CSC 550 – Algorithms in Bioinformatics

Term Paper: Phylogenic Tree Reconstruction

Steven Eiselen (seiselen24)

FINAL VERSION 8/13/19

Contents

Section 1: Problem Motivation from Bioinformatics

Introduction to Phylogeny and Phylogenic Trees

What does mapping humanity's expansion across Earth, tracking infectious diseases such as the SARS virus, and settling the debate on whether the Giant Panda is a bear have in common? The answer is Phylogeny, a branch of Biology involved with studying both the evolutionary history of individual species and relationships between groups of species; especially in terms of common ancestry thereof. The primary means of representing and visualizing the 'evolutionary hierarchy' of such relationships is with a Phylogenic Tree, also known as an Evolutionary Tree.

A Phylogenic Tree describes the evolutionary relationships between a set of different species; such that each internal node represents a speciation event wherein one species diverges into two species, each leaf node represents a species with no divergent successors, and the root represents the ancestor common to all species within the tree. As a whole then: the tree encompasses a hierarchy of ancestral relationships caused by speciation events with respect to both the common ancestor between all species therein, as well as all subtree 'local phylogenies' therein.

Examples of Phylogenic Trees and Motivations Thereof

Tree of Life: The most well-known example of a Phylogenic Tree is the 'Tree of Life'. It is, pardon the pun, the "Mother" of all Phylogenic Trees, as it seeks to express the phylogeny for every species which existed on Earth. Research continues on progress [REF] towards a more complete phylogeny for as many species as possible, for which methods are needed to compare the similarity between them. A further motivation is use of this tree to improve the ability to classify and relate newly discovered species.

Human Ancestry: DNA-based Phylogenic Trees (which we study in this paper) were crucial towards defending the 'Out of Africa' theory of human migration across the world, as it was determined that African peoples contain the most amount of genetic diversity with respect to each other, while being 'mapped' further up the phylogeny of human variations; both of which imply Africa as the origin of the Homo Sapiens. A corollary benefit of this finding was production of a 'roadmap' showing the entire migration, which further defended migration theories: from post-Ice Age expansion into Northern Europe, to migration of Asian populations across the 'Beringia Land Bridge' into the Americas.

Giant Pandas and SARS Virus: Dr. Phillip Compeau (hereafter referred to as 'Compeau'), who is the co-author for one of our primary sources 'Algorithms in Bioinformatics', mentions two more examples and their motivations for the use of Phylogenic Trees. The first involved settling a long-time debate on whether the Giant Panda is a marsupial or bear; the uncertainty due to them having anatomical similarities to both groups. This is the 'dead end' encountered with classical morphological-based methods for constructing phylogenies and rationale for the use of genetic-based methods, as we discuss below. Indeed, genetic phylogeny resolved the debate by showing greater genetic similarity to bears. The second example, as the theme of their chapter on Phylogenic Tree Reconstruction, was utilizing the phylogeny formed from samples of the SARS virus from infected subjects to correlate with and add additional data to the timeline and geographic roadmap for how and when the disease was spread. This motivation trivially proved crucial towards both tracking and combatting SARS and future epidemics of concern.

Phylogenic Tree Datatypes

How are Phylogenic Trees Built? First, it depends on the data used. Our other primary source was a video lecture series by Dr. Kevin Yuk-Lap Yip (hereafter referred to as 'Yip') at the Chinese University of Hong Kong. When introducing phylogenic trees, Yip offers three 'progressions' leading to the datatype which our algorithms will use for Phylogenic Tree Reconstruction:

1. **Morphologic:** "How are the *observable features* of different species similar and/or different?" This is the 'classical method' (i.e. as used back in Charles Darwin's day when visual observation was the only classification method available). Compeau notes that in the mid-20th century, researchers advanced this concept to produce 'Character Tables'; which are matrices whose rows correspond to species and whose columns correspond to features for which each row species therein has its respective information entered (e.g. 'Has Wings', 'Breathes Air', 'Warm Blooded', etc.) We note an interesting irony in that many machine learning classification systems working with images of animals utilize a similar idea. And as with the challenges faced by ML image classifiers: the downsides to this method correspond to three speciation events:
    a. Convergence, wherein two or more distant species develop similar traits independently from one another. Two notable examples include the development of wing structures in Birds, Bats, and Pterosaurs; and the development of eyes in Vertebrates and Cephalopods.
    b. Violations of Dollo's Principle, which states that evolution does not redevelop the same features for species whose ancestors discarded them; for which Compeau provides such a contradictory example of cyclic reemergence of insect wings.
    c. Uncertain similarity, such as the case of the Giant Panda.

   Biologic/Chemical: Determines similarity via chemical and biological processes such as respiration, metabolism, neurology, etc. Effectively speaking: another set of features.

   Genetic: Determines similarity via [multiple] sequence alignment of samples of species' genomes. This is the data type utilized by the methods discussed in this paper. Effectively, and thus 'most powerfully' as evidenced by the Panda Bear and human expansion examples: we can compare and relate species directly via their "source code" versus the features that such code generates.

Complexity of Genetic Data and Solutions Thereof: We observed in class that while the most accurate and 'ideal' data would be the entire genomes of the subjects composing the input, this is unfeasible due to the massive size thereof. This is especially given that the sequence-based method works with each of the input sequences' characters, as does the data structure used as input for the distance-based methods. To address this, Yip suggests that only some genes could be used: particularly those that are both 'essential' to the functioning of the species and for which evolution occurs very slowly. Towards this, he informs that Ribosomal RNA, used by an organism for protein synthesis, is a popular choice.

Phylogenic Tree Reconstruction

Introduction: We discuss the general process of reconstructing a Phylogenic Tree from genetic data. We are given a set of equally sized, multiple aligned DNA sequences, each representing a particular species and/or subject thereof. From these 'known' species which will compose the leaf nodes for their phylogenic tree, we work towards 'reconstructing' the unobserved and/or unobservable ancestors as internal nodes which relate each of the leaf nodes to each other as accurately as possible. What does it mean to 'reconstruct unobserved ancestor nodes'?

