university of groningen / faculty of science and engineering / mathematics and applied mathematics

Computational Stemmatology Literature Review

Computational Stemmatology Literature Review

November 2023

Student: Darren Zammit s5284236

1 A Brief Introduction to Computational Stemmatology

Copying a text multiple times over long periods of time usually results in changes in its content. Stemmatology is a methodical approach to textual criticism and is based on the assumption that if two or more surviving witnesses have common errors then they were likely derived from a common intermediate ancestor (hyparchetype). Relations between the lost ancestors are determined similarly and such predicted ancestors are placed in with all surviving witnesses in a family tree or "stemma" descended from a single (not necessarily known) "archetype".

Once the stemma is derived, the text of the archetype is constructed by examining variants from the hyparchetypes closest to the archetype and selecting the most likely ones. This phase is called selection. The most common variant at each locus (position in the script) at the same level of the tree, is selected as the most likely variant. If multiple variants occur equally often, then the philologist must use his judgement to determine which makes the most sense. A common example is that scribes tend to replace more difficult words by simpler words, thus whilst constructing the archetype, the philologist typically would favour the more complex words.

The text must then undergo the process of "examination" during which the philologist must identify errors as in some loci it may be that none of the surviving variants preserve the original text. If the text is determined to be corrupt, it must be corrected through "emendation". At the end of this process it is sometimes possible to reconstruct a text closer to the original than any of the surviving witnesses.

The process of deriving a stemma has been described in mathematical terms [**?**], however some of the crucial steps in the process are largely impregnable to algorithmic description. This has led some to dismiss stemmatology as a scholarly discipline. Finding informative shared errors as is determining whether one is dealing with primary or secondary texts. Usually, familiarising oneself with a text and its contexts is instrumental in understanding its transmission which can be very time-consuming since the philologist often starts knowing very little about the original text or its author(s), the author's writing style, the environment where the text was written or the time-period it was written in. This information plays a crucial role in determining the direction of copying. With each successive pass through the scripts, the philologist understands and sees more subtleties, thus the process of constructing a stemma is not linear (though this can lead to the philologist to see things which are not there). This non-linearity makes the process difficult to program fully, however some steps in the process could be computerised to save time for the philologist or derive a hypothetical stemma when one deals with impregnable (or very numerous) scripts. There exists a wide variety of approaches in dealing with textual and stemmatic reconstruction and an equally wide variety of tools are available. Computer tools have made many steps in editing a text easier and more efficient. Moreover, computer simulations allow for the study of the behaviour of large amounts of textual data. On top of this, computational stemmatology has potential applications in plagiarism detection [10], analysing the evolution of computer viruses [11] and content-based social network analysis [12].

Graphical modelling takes a central role in constructing stemmata [1]. Contemporary methods – which are primarily analogous to those in phylogenetics, the study of the evolutionary history of species – can be split into distance-based [2], parsimony-based [3], and statistical methods [4]. In these methods every word in the text is equally important. In reality there are many different kinds of errors and changes, some more significant than others, leading to the algorithm failing to faithfully construct a stemma. For example, spelling mistakes or changing symbols should have far less impact on a script than adding or redacting text. Another issue which these methods struggle to address is that of contamination, that is when a text is copied from multiple manuscripts [1]. If different scripts were used for different chapters then this could lead to outputs which faithfully reflect the stemma of most chapters but not all if different chapters were copied from different manuscripts.

Since Lee's seminal paper [5] in applying computational phylogenetics models in stemmatology multiple advances in computational stemmatology have been made. However, most papers in computational stemmatology are authored by computer scientists and philologists with mathematicians and statisticians representing a small minority of authors [1]. Moreover, algorithms designed specifically for stemmatology, even ones whose framework is adapted for stemmatolology, are very few in number [1]. Computational methods usually rely on transforming the data into analogues of genetic data and feeding it directly into a

phylogenetics program as though it were phylogenetic data, with very few algorithms developed exclusively for stemmatology.

2 Trees

3 Algorithms

3.1 Distance Based Methods

Distance based methods are two-step processes. In the first step one calculates a symmetric dissimilarity matrix with zeros on the diagonal for the input whilst the second step involves constructing a tree from the distance matrix. Such matrices do not generally satisfy the triangle inequality and thus are referred to dissimilarity matrices rather than distance matrices. Many distance-based tree estimation methods are polynomial time and efficient, and thus are favoured over most other algorithms which tend to be computationally expensive.

