and the null distribution. Note that the silhouette change can theoretically fall between $[-2, 2]$ but only a few pairs actually fall outside the range $[-1, 1]$ so we restricted the plot limits for clarity. The distribution on candidate pairs has a mean of -0.0197 and a standard deviation of 0.1797. The null distribution has mean -0.1054 and standard deviation 0.0538. We calculated the threshold t which satisfies an FDR from 1% to 10%. Table 4.1 contains the computed thresholds and the number of candidates whose silhouette improvement passes the threshold.

Using a strict false discovery rate of 1% results in only a small portion of the $\approx 5,000$ candidate pairs to be accepted. Relaxing the FDR value lets us accept more pairs but this is due to the threshold t getting very low. Low t values means that even small silhouette

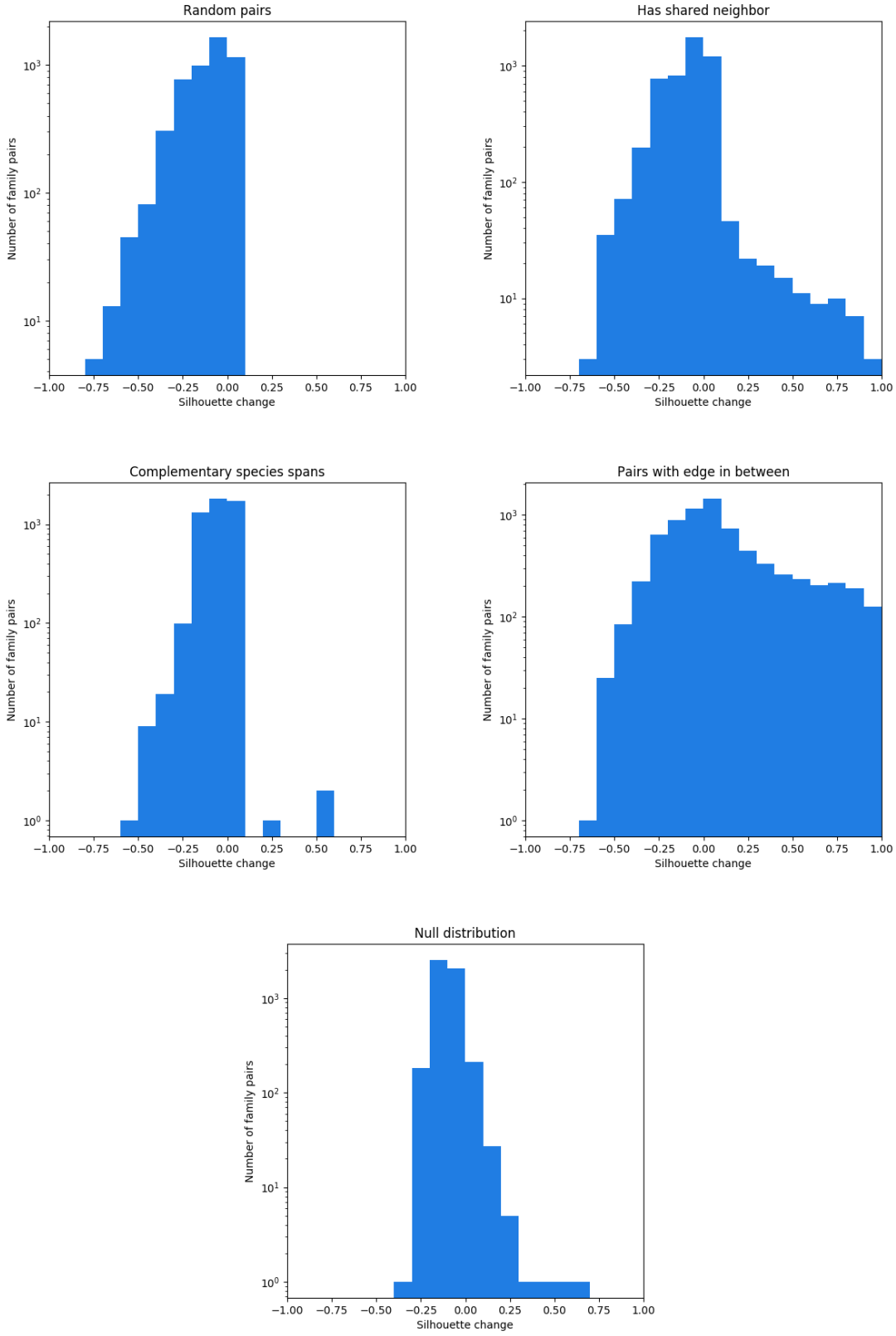

Figure 4.3: Distribution of silhouette change in null data sets. The accepted null distribution is the graph made by shuffling inter-cluster edges and intra-cluster edge weights.

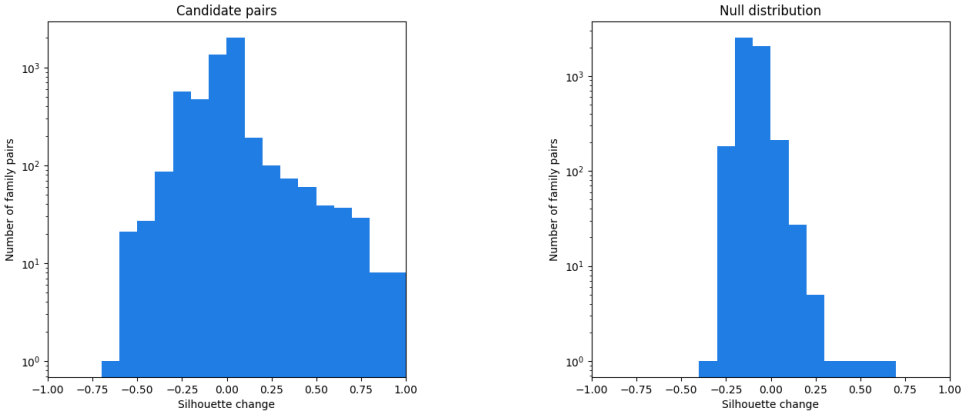

Figure 4.4: Distribution of silhouette improvements after joining family pairs

improvements are significant enough to join candidate pairs. We do not yet know which choice of FDR is optimal. However, we can observe two interesting phenomenons. First there is a significant difference between FDR 1% and 2%, and then between FDR 2% and 3%, both in terms of number of accepted candidates and silhouette improvement; second, there is a clear phase transition-like phenomenon between FDR 8% and 9%, preceded by slow steady increase of accepted candidates until the FDR is 8%. This is also coupled with the fact that at 9% we observe a negative silhouette improvement for the first time. This suggests that actually a proper FDR is likely below 9%, although the precise threshold, between 1% and 8% is still elusive.

