Statistical Genetics, Spring 2024

Class notes 3: Phylogenetic trees

Definitions:

- A **tree** is an undirected graph $T = \{V, E\}$ with no cycles.
- A **binary unrooted tree** has internal nodes of rank 3, and leaves of rank 1. If the tree is non-binary, internal ranks can be higher than 3.
- A **rooted tree** has one special ("root") node of rank 2.
- **Rooting** an unrooted tree: "hanging" it from an arc.

A **Phylogenetic tree** has observed sequences at the leaves, and the tree describes the "familial" relationships between the sequences, the root (if the tree is rooted) is the *most recent common ancestor* (MRCA) of all sequences.

In genetics it is not just a representation, as there is typically a *true* phylogenetic tree, since the sequences really do sit on a tree. For sequences of non-recombining DNA like mtDNA or NRY (non-recombining region of Y chromosome), a single true tree exists for the full sequences.

In presence of recombination there is not a single tree, but each non-recombinant unit (typically nucleotide/site) does have a tree, since it has gone through no recombination! If for a set of sequences, there has been no recombination between two (or more) neighboring nucleotides in the tree, then they share the same tree for this "region". Therefore autosomal sequences are truly represented by a *forest* of phylogenetic trees, one for each site or region.

A special situation is when the sequences are from different species (man, cow, bird). In this case, it is likely that all sites share the same tree ((human, cow),bird) because all of the sites *coalesce* between human and cow first, then with bird. These assumptions can be violated by:

1. Admixture: two separate species interbreeding and mixing their DNA (like modern humans and neanderthals)
2. Species that are close enough that some sites might coalesce between species rather than within species.

Phylogenetic inference

Now given a collection of sequences, we want to infer the phylogenetic tree that links them, how can we do it?

First we describe them as a table or matrix X:

	sites			
	1	2	.....	m
Seq 1	A	G	.....	T
Seq 2	A	C	.....	T
⋮	⋮	⋮	⋱ ⋱	⋮
Seq n	C	G	.....	T

Note that this table requires *alignment* of the sequences, which is a big and complicated algorithmic topic in itself, if there are many mutations, insertions and deletions involved. In fact, alignment and phylogenetic inference can be thought of as one process, since both have uncertainty and each informs and affects the other. We will assume for now alignment is not an issue.

Approaches to tree building

There are many, a brief overview of major approaches:

1. **Neighbor joining:** Given some distance metric between sequences:

   - Assume the two closest ones are neighboring leaves.
   - Combine them and infer their common ancestor.
   - Now you have $n-1$ sequences, repeat.

   This is the simplest and most computationally efficient approach.

2. **Maximum parsimony:** The goal is to infer a tree that requires a minimum number of mutations to happen to generate the observed sequences. This is an important combinatorial problem that has been extensively studied. Most parsimonious trees can be approximated well up to thousands of sequences, but this is not the focus of our discussion.

3. **Maximum likelihood:** Parameters that need to be inferred:

   - Tree structure (topology) $T = \{V, E\}$.
   - Arc lengthes $s(e) \forall e \in E$.
   - Parameters of the mutation model, as discussed last week.

   We will assume that mutations in different sites are independent (otherwise ML analysis is significantly complicated).

Root finding

For maximum parsimony and also ML if the mutation model is reversible (as we will see), the root plays no role in the calculations, so rooted and unrooted trees are equivalent. However in interpreting the tree we care about the location of the root: the tree ((modern human, chimpanzee),neanderthal) is very different from the (correct) tree ((modern, neanderthal), chimpanzee), although they are equivalent as unrooted trees.