'Principle of Inferred Phylogeny': The Phylogenic Trees produced by reconstruction do not presume to be the actual Phylogeny of the given species/sequences; but an inference thereof (based on the particular method and input). That is, the path from a leaf to the root provides an inference for how a species evolved and from which unobserved species thereof, based on its similarity thus relationships with the other leaf level species. The path and tree which produced it could be 100% accurate to its correspondence in nature, and this is certainly the ideal scenario we wish to converge towards. But as we discuss in the solution methods sections, the reconstructed trees may encounter errors.

To provide some supporting intuition towards determining some assumptions i.e. heuristics we can make to most accurately reconstruct phylogenies, Yip offers three initial assumptions and we add a fourth from Compeau for which we will use as lemmas and expand upon within our problem definitions and solution methods. They are as follows:

- All species share exactly one common ancestor
- All species mutated somewhere along the phylogeny from this common ancestor
- Mutations are 'rare' events, such that many should not exist between closely related species.
- Internal nodes imply speciation events wherein one species diverges into two.

From a general definition of our goal, we can now produce a computational problem definition.

Section 2: Computational Problem Definition

Overview and Problem Definition

Introduction: In this section, we discuss the computational problem definition common to both the sequence-based and distance-based solution methods. More specific augmentations to this definition will be detailed in their respective sections / subsections for each of the methods. We first express the general problem definition in a sentence or two before further detailing it immediately thereafter.

General Problem Definition: We are given a set of $\boldsymbol{n}$ equally sized, multiple aligned DNA/RNA sequences, each representing a particular species. With one exception discussed in the Neighbor-Join Algorithm: we have no other information on how each species relates to one another within their real-world phylogeny. From this input, perform Phylogenic Tree Reconstruction such that the tree produced is as accurate an inference to the real phylogeny as possible. The tree should then encompass the following information:

- ***Speciation events*** (i.e. where speciation occurred, from which parent, and the resulting siblings)
- ***The order thereof*** (i.e. how, when and from which ancestral path speciation events occurred)
- ***The distances thereof*** (as applicable, and i.e. the 'time' between two separate speciation events)

Phylogenic Tree Data Structure

The Phylogenic Tree Data Representation is a Binary Tree DAGs whose properties are as follows:

- **Root Node:** Most common ancestor ∀ species within the tree (*as applicable).
- **Leaf Node:** Observed species, i.e. part of the input set of species/sequences.
- **Internal Node:** Unobserved ancestor species. Not known, will be inferred and created
- **Edges:** Indicate direct ancestry from predecessor to successor specie; i.e. an internal node has two edges directed at its child nodes, which represents the relationship between a parent species and the two species which diverged directly from it, respectively.

- **Edge Weights:** Used for Phylogenic Tree Reconstruction methods and to infer distance

***Unrooted Trees:** Some of the methods are unable to produce a rooted tree, indicating that the most common ancestor is either unknown or unclear. Such is typically due to either unobserved/unobservable data as discussed in the Principle of Inferred Phylogeny', and/or the complexity and accuracy of the genome sequences as discussed in the data definition (each aforementioned).

Problem Complexity and Solution Methods Thereof

The problem definition implies that Phylogenic Tree Reconstruction is an optimization problem due to desiring an output 'as accurate as possible'. Consequently, producing a 'best possible tree' is a hard problem, more specifically: NP-Complete. As Yip notes, this is chiefly due to the fact that the space of total permutations for all possible tree topologies is exponential. However, he also provides a basic discussion of core problem-solving methods for which the methods discussed in this paper derive, as follows:

1) **Define Easier Version Of Problem:** Make simplification assumptions and approximate solutions that could be the optimal if not 'close enough' thereof. Such techniques have been studied in class, such as the family of approximate matching i.e. sequence alignment algorithms. In fact, not only are their problem complexities similar to Phylogenic Tree Reconstruction, but they can also produce an optimal output or an otherwise 'good fit' if such exists.
2) **Design Smarter Algorithms/Data:** If possible, engineer algorithms and data structures to avoid 'redundant' or unnecessary calculations and utilize useful representations. We have also seen this technique in class with sequence alignment Dynamic Programming; and with the accessory data structures and methods of Karkainnan-Sanders Algorithm.
3) **Delegate (i.e. Utilize) Heuristics:** If a 'rule-of-thumb' has been shown to deliver consistently good results, use it. A good is the classic A* pathfinding algorithm, which utilizes a heuristic to measure distances between nodes and a goal node to determine which node to traverse into; of which two common heuristics are Manhattan and Euclidean distance.

Specifically, we will discuss two solution method types and the techniques thereof. The first of these types works with the sequence data itself, for which we discuss the Small and Large Parsimony Problems. The second type works with the Distance Matrix data structure aforementioned, and features three different algorithms: Additive Phylogeny, UPGMA, and Neighbor-Join.

Section 3: Sequence-Based Solution Methods

Introduction

In this section, we discuss a single method for performing Phylogenic Tree Reconstruction via working with the DNA sequences themselves. Doing so allows this method to support a feature which Distance-Based methods cannot: infer ancestor sequences in addition to the tree which composes them. We start by introducing an extension to the Character Tables discussed in Section 1, then the Tree Parsimony Score and Max Parsimony Problem, followed by discussing the Small and Large Parsimony Algorithms.

DNA Sequence Character Tables

Recap of Morphological Character Tables: In Section 1, we introduced the technique of constructing a 'Character Table' wherein each row corresponds to a species and each column corresponds to some attribute, such that the column data for each row corresponds to the attributes of the species that row represents. From this table, one could evaluate a basic 'distance' score to determine the similarity between two species as the number of columns for which their data does not match.

DNA Sequence Character Tables: Compeau takes this idea further to define a DNA Character Table wherein each row corresponds to a species and each column corresponds to an individual nucleotide, such that the column data for each row corresponds to the DNA sequence for its respective species. Thus, we can express the multiple sequence alignment of the input as a matrix of dimension $n \times m$ such that m is the common sequence length. That is, we effectively 'stack' the multiple alignment such that each sequence is a row and each of their individual characters the column values.

Distance Between Sequences: As with the Morphological Character Tables, the 'distance' between sequences corresponds to their pairwise Hamming Distances given the multiple alignment. Indeed - this is how the Distance Matrix could be formed. Lastly, we note that Yip refers to these pairwise distances as corresponding to the number of mutations between them.