Definition 3.1. *An ultrametric matrix is an $n \times n$ matrix M corresponding to distances between the leaves in a rooted edge-weighted tree T (with non-negative edge weights) where the sum of the edge weights in the path from the root to any leaf of T does not depend on the selected leaf.*

3.1.1 UPGMA (Unweighted Pair Group Method with Arithmetic Mean)

UPGMA is an agglomaration algrothm which computes a rooted tree from an input distance matrix. In the first iteration, it finds the pair of taxons x, y which have the minumum distance, clusters them together as $\{x, y\}$, and the distance from $\{x, y\}$ to every other taxon z is calculated as the average of $d(x, y)$ and $d(x, z)$:

$$d(\{x, y\}, z) = \frac{d(x, z) + d(y, z)}{2}. \tag{1}$$

In the subsequent iterations, the elements are clusters that are either singletons (the original taxons) or sets of two or more taxons. In each iteration, the pair of clusters A and B with the smallest distance are clustered together. The distance matrix is then updated by removing the rows and columns for A and B and replacing them with a row for $A \cup B$:

$$d(A \cup B, C) = \frac{|A|d(A, C) + |B|d(B, C)}{|A| + |B|}. \tag{2}$$

This process is repeated until all the taxons are merged into a single cluster which produces a rooted tree. The distance matrix can be updated in a variety of ways such as by simply removing the row and column corresponding to one of the closest two taxons and this has an impact on the output tree. However, the main idea of this algorithm is to iteratively find the closest pair of clusters and replace them with a new cluster.

All agglomerative methods which clusters the closest taxons would fail on any input distance matrix which has its closest taxon not being siblings. Moreover, for all variants of UPGMA, given that it follows such an algorithmic design allows one to determine the first closest pair (up to ties), thus, if the dataset has only three leaves, one can determine the rooted tree that will be constructed, and if the dataset has four leaves, then one can determine the unrooted tree it will produce. Hence, for at least some model conditions, proving that UPGMA produces the incorrect tree is generally easy. Given that the first sibling pair it produces is not a true closest pair, the result will be incorrect, no matter what the subsequent steps are.

3.2 Neighbour Joining

Neighbor joining is one of the most widely distance-based method due to its accuracy and even to this day research into why it performs so well is still ongoing. Neighbor joining is also an iterative agglomerative technique wherein the tree is built from the bottom up. The input is an $n \times n$ dissimilarity matrix d and in the first iteration, the n leaves are all in their own clusters. In subsequent iterations, each cluster is a set of leaves, but the clusters are disjoint. At the beginning of each iteration, the taxa are partitioned into clusters,

and each cluster has a rooted tree that is leaf-labeled by the elements in the cluster. During each iteration, a pair of clusters is selected to be made siblings, resulting in the cluster's respective trees to be merged into a rooted tree by making their roots siblings. Given three sub-trees, the three sub-trees are merged into a tree on all the taxons by adding a new node, r, and making the roots of the three sub-trees adjacent to r. Once the tree is generated the root is ignored such that neighbor joining yields an unrooted tree. The advantage of Neighbor joining over UPGMA is that it chooses which pair of clusters to make siblings (and how to update the distance matrix) using a more sophisticated strategy:

Initialisation: Compute the $n \times n$ Q matrix:

$$Q_{ij} = (n-2)d_{ij} - \sum_{k=1}^{n}(d_{ik} + d_{jk}). \tag{3}$$

Whilst $n > 3$: Find the pair i, j minimising Q_{ij} and call that pair a, b. Make the rooted trees associated with taxons a and b siblings, and call the root of the tree formed u. Update the distance matrix by deleting the rows and columns for a and b, and including a new row and column for u and set $d_{u,k} = \frac{d_{ak}+d_{bk}-dab}{2} \forall k \neq u$. Subtract n by 1.

If $n = 3$; return the star tree with a single internal node v where the roots of the three rooted trees are all adjacent to v.

3.3 Parsimony Based Methods

Maximum parsimony is another concept which can be used to construct many trees. The output is a tree T in which the input sequences are placed at the leaves of T and additional sequences are placed at the internal nodes of T such that the total number of changes over the entire tree, is minimised. Another way of defining maximum parsimony is as the Hamming Distance Steiner Tree Problem wherein the input is a set of sequences and the output is a tree connecting these sequences at the leaves with other sequences (the Steiner points) at the internal nodes, which minimises the total of the Hamming distances on the edges of the tree. The Hamming distance between two sequences of the same length is the number of positions in which they disagree, thus the sum of the Hamming distances on the edges of the tree is equal to its tree length. Determining the optimal tree under this criterion is an NP-hard problem [**?**], and hence heuristics are used to find suitable solutions which are not necessarily globally optimal. Maximum parsimony heuristics are computationally very intensive on large datasets. Moreover, this method has been proven to be statistically inconsistent under standard DNA sequence evolution models, and may even converge to the wrong tree as the sequence length increases.