Methods for finding the root:

- Using the *molecular clock* assumption: if evolution proceeds at a fixed pace, then the root is the point in the tree *furthest away from the leaves*, and also in same time distance from all leaves, and hence approximately same mutation distance from all leaf sequences. This assumption can help in rooting obvious trees like ((human, chimpanzee),frog), but is probably in itself very inaccurate in such cases.
- Using an *outgroup* — identifying (or adding) a sequence that we know for sure is outside the tree of all other sequences, like adding frog to the ((modern, neanderthal), chimpanzee) tree. We know the root will be on the edge leading to the outgroup.
  - Unrooted tree: ((modern,neanderthal),(frog, chimpanzee))
  - Hang it from frog branch and get (((modern,neanderthal), chimpanzee),frog)

Example: Rooting the human mtDNA tree: How did we find that the root is between the African branch L0 and all other African and non-African branches?
We could do it with molecular clock, which gives reasonable result.
Using outgroup:
Outgroup 1: Chimpanzee or neanderthal. Would also work, although the hypervariable control region is too different
Outgroup 2: Nuclear Mitochondrial Inserts (Numts). During evolution, some copies of mtDNA got accidentally copied in the autosomes. Since autosomal mutation rate is much slower, these are similar to ancestral sequences, perfect for rooting.

ML estimation of trees

Assume for now the toplogy T is known. X is our sequences matrix: $X_{i\cdot}$ is a row (sequence), $X_{\cdot j}$ is a column (site).
Also denote by Y a matrix of the (unknown) sequences at internal nodes. For example a rooted binary tree with 4 sequences has three internal nodes with sequences: $Y_{0\cdot}$ (root), $Y_{1\cdot}$, $Y_{2\cdot}$.
Denote the leaves by indexes $3, 4, 5, 6$.
From independence between sites: $Pr(X;\theta) = \prod_{j=1}^{m} Pr(X_{\cdot j};\theta)$.
If we start from assuming Y is also observed we can write:

$$Pr(X_{\cdot j}, Y_{\cdot j};\theta) = \pi_{0j} P_{Y_{0j}Y_{1j}}(s_{01}) P_{Y_{0j}Y_{2j}}(s_{02}) P_{Y_{1j}X_{3j}}(s_{13}) P_{Y_{1j}X_{4j}}(s_{14}) P_{Y_{2j}X_{5j}}(s_{25}) P_{Y_{2j}X_{6j}}(s_{26}),$$

where $P_{ij}(s) = P_{ij}^{\theta}(s)$ is the transition probabilities matrix from i to j in time s implied by the parameters. π are the "baseline" probabilities (for example stationary distribution).

Since Y is in fact unobserved, we have to sum over all possible Y matrices to get the proper likelihood:

$$Pr(X_{\cdot j};\theta) = \sum_{Y} \pi_{0j} P_{Y_{0j}Y_{1j}}(s_{01}) P_{Y_{0j}Y_{2j}}(s_{02}) P_{Y_{1j}X_{3j}}(s_{13}) P_{Y_{1j}X_{4j}}(s_{14}) P_{Y_{2j}X_{5j}}(s_{25}) P_{Y_{2j}X_{6j}}(s_{26}),$$

Removing the root

By reversibility we have: $\pi_{0j} P_{Y_{0j}Y_{1j}}(s_{01}) = \pi_{1j} P_{Y_{1j}Y_{0j}}(s_{01})$, and once we sum over all values of Y_0 we can see that necessarily:

$$\sum_{Y_0 j} \pi_{0j} P_{Y_{0j}Y_{1j}}(s_{01}) P_{Y_{0j}Y_{2j}}(s_{02}) = \pi_{1j} P_{Y_{1j}Y_{2j}}(s_{12} = s_{01} + s_{02}),$$

hence even if the true tree is rooted, there is no need to include the root in the calculations, as long as we do not assume a molecular clock.