Tree Parsimony Scores and the Max Parsimony Problem

Parsimony Score: As discussed in the introduction, the Sequence-Based method allows for inferring the sequences of unobserved ancestors. Compeau invites the question of how determine the 'goodness' of these inferences throughout the tree. Towards this, he defines a 'Parsimony Score' for the tree as the sum of Hamming Distances along each edge therein, which will lead directly into the Small Parsimony Problem discussed in the following subsection.

Max Parsimony Problem: Yip provides further intuition with a small 'lemma' problem called 'Max Parsimony' which leads into the Large Parsimony Problem discussed further below. The Max Parsimony Problem is as follows: "Find a phylogenic tree whose leaf sequences match the input sequences and whose generated ancestor sequences contain the smallest possible number of mutations".

Assumptions from Max Parsimony: Yip expresses further thoughts from the Max Parsimony Problem which provide some key assumptions regarding the frequency of mutations. First, that a tree is more likely to accurately infer reality if it encompasses a minimum number of mutations; for which the rationale derives from another assumption we made in Section 1 wherein mutations are said to be rare events. He then invokes Occam's Razor to express that if the "simplest explanation is most likely the correct one", then then given two sequences of significant distance: it is more likely that they are distant relatives from one another than it is that they are close relatives sharing such a significant number of mutations. To put it differently (and in morphological terms): Given two species 'Eagle' and 'Squid', it is more likely that the mutations which effect their genetic differences are due to being [very] distant relatives versus them sharing some parent species fit for a Science Fiction flick. Inversely, it is just as likely that the minimum distance between 'Wolf' and 'Dog' infers that they are 'sibling' species of a common parent than it is to force 'Wolf' to be the sibling of 'Dolphin' such that their parent would also be fit for Science Fiction.

The Small Parsimony Problem

Problem Definition - Version A: Given a rooted binary tree topology T with each leaf labeled by a string of length m, find the most parsimonious labelling of the internal nodes therein. That is: produce an [inferred] labeling of similar size for all internal nodes of T based on their child labels, such that the Parsimony Score of T is minimum; and note that this problem works with an already created tree.

Simplification-Via-Reduction: Compeau observes that this problem seems difficult considering sequence length and suggests a modification to the Problem Definition such that we only consider the sequences one character at a time. That is: treat the DNA Character Table columns as disjoint from each other and solve the problem for each column independently. Yip provides intuition for how we can utilize the same simplification for our final algorithm, which we discuss later on. Thus…

Problem Definition - Version B: Given a rooted binary tree topology T with each leaf labeled by a character, find the most parsimonious labelling of the internal nodes therein. That is: produce an [inferred] labeling of characters for all internal nodes of T based on their child labels, such that the Parsimony Score of T is minimum.

Solving Small Parsimony - Simple Approach to Fitch's Algorithm

Overview: Yip starts with a simple form of Fitch's Algorithm before advancing into an extended version analogous to the one discussed by Compeau; for which we note the only major fundamental difference between the two comes down to a selection heuristic which can produce less unnecessary mutations in the ancestor nodes. We first start with some ideas that are important both versions of this algorithm:

- **Idea 1:** For a given internal node, if both children have the same character (a.k.a. 'agree' on a character), the parent should be assigned this character; as such implies no mutation event occurred for this particular character (recall that we're dealing with columns independently).
- **Idea 2:** For a given internal node, if both children have different characters (a.k.a. 'disagree' on a character), the parent should be assigned one of the two child characters versus some other character; as such would encompass mutation for one child versus both, which is more likely.
- **Idea 3:** An internal node is in 'no rush' to have a character assigned to it until at least its parent has had a character assigned; but should select preference[s] in the meantime. That is, it can determine its preferred character[s] first, then wait to see which character its parent chose, then make a decision consistent with Ideas 1 and 2. Specifically: that if the node's parent was assigned a character that is on the node's preference list - the node should also be assigned that character; else otherwise select from its preference list. This policy ensures that the scenario of two mutations is limited, as an ancestor either selects its parent assignment which avoids a mutation from the parent to the node, else selects from its children which avoids two mutations.
- **Idea 4:** Idea 3 is extremely important and in fact one of the keys to the solution, as it sets up a condition for a recursive divide-and-conquer algorithm involving an upwards and downwards sweep suitable for a corresponding application of Dynamic Programming as we present below.

Algorithm:

Let $\boldsymbol{M}$ *= Matrix of dimension* $[n \times 1]$ *representing a certain column from DNA Character Table*
Let $\boldsymbol{S_v}$ *= Preference Set of node* $\boldsymbol{v}$ *as list of one or more characters from alphabet i.e.* $\{A.C.G.T\}$
Let $\boldsymbol{C_i}$ *= Character assigned to internal i.e. ancestor nodes*
Let $\boldsymbol{C_p}$ *= Character assigned to the parent of a node*

Let C_{root} = root node
Let iterations in supersteps 2 and 3 = breadth-first traversal in accordance to a Binary Tree.

FitchSmallParsimonyBasic(M)

1. *For each leaf node v, add its corresponding sequence character $M[v]$ to S_v*
2. *Foreach internal node v with children p and q from leaf to root level:*
    a. *If set intersection of S_p and S_q is empty: Append set union thereof to S_v. That is, if both children have no preferences in common, add all contents of their lists.*
    b. *Else: Append intersection of S_p and S_q to S_v. That is, they have 1 or more characters in common, append all of these intersecting characters.*
3. *Randomly assign a member of S_{root} to C_{root}*
4. *Foreach internal node v sans the root with children p and q from root to leaf level:*
    a. *If $C_p \in S_v$, $C_v = C_p$. That is, if the selected character of the parent is within the node's preference list (i.e. the node 'agrees with' the parent), assign that character as also the node's selection.*
    b. *Else $C_v = random(S_i)$. That is, otherwise randomly select a character from the node's preference list*

Inductive Proof for Fitch's Algorithm: Yip discusses a proof for which we provide an abridged version as it provides insight towards the expanded algorithm. Foremost at the basis: correctness of the leaves is trivial - they never change from their representation in the specific column of the Character Table. From the leaves to their parent is also trivial, as either their sibling thus parent shares the same character (for which there is no mutation) xor there is exactly one mutation vis-a-vis Idea 3 above. And due to the recursive, incremental upwards and downwards sweep which is based on Idea 3, the same will hold for parents of other internal nodes. Put differently: The divide-and-conquer nature of this algorithm and upwards/downwards traversal thereof is such that subtrees within the main tree are 'locally optimal', which fundamentally follows for the greater composition/topology.