Given a tree T and character state assignments to the leaves of T. The aim is to find character state assignments to the internal nodes of T so as to minimise the number of edges with different states at the edge's endpoints. This is the "fixed tree parsimony problem" or the "small parsimony problem" to distinguish it from the "large parsimony problem," which seeks to find the best tree and its internal node labels achieving the optimal parsimony score. Since only the total number of changes is counted, the parsimony score of a tree is not affected by the tree's rootedness. The input is an unrooted binary tree T with set of leaves S, and character $c : S \to \{1, 2, ..., r\}$ whilst the output is the assignment of character states to the internal nodes of T such that the number of edges $e = (u, v)$ where $c(u) \neq c(v)$ is a minimum. For binary trees as in phylogenetics the Fitch Margoliash algorithm can be used to calculate the parsimony score and determine an optimal labeling of the internal nodes. For arbitrary trees the Hartigan algorithm is used.

3.4 Minimum Evolution

The minimum evolution is a collection of methods. The input is an $n \times n$ dissimilarity matrix d, and every tree topology on n leaves is assigned edge weights using some version of least-squares. The total length of each tree is then computed by adding up the lengths of all the branches in the tree. The tree T with its edge weighting w which minimises $\sum_{e \in E(T)} w(e)$ is returned. Such methods are distinguished by how they define the edge weighting of each tree. For example, given a tree T and an input dissimilarity matrix d, the optimal weights on the edges could be based on minimizing the L_2 (OLS) or the weighted L_2 (WLS) distances. Since least squares is used, this runs in polynomial time. On the other hand, finding the tree

with the minimum total branch length is generally much harder. For example, it is NP-hard under the OLS criterion for integer branch lengths and under a variant of the WLS criterion called "balanced minimum evolution". Such methods, tend to be heuristics without necessarily provable guarantees about solving their optimization problems. The statistical consistency of the exact solutions to such methods depends on how the branch lengths are defined.

3.5 Maximum Likelihood Based Methods

Maximum likelihood Estimation is a very popular statistical technique where an explicit model of nucleotide substitution is used to evaluate trees. The model tree consists of a tree topology T and a set of parameters θ. MLE takes as input a set S of sequences of identical lengths, and finds the model tree $(\hat{T}, \hat{\theta})$ which maximises the probability that the sequences occur given the input tree T and the parameters θ $\mathbb{P}[S|T^*, \theta^*]$. The only difference between MLE approaches with different evolution models are the parameters. Since MLE requires the estimation of all the model parameters, increasing the number of parameters rapidly increases the computational demands and the risk of over-fitting. Moreover, whilst the MLE the differences in scores between different trees can be insignificant and thus calculating the optimal and the near-optimal trees is sometimes the objective. Moreover, MLE algorithms are not always statistically consistent. Finding an optimal ML tree is an NP-hard problem, thus requiring the use of heuristics rather than finding the exact solution. Since the number of trees on n grows exponentially with n, examining each possible tree topology is not feasible and thus heuristic searches are used to explore the treespace. Heuristic searches for ML are more complicated than heuristic searches for maximum parsimony because even scoring a tree is computationally intensive as the estimation of numeric parameters on a tree is also computed heuristically and does not guarantee global optima. Hence, ML phylogeny estimation is complicated and computationally intensive, even for basic sequence evolution models. The evolution of text is far more complicated than the evolution of genetic sequences due to the larger number of possible characters, various different mechanisms by which errors occur. Unlike genetic changes which are usually assumed to be independent, in textual evolution spatial correlations are very common. Moreover, the fact that the sequences are typically much longer, the multifurcating nature of textual evolution and the fact that not all extant scripts could be placed in the leaf nodes of the tree vastly increase the computational demands. Evolution models are defined by their parametric transition probability matrices, which in the case of DNA sequences can be expressed in the form

$$P(t) = \begin{pmatrix} P_{A\to A}(t) & P_{G\to A}(t) & P_{C\to A}(t) & P_{T\to A}(t) \\ P_{A\to G}(t) & P_{G\to G}(t) & P_{C\to G}(t) & P_{T\to G}(t) \\ P_{A\to C}(t) & P_{G\to C}(t) & P_{C\to C}(t) & P_{T\to C}(t) \\ P_{A\to T}(t) & P_{G\to T}(t) & P_{C\to T}(t) & P_{T\to T}(t) \end{pmatrix} \tag{4}$$