Chapter 5

Conclusion

Clustering genes into families is an important part in the research of Malaria vectors from the *Anopheles* genus of mosquitoes. Our current work aims to improve family clustering that was produced using unsupervised sequencing and clustering methods. We designed the pipeline that identifies pairs of clusters whose union is likely to form a true gene family. This pipeline is outlined in Figure 5.1 and is available in the `Fuse_Anopheles_Families` repository [10]. The principle behind our method is to find pairs of clusters showing signs of synteny conservation, then test whether the pairs should be joined by examining sequence similarity based silhouette and evaluating the statistical significance of any silhouette improvement after joining the cluster pair.

The pipeline was applied on a clustering of *Anopheles* mosquito genomes. Out of nearly four thousand candidate pairs, only 411 has a 1% false discovery rate. Relaxing the FDR requirement allows us to accept more candidates, but it is still unknown what FDR is optimal. Some work to follow is to verify whether joining the pairs of clusters found by our method truly forms gene families by aligning genes belonging to the joined pairs of clusters. We could manually examine candidate pairs that passed the FDR threshold and those that are just below the threshold, and look at multiple sequence alignments to see if the fused candidates truly form a single family. Checking gene functions is another way to validate our results. This verification may also help in identifying optimal values for parameters at which we run our tests. It may also be worthwhile to explore other statistical tools for deciding when silhouette improvements are significant enough. We could also compare our results with GenFamClust, which uses a combination of synteny conservation and sequence similarity but in a way that is different to our method.