Solving Small Parsimony - Extended Approach to Fitch's Algorithm

Overview: The main extension made to Fitch's Algorithm involves utilizing a 'reverse voting system' for preferences versus simply appending them to some preference list. Furthermore, the structure of the preference list for each node now changes to being length of the cardinality of the alphabet, as votes will be incremented therein. Thus, in effect, these preference lists represent 'local parsimony' for nodes given their subtrees, as is similarly the case for the basic version of the algorithm. We will want to select characters for which the local parsimony i.e. number of mutations is minimal, hence the idea of a 'reverse voting system' where the least amount of votes are preferred.

Algorithm:

Let M = Matrix of dimension $[n \times 1]$ representing a certain column from DNA Character Table
Let $S_v[c]$ = Value of votes for character c in Preference Set of node v
Let C_i = Character assigned to internal i.e. ancestor nodes
Let C_p = Character assigned to the parent of a node
Let C_{root} = root node
Let iterations in supersteps 2 and 3 = breadth-first traversal in accordance to a Binary Tree.

FitchSmallParsimonyExtended($\boldsymbol{M}$)

1. *For each leaf node $\boldsymbol{v}$, $\boldsymbol{S_v[M[v]] = 0}$, $\forall\, \boldsymbol{x} != \boldsymbol{M[v]}\ \{\ \boldsymbol{S_v[x]}\ = \infty$. That is, set the value of its preference list corresponding to its sequence character to zero, and infinity for all other elements therein.*
2. *Foreach internal node $\boldsymbol{v}$ with children $\boldsymbol{p}$ and $\boldsymbol{q}$ from leaf to root level, do:*
   a. $\boldsymbol{S_v[k] = min\{\, s_p[c] + \big((c == k)?\ 0 :\ 1\big)\}\ \forall\, c \in \Sigma\ +}$
      $\boldsymbol{min\{\, s_q[c] + \big((c == k)?\ 0 :\ 1\big)\}\forall\, c \in \Sigma}$
3. *Assign the lowest voted character of $\boldsymbol{S_{root}}$ to $\boldsymbol{C_{root}}$*
4. *Foreach internal node $\boldsymbol{v}$ sans the root with children $\boldsymbol{p}$ and $\boldsymbol{q}$ from root to leaf level:*
   a. *If $\boldsymbol{C_p \in S_v}$, $\boldsymbol{C_v = C_p}$. That is, if the selected character of the parent is within the node's preference list (i.e. the node 'agrees with' the parent), assign that character as also the node's selection.*
   b. *Else $\boldsymbol{C_v = min(S_v)}$. That is, otherwise select the min score from the node's preference list*
5. *Lastly, the min parsimony score for the resulting tree will be the min value of $\boldsymbol{S_{root}}$*

Solving Small Parsimony for Entire Sequence

Yip provides a very intuitive extension for expanding Small Parsimony to account for and produce the parsimony score given reconstructions for the entire sequence versus just one character. We first make an additional assumption that each character in a sequence mutates independently. This nicely aligns with our current approach of considering columns independently. We then apply Fitch's Algorithm for each column in the Character Table as to record the scores for substrings versus characters such that the preferences are expressed as 'mergeable preferred substrings'. Effectively, we are working with the intersections of similarity between character strings similar to the simple method for Fitch's Algorithm.

Large Parsimony Problem

Problem Definition: Given a set of sequences $\boldsymbol{S}$, find the rooted tree topology $\boldsymbol{T}$ with each leaf labeled by a string of length $\boldsymbol{m}$ that has a minimum parsimony score. That is: produce a tree with an [inferred] labeling of similar size for all internal nodes minimizing parsimony score among all possible topologies.

Problem Complexity: The phrase 'all possible tree topologies' has been encountered before, alongside the corresponding fact that there are an exponential number of such trees. This makes the Large Parsimony Problem NP-Complete.

Solution Method 1 - Linear Programming / Machine Learning Approach: Yip discusses the 'brute force strategy' of running Small Parsimony on different permutations of tree topologies, as to compare attempts with one another to find some best encountered model. This idea, alongside the 'mechanics' of the problem which engenders both optimization and a graph representation with edge weights; compels us to suggest possible utilization of either Linear Programming and/or Machine Learning techniques towards finding such a best possible model.

Solution Method 2 - Greedy Heuristic via Nearest-Neighbor Interchange: Compeau suggests a greedy iterative approach which also has motivations in an 'informed' brute force approach. This method effectively 'permutes' through minor iterative adjustments of the structure of the tree until finding some optimal topography thereof, though we reserve further details in brevity.

Section 4: Distance-Based Solution Methods

Introduction

In this section, we discuss three methods for performing Phylogenic Tree Reconstruction based on the distances between all pairs of the input sequences. Each of these methods utilize a preprocessed data structure called a Distance Matrix which provides this information, and reconstructs the tree in similar ways via joining exactly two nodes to a single shared parent until all nodes and/or subtrees have been joined. Yip introduces two main advantages for using Distance-Based methods: working with a simplified representation of the original sequence data which provides enough useful information (while being subject to an information loss consequence); and producing trees in a relatively more express manner than Sequence-Based methods if the structure of the inferred tree is desired more than the sequences which compose the internal nodes therein (i.e. the aforementioned consequence).

To the former: the Distance Matrix expresses the alignment distance between all pairs of the sequences with a single $\boldsymbol{n} \times \boldsymbol{m}$ matrix of integer values, versus having to keep and iterate through $\boldsymbol{n}$ sequences of what are often massive length; however this means that we assume the Distance Matrix as the problem's only input; thus cannot recover or observe the original sequences which composed it. To the latter, because the sequence data is unavailable - inference cannot be done on what the ancestor sequences might have been, at least within the scope of the problem definition (WLOG). However, as implied above, the lack of consideration for the actual sequences could encompass performing substantially less computational work if all that is desired is the Phylogenic Tree itself.