where $P_{A\to A}(t)$ is the probability that a site in state x evolves into state y in time t. There are various ways to define the transition probabilities and similar models exist for protein sequences however for textual evolution such models are not well established. Textual evolution can be modeled in a similar manner. In textual evolution t does not have the same role as in biological evolution. Namely, the existing manuscripts can be used as sources for copying after an arbitrarily long time and thus a copies made of older manuscripts cannot be assumed to contain more errors than a copies made of newer manuscripts. Another major difference in modeling text compared to genomic sequences is that the alphabet is not fixed.

3.6 Bayesian Methods

Bayesian methods are similar to ML methods in that they also calculate likelihoods of trees based upon explicit parametric mathematical models of evolution. Instead of attempting to find the model tree with the largest probability of generating the observed data, Bayesian methods sample from the set of model trees with frequency proportional to their likelihoods, given the observed data. The result is that Bayesian methods produce a set of trees rather than a single tree which is then used to evaluate the statistical support for different evolutionary hypotheses. Bayesian methods are usually implemented using Markov Chain Monte Carlo (MCMC) techniques where a Bayesian method performs a random walk through the space of model trees. Each time it visits a model tree, it computes the probability of the observed data, using Felsenstein's Pruning Algorithm. The probability of accepting the new model tree depends on whether the probability

has increased or decreased; for example, in Metropolis–Hastings MCMC, the new tree may be accepted even if the probability has decreased. Whenever the new tree is accepted, the MCMC chain continues from the new model tree. The Bayesian MCMC walk is guaranteed to converge to the stationary distribution on model trees, as long as the MCMC walk satisfies the detailed balance property, which essentially says that $\pi(i)P_{ij} = \pi(j)P_{ji}$, where $\pi(x)$ is the equilibrium probability of state x and P_{xy} is the probability of moving to state y from state x. After the MCMC chain has run long enough to become stationary, a sample of the model trees it visits is then taken from the trees visited after some "burn-in" period. If a point estimate of the tree topology is desired, then a summary of the distribution such as a consensus tree can be computed using various techniques. The frequency with which a tree topology appears in the sample is called the posterior probability. Another common summary statistic is the maximum a posteriori (MAP) tree, which is the tree topology that appears the most frequently in the sample. A third approach assembles a set of tree topologies, starting with the most frequently observed trees and then decreasing, until a desired threshold is achieved (e.g., where 95 percent of the probability distribution is in the set), and then computes a point estimate based on this subset of trees. Each point estimate can then be used as an estimated phylogeny, and the support for the branches of the phylogeny can similarly be computed based on the observed distribution.

There are several challenges in using Bayesian methods. Bayesian methods require priors, and the choice of priors can affect the accuracy of the resultant trees. Another disadvantage is that Bayesian MCMC methods need to be run until they have reached the stationary distribution, and it is not at all straightforward to assess whether stationarity has been reached. On larger datasets Bayesian methods become computationally expensive as it typically increases the time to reach stationarity.

3.7 Compression Based Methods

3.8 Accuracy Measures

The methods are evaluated based on their success of finding a stemma that is close to the true stemma. Since two arbitrary latent tree structures are being compared, the usual measures such as counting the number of shared edges do not apply since there is no one-to-one correspondence between the latent nodes in the true stemma and the estimated one. It is not guaranteed that the number of latent nodes will be the same, let alone their positions in the stemma. Thus, we employ the average sign similarity score [13].

This measure depends on the number of edges between pairs of nodes and ignores possible edge length information. For each pair of nodes A, B, the true distance, $d(A, B)$ is defined as the number of edges on the shortest path between A and B in the true stemma. Thus $d(A, B) = d(B, A)$. Similarly, the same quantity computed from a hyptothetical stemma is denoted by $d'(A, B)$. Whenever the hypothesised stemma is correct, $d'(A, B) = d(A, B)$ for all pairs of nodes, and otherwise the two values differ for some A and B. Given three nodes A, B, and C, we can measure the distances $d(A, B)$ and $d(A, C)$. We consider which one of these two distances is greater than the other, or whether they are equal. Let $sign(d(A, B) - d(A, C))$ be the sign of the difference between the two distances, so that the index:

$$u(A, B, C) = 1 - \frac{1}{2}|\text{sign}(d(A, B) - d(A, C)) - sign((d'(A, B) - d'(A, C))|, \tag{5}$$

so that:

$$\begin{aligned} 1 \text{ if } sign(d(A, B) - d(A, C)) &= sign(d'(A, B) - d'(A, C)) \\ 0 \text{ if } sign(d(A, B) - d(A, C)) &= -sign(d'(A, B) - d'(A, C)) \\ 1/2 \text{ otherwise.} & \end{aligned}$$