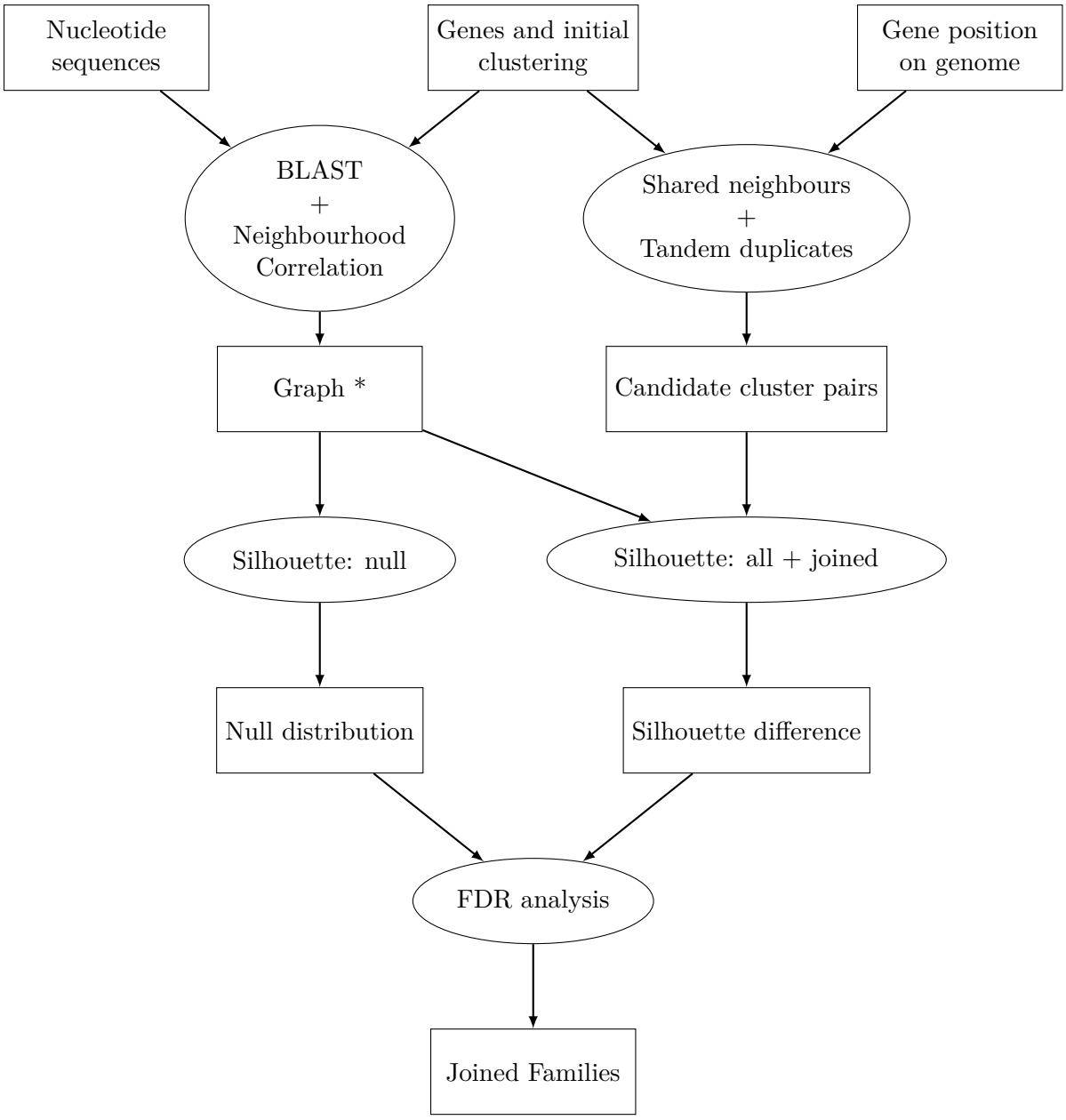

Figure 5.1: Flow chart of pipeline. Items in ovals are scripts or programs, and rectangles are data. In depth description of each script and input formats can be found in the README file of the `Fuse_Anopheles_Families` repository [10]. Note that the sequence similarity module, BLAST + Neighbourhood Correlation, may be replaced with other similarity measures. *The graph refers to the clustering graph described in Figure 1.3, presented as a list of edges and edge weights .