Distance Matrix Data Structure

A Distance Matrix is a $\boldsymbol{n} \times \boldsymbol{m}$ matrix formed from performing multiple sequence alignment upon the initial input sequences. Its values contain the distances between each pair of sequences, defined as the number of different characters between them. Thus, such a matrix $\boldsymbol{M}$ features the following attributes:

- $\boldsymbol{M}[\boldsymbol{p}, \boldsymbol{q}] \rightarrow$ Alignment of sequences $\boldsymbol{p}$ and $\boldsymbol{q}$
- $\boldsymbol{M}[\boldsymbol{p}, \boldsymbol{q}] = \boldsymbol{M}[\boldsymbol{q}, \boldsymbol{p}] \rightarrow$ We assume matrix symmetry (WLOG)
- $\boldsymbol{M}[\boldsymbol{x}, \boldsymbol{x}] \rightarrow$ Alignment of a sequence against itself, trivially zero

Input Pre-Processing Assumption: As the input to the problem definition for these methods is the Distance Matrix, we assume that the work required to perform multiple sequence alignment and produce the Distance Matrix itself has already been performed ahead of time, thus immediately available for use.

'Fitting Phylogenic Trees to Distance Matrices'

Definition and Intuition: 'Tree Fitting' is the process of assigning edge weights to the tree $\boldsymbol{T}$ based on information from $\boldsymbol{M}$ such that the path from one node $\boldsymbol{p}$ to any other node $\boldsymbol{q}$ within the Phylogenic Tree corresponds to their Distance Matrix value $\boldsymbol{M}[\boldsymbol{p}, \boldsymbol{q}]$; especially if that path goes through several ancestor nodes i.e. subtrees within $\boldsymbol{T}$. Put differently: this process seeks to construct $\boldsymbol{T}$ based on and consistent with $\boldsymbol{M}$ such that $\boldsymbol{T}$ composes an accurate a phylogenic inference thereof as possible; which is, effectively, the problem definition!

Optimal Tree Fit Theorem: An obvious question arises - Can a tree be perfectly fit given any Distance Matrix? The answer depends on whether or not $\boldsymbol{M}$ is an Additive Matrix. Compeau provides a theorem that if $\boldsymbol{M}$ is an Additive Matrix, then there [will] exist exactly one simple tree that perfectly fits it. An Additive Matrix is a Distance Matrix from which a tree can be fit, and is generally qualified by meeting the '4 Point Condition' requirement which we will not detail for brevity. A Simple Tree is defined as having no nodes of degree 2, i.e. no node may connect to exactly two other nodes. Our definition of a valid Phylogenic Tree satisfies this condition, as leaf nodes have degree 1 (to their parent) and all internal nodes sans the root have degree 3 (to their parent and children). The root node is a special case we handle later, as these methods produce unrooted trees from which roots need to be further inferred.

Suppose Distance Matrices are not additive? There do exist methods for producing an "approximate fit" such as the UPGMA Heuristic Method, as will be discussed later in this section.

Problem Definition for Distance-Based Phylogenic Tree Reconstruction

With some important concepts introduced, we can now express the generic problem definition for the Distance-Based Phylogenic Tree Reconstruction methods:

Given the input of a Distance Matrix $\boldsymbol{M}$, construct an inferred Phylogenic Tree $\boldsymbol{T}$ that best fits the matrix, if such a phylogeny exists. If $\boldsymbol{M}$ is an Additive Matrix, then produce the optimal best-fit thereof.

Limb Length and Computation Thereof

Definition and Intuition: Two of the three methods in this section, Additive Phylogeny and Neighbor Join, utilize some form of a 'Limb-Length' distance; such that what Limb Length is and how its calculated encompass the key to their solution methods. Limb Length refers to the distance from a leaf node to its parent … which should appear suspect, as we have assumed all parent and ancestor species are unknown and/or unknowable, thus their distance values likewise unknown. Certainly, the Distance Matrix doesn't contain, nor can infer this information, *right?!?...*

A First Step - Distance Between 'Sibling' Nodes: Compeau introduces the means to infer limb length by considering two 'neighbors' i.e. sibling nodes $\boldsymbol{p}$ and $\boldsymbol{q}$ which share the same parent node $\boldsymbol{r}$. We know from our introduction in Section 1 of Yip's 'Rare Mutations Assumption' that such siblings are inclined to have the most similarity to each other, thus their distance matrix scores should be minimal with respect to one another. We're also aware that 'close cousins' may have a matching score - which we will need to watch out for. Regardless, it then makes sense that if $\boldsymbol{p}$ and $\boldsymbol{q}$ are siblings - then the sum of their two edges to their parent should total their distance with each other, i.e. the distance from $\boldsymbol{p}$ to $\boldsymbol{r}$ plus the distance from $\boldsymbol{q}$ to $\boldsymbol{r}$ should equal the distance from $\boldsymbol{p}$ to $\boldsymbol{q}$. However, it does not suffice to simply divide the siblings' distances in two and assign the edge weights one half each, as they could be uneven. Can we use other entries in the distance matrix?

A Second Step - Limb Length Via Sibling, Parent, and non-sibling leaf: Compeau then invites the idea that the distance from node $\boldsymbol{p}$ to some other node $\boldsymbol{b}$ is the distance from $\boldsymbol{p}$ to its parent $\boldsymbol{r}$ plus the distance from $\boldsymbol{r}$ to $\boldsymbol{b}$, and analogous for node $\boldsymbol{q}$. As we know the distance from $\boldsymbol{p}$ to $\boldsymbol{q}$, $\boldsymbol{p}$ to $\boldsymbol{b}$, and $\boldsymbol{q}$ to $\boldsymbol{b}$, we can infer a distance from $\boldsymbol{r}$ to $\boldsymbol{b}$ from these facts and some algebra to produce the following equation:

$$\boldsymbol{dist_{p,b} = M[p][b] = dist_{p,r} + dist_{r,b}}$$
$$\boldsymbol{dist_{q,b} = M[p][b] = dist_{q,r} + dist_{r,b}}$$

$$\boldsymbol{dist_{p,q} = M[p][q]}$$
$$\boldsymbol{dist_{r,b} = \frac{dist_{p,b} + dist_{q,b} - dist_{p,q}}{2} = \frac{M[p][b] + M[q][b] - M[p][q]}{2}} \therefore$$
$$\boldsymbol{dist_{p,r} = dist_{p,b} - dist_{r,b} = M[p][b] - \left(\frac{M[p][b] + M[q][b] - M[p][q]}{2}\right)} \therefore$$
$$\boldsymbol{dist_{p,r} = \frac{M[p][b] + M[q][b] - M[p][q]}{2}} \wedge \boldsymbol{dist_{q,r} = \frac{M[q][b] + M[p][b] - M[p][q]}{2}}$$