Efficient computation: the Felsenstein algorithm

The calculation above requires — for a single set of parameters and single site — to sum over 4^{n-2} possible values of $Y_{\cdot j}$. This is intractable for $n = 10$ already for sure. However we can divide the computation between the nodes efficiently. For the example above, let's rewrite it:

$$\begin{aligned} Pr(X_{\cdot j};\theta) &= \sum_{Y} \pi_{1j} P_{Y_{1j}Y_{2j}}(s_{12}) P_{Y_{1j}X_{3j}}(s_{13}) P_{Y_{1j}X_{4j}}(s_{14}) P_{Y_{2j}X_{5j}}(s_{25}) P_{Y_{2j}X_{6j}}(s_{26}) = \\ &= \sum_{Y_{1j}} \pi_{1j} P_{Y_{1j}X_{3j}}(s_{13}) P_{Y_{1j}X_{4j}}(s_{14}) \left(\sum_{Y_{2j}} P_{Y_{1j}Y_{2j}}(s_{12}) P_{Y_{2j}X_{5j}}(s_{25}) P_{Y_{2j}X_{6j}}(s_{26}) \right). \end{aligned}$$

The second sum only considers the tree under Y_2, and depends on the tree above it only through Y_{1j}, denote it:

$$L_2(c) = \sum_{Y_{2j}} P_{cY_{2j}}(s_{12}) P_{Y_{2j}X_{5j}}(s_{25}) P_{Y_{2j}X_{6j}}(s_{26}),$$

then we have:

$$Pr(X_{\cdot j};\theta) = \sum_{Y_{1j}} \pi_{1j} P_{Y_{1j}X_{3j}}(s_{13}) P_{Y_{1j}X_{4j}}(s_{14}) L_2(Y_{1j}).$$

We can generalize this to an explicit efficient algorithm over a big tree. For a given node i assume its children nodes are l and r, and we have already calculated the corresponding functions $L_r(c), L_l(c)$. Then it is easy to see for site j:

$$L_i(c) = \left(\sum_{Y_{lj}} P_{cY_{lj}}(s_{il}) L_l(Y_{1j}) \right) \left(\sum_{Y_{rj}} P_{cY_{rj}}(s_{ir}) L_r(Y_{rj}) \right),$$

meaning calculating $L_i(\cdot)$ requires a fixed number of calculations, regardless of how deep the tree is. So calculating the total likelihood for m sites on n sequences takes only $O(nm)$.

Embedding the Felsenstein algorithm within real ML calculations

The calculation above is for a given tree and parameters, but we need to estimate all of these:

- Tree topology (this is also unknown in the general case)
- Branch lengths s_{ij}
- Mutation model parameters that go into calculation $P_{**}(s)$.

Given all of these we can use Felsentein's algorithm to efficiently calculate the likelihood, but now we need to embed this calculation within a method for finding MLE's for all of the parameters. The lengths and mutation model parameters appear in the expressions in complex but analytical manner, so we can hope to differentiate according to them and do gradient descent or even Newton's method with second derivatives. Or we can use a local search approach such as simulated annealing.

For finding the topology parameters it is more complicated, since the space of possible topologies is discrete and very big, it quickly becomes very difficult to search in that space. In practice, with a few dozen sequences it is still somewhat practical to do a full maximum likelihood. Beyond that, if we have a known topology it may still be possible to estimate branch lengths and mutation parameters up to a few hundreds or even thousands of sequences.

One major complication is if we want to allow variation in mutation model parameters such as the "+Γ" model we discussed last week. It turns out that these parameters cannot be conveniently included in the optimization. In the software PAML (Phylogenetic Analysis by Maximum Likelihood, Yang 2000) there is a discrete approximation (of *Gamma*) that works reasonably for several hundred samples with known topology.

The most well known software for phylogenetic estimation with ML is Phylip by Felsenstein, which you will be playing with in the HW.

Since the tree space is super exponential in n ($\approx n!2^n$), we can not do a full enumeration over topologies even for small sizes, and heuristics typically work up to a few dozen samples in finding topologies. The parsimony approach supposedly works reasonably well even for thousands of samples (with the caveat that parsimony might not be the right measure, of course).

Molecular clock effect on parameters

The approach we described above has a parameter s_{ij} on each branch that is its length. With the molecular clock assumption and a root, the branch lengths are constrained so all paths from the root to a leaf are of the same length. It is easy to see that this can be expressed by having a parameter for each internal node in the graph ($n-1$ in a rooted binary tree) instead of each arc in the graph ($2n-2$ in a binary tree), so a factor of about 2 in the number of these parameters.