The average sign similarity score is the average of over all distinct observed nodes corresponding to the extant manuscripts (but not the latent nodes) given that none of the three nodes are equal to each other. This measure can be used with any pair of graphs, givenn that both include all the observed nodes and may include any number of additional nodes, which need not be identical for the correct and the proposed stemma. The computation of the distance requires that the pair-wise distances between all nodes are computed and recorded in a table. After this, the average can be computed.

References

[1] Roelli, Philipp. (2020). Handbook of stemmatology : history, methodology, digital approaches. Berlin, Germany: De Gruyter.

[2] M. Spencer and C. J. Howe, "Estimating distances between manuscripts based on copying errors," Literary and Linguistic Computing, vol. 16, p. 467–484, 2001.

[3] W. M. Fitch, "Toward defining the course of evolution: minimum change for a specific tree topology," Systematic Biology, vol. 20, p. 406–416, 1971.

[4] Q. Nguyen and T. Roos, "Likelihood-based inference of phylogenetic networks from sequence data by phylodag," in International Conference on Algorithms for Computational Biology, 2015.

[5] A. Lee, 'Numerical Taxonomy Revisited: John Griffith, Cladistic Analysis and St. Augustine's Quaestiones in Heptateuchum.' Studia Patristica XX: 24-32, 1989.

[6] T. Roos, T. Heikkilä, Evaluating methods for computer-assisted stemmatology using artificial benchmark data sets, Literary and Linguistic Computing, Volume 24, Issue 4, December 2009, Pages 417–433, https://doi.org/10.1093/llc/fqp002.

[7] D. L. Swofford, PAUP*. Phylogenetic Analysis Using Parsimony (*and Other Methods). Version 4. Sinauer Associates, Sunderland, Massachusetts. 2003.

[8] J. Felsenstein, PHYLIP (Phylogeny Inference Package) version 3.6. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle, 2005.

[9] F. Ronquist, and J. P. Huelsenbeck. MRBAYES 3: Bayesian phylogenetic inference under mixed models. Bioinformatics 19:1572-1574. 2003.

[10] C. Liu, C. Chen, J. Han, and P. S. Yu, "GPLAG: Detection of software plagiarism by program dependence graph analysis," Proc. KDD 2006, pp. 872–881, 2006

[11] J. Sun, S. Papadimitriou, C.-Y. Lin, N. Cao, S. Liu, W. Qian, "MultiVis: Content-based social network exploration through multi-way visual analysis," Proc. SDM 2009, pp. 1063–1074, 2009

[12] S. Wehner, "Analyzing worms and network traffic using compression," J. Comp. Secur., vol. 15, pp. 303–320, 2007

[13] T. Roos, T. Heikkilä, Evaluating methods for computer-assisted stemmatology using artificial benchmark data sets, Literary and Linguistic Computing, Volume 24, Issue 4, December 2009, Pages 417–433, https://doi.org/10.1093/llc/fqp002

[14] T. Roos, and Y. Zou. 2011. "Analysis of Textual Variation by Latent Tree Structures." In Proceedings of the 11th International Conference on Data Mining (ICDM-2011), edited by Diane J. Cook, 567–576. Los Alamitos, California: IEEE Computer Society. Available at: http://www.cs.helsinki.fi/u/ttonteri/pub/icdm2011.pdf. Accessed 28 October 2015.

[15] N. Friedman 1998. "The Bayesian Structural EM Algorithm." In Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference (1998), edited by Gregory F. Cooper, 129–138. San Francisco: Morgan Kaufmann. Available at: www.cs.huji.ac.il/ nir/Papers/Fr2.pdf Accessed 28 October 2015.

[16] T. Roos, T. Heikkilä and P. Myllymäki. 2006. "A Compression-Based Method for Stemmatic Analysis." In ECAI 2006: Proceedings of the 17th European Conference on Artificial Intelligence: August 29 – September 1, 2006, edited by Gerhard Brewka et al., 805–806. Amsterdam: IOS Press.