Note that this method utilizes distance to some other node of min distance for which we assume is the neighbor. While a great intuition which is sometimes correct and does have merits that will prove useful and always correct further on, this method suffers from a flaw which we disregarded in the previous paragraph: 'close cousins' i.e. nodes of similar or equal distance to the actual sibling; and this flaw can produce such erroneous results.

A Third Step - Limb Length With Consideration to All Possible Leaves: Compeau introduced the previous formula to set up the intuition for what will be a correct distance measurement from a leaf to its parent via utilizing its sibling node and a non-sibling connected by the parent; except the previous method cannot detect 'close cousins'. Can it still be utilized? The answer is yes, with some modification and at a cost. Foremost, we introduce the new equation which we call 'LimbLength', which utilizes the previous equation and its intuition, though only utilizing leaf nodes and with more consistency over the domain thereof as follows:

$$\boldsymbol{LimbLen(x) = \min(M[x,p], M[x,q] - M[p,q])} \ \forall \{\boldsymbol{p,q \in M \mid p \neq x \wedge q \neq x}\}$$

Basically speaking, this revised method computes a 'candidate' limb length for the input leaf given all combinations of the other leaves, and correctly returns the minimum value thereof. We mentioned a cost, which the time complexity of this operation; for which we compute via the combination of all nodes not the query taken two nodes at a time. That is, each iteration computes distances between a combination of two other nodes and the query, for all combinations thereof. The combinatorics and runtime analysis are as follows:

$$\boldsymbol{T(n) = {}_{n-1}C_2 = \frac{(n-1)!}{2! * (n-3)!} = \frac{(n-1) * (n-2) * (n-3)!}{2 * (n-3)!} = \frac{(n-1) * (n-2)}{2} \approx n^2} \therefore$$
$$\boldsymbol{T(n) \in n^2}$$

We will reserve comment on this quadratic nature as Compeau invites an implementation of this method without discussion on any optimization algorithms thereof, which would be out of scope for an introduction in any case

Distance-Based Solution Method 1: Additive Phylogeny Algorithm

With a more accurate limb length function defines, Compeau then discusses the Additive Phylogeny algorithm for creating Distance-Based Phylogenic Trees from Distance Matrices known to be Additive Matrices as follows, which we reproduce with minor edits and additional details:

Algorithm:

*AdditivePhylogeny(**M**)*

1. *Pick an arbitrary leaf* $\boldsymbol{x}$

2. *Compute its limb length via* $\boldsymbol{LimbLen}(\boldsymbol{x})$
3. *Subtract this limb length from each entry in the* $\boldsymbol{x^{th}}$ *row and col of* $\boldsymbol{M}$ *to effect matrix* $\boldsymbol{D^{BALD}}$*. This step effectively 'blends' the node into its currently unknown parent, from which a correct location can later be found from which to attach to the parent (if it exists by that time) else create the parent via Step 6.*
4. *Then remove the* $\boldsymbol{x^{th}}$ *row and col of* $\boldsymbol{D^{BALD}}$ *to effect the* $\boldsymbol{n \times 1}$ *matrix* $\boldsymbol{D^{TRIM}}$*. A representation of* $\boldsymbol{M}$ *with this change reflected will form in Step 5.*
5. *Construct Simple Tree representation of* $\boldsymbol{D^{TRIM}}$ $\boldsymbol{Tree}(\boldsymbol{D^{TRIM}})$ *via recursion with call of* $\boldsymbol{AdditivePhylogeny}(\boldsymbol{D^{TRIM}})$
6. *Identify 'attachment point' in* $\boldsymbol{Tree}(\boldsymbol{D^{TRIM}})$ *where the leaf should be attached. This could either be an existing ancestral node, else require creating a new ancestor along some edge. In any case: the limb length theorem is utilized to infer that the attachment point must be somewhere on the path from the two nodes composing limb length* $\boldsymbol{D^{BALD}}$ *for* $\boldsymbol{x}$ *as follows:*

$$\boldsymbol{Point} = \left(\boldsymbol{D^{BALD}}[\boldsymbol{x},\boldsymbol{p}] + \boldsymbol{D^{BALD}}[\boldsymbol{x},\boldsymbol{q}] - \boldsymbol{D^{BALD}}[\boldsymbol{p},\boldsymbol{q}]\right) = \boldsymbol{0} \rightarrow$$
$$\boldsymbol{Point} = \left(\boldsymbol{D^{BALD}}[\boldsymbol{x},\boldsymbol{p}] + \boldsymbol{D^{BALD}}[\boldsymbol{x},\boldsymbol{q}] - \boldsymbol{D^{BALD}}[\boldsymbol{p},\boldsymbol{q}]\right) + \boldsymbol{D^{BALD}}[\boldsymbol{p},\boldsymbol{q}] = \boldsymbol{D^{BALD}}[\boldsymbol{p},\boldsymbol{q}] \rightarrow$$
$$\boldsymbol{Point} = \boldsymbol{D^{BALD}}[\boldsymbol{x},\boldsymbol{p}] + \boldsymbol{D^{BALD}}[\boldsymbol{x},\boldsymbol{q}] = \boldsymbol{D^{BALD}}[\boldsymbol{p},\boldsymbol{q}] \therefore$$
$$\boldsymbol{Point} = \boldsymbol{D^{BALD}}[\boldsymbol{p},\boldsymbol{q}]$$

7. *Attach it at the point with a single edge of length* $\boldsymbol{LimbLen}(\boldsymbol{x})$ *computed in Step 2*
8. *Repeat/Recurse until complete tree is formed.*

How To 'Force' a solution from Non-Additive Matrices, and Why Not To Force

As discussed above, this method will work for Additive Distance Matrices, but not necessarily Non-Additive Distances Matrices. Towards this, Compeau offers the idea of a Linear Least Squares optimization algorithm on the weight assignments for an output Phylogenic Tree given its Distance Matrix. A quadratic error/loss function, which Compeau calls 'Discrepancy', scores the difference between distances represented in M (i.e. the 'control') versus in the resulting tree T (the experiment).