Bibliography

[1] Genetics and health. https://ghr.nlm.nih.gov/primer/mutationsanddisorders/possiblemutations. Accessed: 2017-11-16.

[2] NCBI BLAST. https://blast.ncbi.nlm.nih.gov/Blast.cgi?PROGRAM=blastn&PAGE_TYPE=BlastSearch&LINK_LOC=blasthome. Accessed:2017-12-14.

[3] Charu C Aggarwal and Chandan K Reddy. *Data clustering: algorithms and applications*. CRC press, 2013.

[4] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. Molecular biology of the cell. new york: Garland science; 2002. *Classic textbook now in its 5th Edition*, 2002.

[5] Raja H. Ali, Sayyed A. Muhammad, and Lars Arvestad. GenFamClust: an accurate, synteny-aware and reliable homology inference algorithm. *BMC evolutionary biology*, 16(1):120, June 2016.

[6] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403 – 410, 1990.

[7] Mo Chen. Math Works, kmeans clustering. https://www.mathworks.com/matlabcentral/fileexchange/24616-kmeans-clustering?requestedDomain=www.mathworks.com. Accessed: 2017-11-16.

[8] James F. Denton, Jose Lugo-Martinez, Abraham E. Tucker, Daniel R. Schrider, Wesley C. Warren, and Matthew W. Hahn. Extensive error in the number of genes inferred from draft genome assemblies. *PLOS Computational Biology*, 10(12):1–9, 12 2014.

[9] S VAN Dongen. *Graph clustering by flow simulation*. PhD thesis, University of Utrecht, 2000.

[10] Mark Forteza. Fusion of Anopheles gene families. https://github.com/mfortz/Fuse_Anopheles_Families, 2017.

[11] Christian Frech and Nansheng Chen. Genome-wide comparative gene family classification. *PLoS one*, 5(10):e13409, 2010.

[12] Zheng Fu, Xin Chen, Vladimir Vacic, Peng Nan, Yang Zhong, and Tao Jiang. Msoar: a high-throughput ortholog assignment system based on genome rearrangement. *Journal of Computational Biology*, 14(9):1160–1175, 2007.

[13] M Stanley Fujimoto, Anton Suvorov, Nicholas O Jensen, Mark J Clement, and Seth M Bybee. Detecting false positive sequence homology: a machine learning approach. *BMC bioinformatics*, 17(1):101, 2016.

[14] D Graur. Fundamentals of molecular evolution. *Massachusetts: Sinauer Associates*, 1991.

[15] Christian Hennig, Marina Meila, Fionn Murtagh, and Roberto Rocci. *Handbook of cluster analysis*. CRC Press, 2015.

[16] Federico G Hoffmann, Juan C Opazo, and Jay F Storz. Gene cooption and convergent evolution of oxygen transport hemoglobins in jawed and jawless vertebrates. *Proceedings of the National Academy of Sciences*, 107(32):14274–14279, 2010.

[17] Richard SP Horler and Carin K Vanderpool. Homologs of the small rna sgrs are broadly distributed in enteric bacteria but have diverged in size and sequence. *Nucleic acids research*, 37(16):5465–5476, 2009.

[18] Li Li, Christian J Stoeckert, and David S Roos. Orthomcl: identification of ortholog groups for eukaryotic genomes. *Genome research*, 13(9):2178–2189, 2003.