This needs to be performed on repeated attempts at producing a successful configuration of the weights, quite similar to problems in Linear Programming and/or Machine Learning (and a possible candidate thereof - for that matter). However: Compeau uses the Linear Least Squares method as a cautionary tale due to the complexity of possible arrangements of weights, which is exponential on input 'n' and thus an NP-Complete Problem; not a good idea. Fortunately, there are other methods which could work for Non-Additive Distance Matrices such as the UPGMA Heuristic Method which we discuss next.

Ultrametric Trees and Properties Therein

The next method we discuss, UPGMA, makes use of Ultrametric Trees and some of the properties they encompass. An Ultrametric Tree is a rooted Binary Tree such that the path length from any leaf node to the root node is the same value; and note that while the Additive Phylogeny Method doesn't necessarily produce a root ancestor - this method will Compeau and Yip discuss similar methods each from different perspectives, which we discuss first.

Compeau introduces a completed Phylogenic Tree for the great apes, and discusses the formation and properties of its edges and nodes. Foremost, the tree has values assigned to each node called 'age', which corresponds to the time period in millions of years ago when the speciation event occurred; such that all existing species i.e. leaves have an age of zero. With the nodes composing this 'Molecular Clock' assigned

age values, edge value assignment can begin. This is a very simple calculation of the difference between the age values of the two nodes linked by a certain edge, for all edges. Lastly, he then invites the idea that due to how the edges were constructed: this tree is also an Ultrametric Tree.

Yip introduces a kind of corollary concept that if the pairwise distances between all nodes i.e. species encompass an Ultrametric Tree, then such a tree is an 'Additive Tree'; which Yip defines as a tree such that the Distance Matrix length of any 2 nodes is equal to the total length of the branches in the tree between them; which suits the definition of an Ultrametric Tree. He then provides the following criteria evaluation by which a Distance Matrix can be determined to compose an Ultrametric Tree:

1. All distances must be positive $\boldsymbol{dist(x,y) \geq 0}$
2. Matrix diagonal must be zeros $\boldsymbol{dist(x,x) = 0}$
3. Matrix Symmetry Property $\boldsymbol{dist(x,y) = dist(y,x)}$
4. Triangle Inequality:

$$\boldsymbol{dist(x,y) + dist(y,z) \geq dist(x,z) \wedge dist(x,y) < max(dist(x,z), dist(y,z)}$$

Distance-Based Solution Method 2: UPGMA Method / Algorithm

UPGMA is an acronym for Unweighted Paired Group Method via Arithmetic mean. It is a Heuristic Method which constructs Ultrametric Phylogenic Trees from both Additive and Non-Additive Distance Matrices via 'clustering' i.e. assembling small subtrees of the greater tree part-by-part into larger and larger clusters which will compose the completed tree. Both references utilize the Distance Matrix for this method, regardless to the introduction example discussed by Compeau. Lastly, the 'Heuristic' behind this method is the assumption mentioned in Section 1 that internal nodes -> speciation events wherein one species diverges into two species.

Algorithm:

UPGMA($\boldsymbol{M}$)

1. *Form a single element cluster for each species in $\boldsymbol{M}$, of which will compose the leaf nodes in $\boldsymbol{T}$.*
2. *Find the two 'closest clusters' $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$ according to their average distance. We will spare the mathematical formulae for this computation for brevity, and resolve that this step involves first computing the average distance between all nodes within two clusters for each cluster, and then finding the min argument thereof.*
3. *Merge $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$ into a single cluster $\boldsymbol{C}$ by first creating a new node representing $\boldsymbol{C}$ (thus, the cluster $\{\boldsymbol{C_1}, \boldsymbol{C_2}\}$), then connecting the subtree/cluster root of $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$ to this node by 2 edges correspondingly, then setting the 'age' of this node to the average distance computed in Step 2.*
4. *Update $\boldsymbol{M}$ by first removing the entries for $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$, then adding in the new cluster $\boldsymbol{C}$, then recomputing the average distance between all clusters therein.*
5. *Repeat Steps 2-4 until there exists a single cluster. This cluster composes the complete and rooted Ultrametric Phylogenic Tree from the Distance Matrix input.*

Disadvantages with UPGMA

Yip concludes his discussion of the UPGMA Method with some discussion on issues encountered with this method. Foremost, UPGMA does not have a method to assign good or even valid limb lengths for Non-Additive Matrices; whereas there are methods to do so from Additive Matrix-fit trees as discussed by Compeau in his introduction and by Yip via utilization of techniques in Linear Algebra such as Gaussian Elimination. Furthermore, once a merge of two [sub] clusters is performed, the resulting structure cannot

be changed. This is one of the main reasons why Additive Matrices might not result in a fit tree, as it's also difficult to decide on choosing ties between multiple average closest clusters, as could be effected by adjacent 'cousin' subtrees.

Distance-Based Solution Method 3: Neighbor-Join Algorithm

Overview: The Neighbor-Join Algorithm takes the concept of limb length from the Additive Phylogeny Algorithm alongside recursive clustering from the UPGMA Heuristic Method to form a solution method that works to either best fit trees for Additive Distance Matrices, else otherwise provide a 'good enough' heuristic approximation (as Yip describes: 'usually typically reasonable trees'). The key to this algorithm are two accessory data structures: the 1D matrix $\boldsymbol{TotalDist}$ and 2D matrix $\boldsymbol{D}^*$ a.k.a. the 'Neighbor-Join Matrix'. For the latter: Compeau compares its utility towards solving the Phylogenic Tree Reconstruction problem to be on a scale similar to the utility of the Burrows-Wheeler Transform for matching patterns to a text corpus.

Data Structures: $\boldsymbol{TotalDist}$ is a $\boldsymbol{n \times 1}$ matrix whose values represent the sum of all distances within a given row from the Distance Matrix, which in turn (and pursuant to its name) encompasses the distance from a certain species to all other species, for all species in the input matrix. The Neighbor-Join Matrix $\boldsymbol{D}^*$ is a $\boldsymbol{n \times n}$ matrix which is defined as Step 1 of the algorithm which we discuss below. It's power comes from what the values within represent versus what the values in $\boldsymbol{M}$ represents. We noted when discussing our 'second step' towards a limb length calculation that the minimum between two pairs within $\boldsymbol{M}$ does not always engender that such pairs are neighbors, but this feature CAN be [difficultly] proven to exist within $\boldsymbol{D}^*$, from which Compeau invites further research into the proof itself which was published 10 years after the discovery of this data structure.

NeighborJoin($\boldsymbol{M}$)

1. Construct elements of $\boldsymbol{D}^*$ s.t.: $\boldsymbol{D}^*[\boldsymbol{i},\boldsymbol{j}] = (\boldsymbol{n}-\boldsymbol{2}) * \boldsymbol{D}[\boldsymbol{i},\boldsymbol{j}] - \boldsymbol{TotalDist}(\boldsymbol{i}) - \boldsymbol{TotalDist}(\boldsymbol{j})$
2. Find a min pairwise element of $\boldsymbol{D}^*[\boldsymbol{i},\boldsymbol{j}]$
3. Compute $\boldsymbol{\delta_{i,j}} = \left(\boldsymbol{TotalDist}(\boldsymbol{i}) - \boldsymbol{TotalDist}(\boldsymbol{j})\right)\left(\frac{\boldsymbol{n}}{\boldsymbol{2}}\right)$. This is the value of the 'patrial limb length' which will be used to compute the actual limb lengths for both nodes.
4. Set $\boldsymbol{LimbLen}(\boldsymbol{i}) = \left(\frac{\boldsymbol{1}}{\boldsymbol{2}}\right) * (\boldsymbol{D}[\boldsymbol{i},\boldsymbol{j}] + \boldsymbol{\delta_{i,j}})$ and $\boldsymbol{LimbLen}(\boldsymbol{j}) = \left(\frac{\boldsymbol{1}}{\boldsymbol{2}}\right) * (\boldsymbol{D}[\boldsymbol{i},\boldsymbol{j}] - \boldsymbol{\delta_{i,j}})$
5. Form the matrix $\boldsymbol{D}^{\oplus}$ by removing the $\boldsymbol{i^{th}}$ and $\boldsymbol{j^{th}}$ row and col from $\boldsymbol{D}$, then adding a new row and col $\boldsymbol{ij}$ such that for all remaining elements $\boldsymbol{k}, \boldsymbol{D}[\boldsymbol{k},\boldsymbol{ij}] = \frac{\boldsymbol{D}[\boldsymbol{i},\boldsymbol{k}] + \boldsymbol{D}[\boldsymbol{j},\boldsymbol{k}] - \boldsymbol{D}[\boldsymbol{i},\boldsymbol{j}]}{\boldsymbol{2}}$. We note the similarity between this equation and similar ones seen with the Additive Phylogeny method.
6. Recursion Step: Call $\boldsymbol{NeighborJoin}(\boldsymbol{D}^{\oplus})$ to obtain $\boldsymbol{Tree}(\boldsymbol{D}^{\oplus})$, similar to Additive Phylogeny.
7. Reattach limbs of $\boldsymbol{i}$ and $\boldsymbol{j}$ to compose $\boldsymbol{Tree}(\boldsymbol{D})$, similar to Additive Phylogeny
   - Output Note: The resulting tree might be unrooted!

Rooting an Unrooted Tree formed from Neighbor-Join: Yip provides some tips for how to produce a root. First, he suggests use of an 'outgroup' i.e. some known 'most distant species' that speciated before all others in the input set such that the root must be somewhere along its limb to all the other nodes. Further, he discusses working with the limb distances between nodes to find one that is equidistant from all other nodes, similar to the structure of the Ultrametric tree in UPGMA.

Conclusion

We discussed three methods for Phylogenic Tree Reconstruction via Distance-Based solutions. We noted the main advantage of these methods were in the simplification of the input data by use of the Distance Matrix formed by multiple alignment of the sequence. We noted the main disadvantage was that these methods are unable to 'speak for' the sequence composition of the ancestor nodes, due to the information lost in the computation of the distance matrix input, as a consequence thereof.

As to comparisons of these three methods, Compeau provides a good intuition for the first two, and we conclude with a recap of the main benefit in the third method. Additive Phylogeny is good for producing Phylogenic Trees that fit Additive Matrices, but fail to produce 'approximate fits' for Non-Additive Matrices. UPGMA is good for producing Phylogenic Trees for Additive or Non-Additive Matrices, but are not necessarily guaranteed to produce best fitting trees for Additive Matrices as with Additive Phylogeny. Lastly, Neighbor Join provides 'the best of both worlds' by providing the benefit of producing Phylogenic Tree which, to paraphrase Compeau: "properly fit additive matrices, while also providing a heuristic approximation for non-additive matrices".

Bibliography

1) Compeau, Philip. Pevzner, Pavel. (2014) *Bioinformatics Algorithms: An Active Learning Approach*. [Chapter 7] Active Learning Publishers.
2) Yuk-Lap Yip, Kevin (2017, September 4th). *Kevin Bioinformatics*. Retrieved from https://www.youtube.com/channel/UCk2ozjkbfpteeJUolHWNsMw/videos
3) Compeau, Philip (2015, July 24th). *Chapter 7 – Which Animal Gave Us SARS?* Retrieved from https://www.youtube.com/playlist?list=PLQ-85lQlPqFPhJxNcuSOxLTNmbp3NzVtU
4) Hug, Laura A et.al. (2016) *A new view of the tree of life*. Nature Microbiology Retrieved from https://www.nature.com/articles/nmicrobiol201648