[19] Daniel E. Neafsey, Robert M. Waterhouse, Mohammad R. Abai, Sergey S. Aganezov, Max A. Alekseyev, James E. Allen, James Amon, Bruno ArcÃă, Peter Arensburger, Gleb Artemov, Lauren A. Assour, Hamidreza Basseri, Aaron Berlin, Bruce W. Birren, Stephanie A. Blandin, Andrew I. Brockman, Thomas R. Burkot, Austin Burt, Clara S. Chan, Cedric Chauve, Joanna C. Chiu, Mikkel Christensen, Carlo Costantini, Victoria L.M. Davidson, Elena Deligianni, Tania Dottorini, Vicky Dritsou, Stacey B. Gabriel, Wamdaogo M. Guelbeogo, Andrew B. Hall, Mira V. Han, Thaung Hlaing, Daniel S.T. Hughes, Adam M. Jenkins, Xiaofang Jiang, Irwin Jungreis, Evdoxia G. Kakani, Maryam Kamali, Petri Kemppainen, Ryan C. Kennedy, Ioannis K. Kirmitzoglou, Lizette L. Koekemoer, Njoroge Laban, Nicholas Langridge, Mara K.N. Lawniczak, Manolis Lirakis, Neil F. Lobo, Ernesto Lowy, Robert M. MacCallum, Chunhong Mao, Gareth Maslen, Charles Mbogo, Jenny McCarthy, Kristin Michel, Sara N. Mitchell, Wendy Moore, Katherine A. Murphy, Anastasia N. Naumenko, Tony Nolan, Eva M. Novoa, Samantha O'Loughlin, Chioma Oringanje, Mohammad A. Oshaghi, Nazzy Pakpour, Philippos A. Papathanos, Ashley N. Peery, Michael Povelones, Anil Prakash, David P. Price, Ashok Rajaraman, Lisa J. Reimer, David C. Rinker, Antonis Rokas, Tanya L. Russell, N'Fale Sagnon, Maria V. Sharakhova, Terrance Shea, Felipe A. SimÃčo, Frederic Simard, Michel A. Slotman, Pradya Somboon, Vladimir Stegniy, Claudio J. Struchiner, Gregg W.C. Thomas, Marta Tojo, Pantelis Topalis, JosÃľ M.C. Tubio, Maria F. Unger, John Vontas, Catherine Walton, Craig S. Wilding, Judith H. Willis, Yi-Chieh Wu, Guiyun Yan, Evgeny M. Zdobnov, Xiaofan Zhou, Flaminia Catteruccia, George K. Christophides, Frank H. Collins, Robert S. Cornman, Andrea Crisanti, Martin J. Donnelly, Scott J. Emrich, Michael C. Fontaine, William Gelbart, Matthew W. Hahn, Immo A. Hansen, Paul I. Howell, Fotis C. Kafatos, Manolis Kellis, Daniel Lawson, Christos Louis, Shirley Luckhart, Marc A.T. Muskavitch, JosÃľ M. Ribeiro, Michael A. Riehle, Igor V. Sharakhov, Zhijian Tu, Laurence J. Zwiebel, and Nora J. Besansky. Highly evolvable malaria vectors: the genomes of 16 Anopheles mosquitoes. *Science (New York, N.Y.)*, 347(6217):1258522, January 2015.

[20] William Stafford Noble. How does multiple testing correction work? *Nature biotechnology*, 27(12):1135–1137, December 2009.

[21] Kevin P O'brien, Maido Remm, and Erik LL Sonnhammer. Inparanoid: a comprehensive database of eukaryotic orthologs. *Nucleic acids research*, 33(suppl_1):D476–D480, 2005.

[22] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[23] Nan Song, Jacob M Joseph, George B Davis, and Dannie Durand. Sequence similarity network reveals common ancestry of multidomain proteins. *PLoS computational biology*, 4(5):e1000063, 2008.

[24] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to data mining*. Pearson Education India, 2006.

[25] Fredj Tekaia. Inferring Orthologs: Open Questions and Perspectives. *Genomics Insights*, 9:17–28, February 2016.

[26] Stijn Marinus Van Dongen. *Graph clustering by flow simulation*. PhD thesis, 2001.

[27] Robert M Waterhouse, Fredrik Tegenfeldt, Jia Li, Evgeny M Zdobnov, and Evgenia V Kriventseva. Orthodb: a hierarchical catalog of animal, fungal and bacterial orthologs. *Nucleic acids research*, 41(D1):D358–D365, 2012.

[28] Mohammed J Zaki, Wagner Meira Jr, and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.