More statistical stuff

CS 394C

Feb 6, 2012

Today

- Review of material from Jan 31
- Calculating pattern probabilities
- Why maximum parsimony and UPGMA are not statistically consistent
- Maximum Likelihood
- Phylogenetic Estimation Software

Simplest model of binary character evolution: **Cavender-Farris**

- For each edge **e**, there is a probability **p(e)** of the property "changing state" (going from 0 to 1, or vice-versa), with $0<p(e)<0.5$ (to ensure that CF trees are identifiable).
- Every position evolves under the same process, independently of the others.

Cavender-Farris pattern probabilities

- Let x and y denote nodes in the tree, and p_{xy} denote the probability that x and y exhibit different states.
- Theorem: Let p_i be the substitution probability for edge e_i, and let x and y be connected by path $e_1e_2e_3\ldots e_k$. Then

$$1-2p_{xy} = (1-2p_1)(1-2p_2)\ldots(1-2p_k)$$

And then take logarithms

- The theorem gave us:
  $$1-2p_{xy} = (1-2p_1)(1-2p_2)\ldots(1-2p_k)$$

- If we take logarithms, we obtain
  $$\ln(1-2p_{xy}) = \ln(1-2p_1) + \ln(1-2p_2)+\ldots+\ln(1-2p_k)$$

- Since these probabilities lie between 0 and 0.5, these logarithms are all negative. So let's multiply by -1 to get positive numbers.

Branch Lengths

- Let w(e) = -1/2 ln(1-2p(e)), where p(e) is the probability of change on edge e.
- By the previous theorem, D_{xy} = -1/2 ln(1-2p_{xy}) defines an additive matrix.
- We can estimate D_{xy} using

  d_{xy}= -1/2 ln(1-2H(x,y)/k),

  where H(x,y) is the Hamming distance and k is the sequence length
- This is a statistically consistent distance estimator for the CF model.

CF model, again

- Instead of defining a CF model tree using substitution probabilities, p(e), we give w(e) = expected number of times the site will change on edge e, under a Poisson random process.
- In this case, p(e) is the probability of an odd number of changes on edge e.
- It is not that hard to show that

$$w(e) = -1/2 \ln(1-2p(e))$$

Rates-across-sites

- Most models allow for sites to vary somewhat, but restrict this variability to “rates” of evolution (so some sites evolve faster, some evolve slower)
- These rates are used to scale the branch lengths w(e) up or down (they do NOT scale the substitution probabilities)
- Typically these rates are drawn from some nice distribution (like a gamma distribution), to ensure identifiability of the tree from the data

DNA substitution models

- Every edge has a substitution probability
- The model also allows 4x4 substitution matrices on the edges:
  - Simplest model: Jukes-Cantor (JC) assumes that all substitutions are equiprobable
  - General Time Reversible (GTR) Model: one 4x4 substitution matrix for all edges
  - General Markov (GM) model: different 4x4 matrices allowed on each edge

Jukes-Cantor DNA model

- Character states are A,C,T,G (nucleotides).
- All substitutions have equal probability.
- On each edge **e**, there is a value **p(e)** indicating the probability of change from one nucleotide to another on the edge, with $0<p(e)<0.75$ (to ensure that JC trees are identifiable).
- The state (nucleotide) at the root is random (all nucleotides occur with equal probability).
- All the positions in the sequence evolve identically and independently.

Tree Estimation

- Distance-based methods can be statistically consistent, if (a) statistically consistent distance estimator is used, and (b) the tree estimation technique has some error tolerance
- Maximum parsimony can be statistically inconsistent
- UPGMA can be statistically inconsistent

Maximum Parsimony

- Parsimony-informative sites on 4-leaf trees
- Parsimony informative sites more generally: at least two “big states”
- The Felsenstein Zone tree

Computing the probability of the data

- Given a model tree (with all the parameters set) and character data at the leaves, you can compute the probability of the data.
- Small trees can be done by hand.
- Large examples are computationally intensive - but still **polynomial** time (using an algorithmic trick).

Cavender-Farris model calculations

- Consider an unrooted tree with topology ((a,b),(c,d)) with p(e)=0.1 for all edges.

- What is the probability of all leaves having state 0?

We show the brute-force technique.

Brute-force calculation

Let E and F be the two internal nodes in the tree ((A,B),(C,D)).

Then Pr(A=B=C=D=0) =

- Pr(A=B=C=D=0|E=F=0) +
- Pr(A=B=C=D=0|E=1, F=0) +
- Pr(A=B=C=D=0|E=0, F=1) +
- Pr(A=B=C=D=0|E=F=1)

The notation “Pr(X|Y)” denotes the probability of X given Y.

Calculation, cont.

Technique:

- Set one leaf to be the root
- Set the internal nodes to have some specific assignment of states (e.g., all 1)
- Compute the probability of that specific pattern
- Add up all the values you get, across all the ways of assigning states to internal nodes

Calculation, cont.

Calculating Pr(A=B=C=D=0|E=F=0)

- There are 5 edges, and thus no change on any edge.
- Since p(e)=0.1, then the probability of no change is 0.9. So the probability of this pattern, given that the root is a particular leaf and has state 0, is $(0.9)^5$.
- Then we multiply by 0.5 (the probability of the root A having state 0).
- So the probability is $(0.5)x(0.9)^5$.

Calculation, cont.

Calculating Pr(A=B=C=D=0|E=F=1)

- There is a change on every edge except the internal edge.
- Since p(e)=0.1, the probability of no change is 0.9. So the probability of this pattern, given that the root is a particular leaf and has state 0, is $(0.1)^4(0.9)$.
- Then we multiply by 0.5 (the probability of the root A having state 0).
- So the probability is $(0.5)x(0.1)^4x(0.9)$.

Calculating Pattern Probabilities

- The brute-force calculation uses exponential time
- Dynamic Programming makes it possible to do this in polynomial time

Fixed-tree maximum parsimony

- Input: tree topology T with leaves labelled by sequences of same length
- Output: optimal assignment of sequences to internal nodes to minimize the parsimony score

Recall DP algorithm for maximum parsimony

Cost(v,a) = min cost of the subtree T_v, given that we label v by letter a.

Questions:

- How to initialize?
- How to order the subproblems?
- Where is the answer?
- What is the running time?

DP algorithm for calculating CF pattern probabilities

Calculate: $W(v,a) = \text{Prob}(S_v|\text{label}(v)=a)$, the probability of the sequences at the leaves of the subtree T_v, given label(v) = a.

Questions:

- How to initialize?
- How to order the subproblems?
- Where is the answer?
- What is the running time?

DP algorithm, continued

- Note: the input is a CF model tree, so substitution probabilities on the edges are part of the input
- The running time is polynomial
- You can root the tree on any edge

CF maximum likelihood for fixed tree

- Input: tree topology T and sequences at the leaves of T
- Output: substitution probabilities θ so that Pr(S|(T,θ)) is maximized

The DP algorithm does *not* solve this

Maximum Likelihood

- Input: sequence data S
- Output: the model tree (tree T and substitution parameters θ) such that Pr(S|(T,θ)) is maximized.

NP-hard.

Important in practice.

Good heuristics!

But what does it mean?

Maximum likelihood under Cavender-Farris

- Given a set S of binary sequences, find the Cavender-Farris model tree (tree topology and edge parameters) that maximizes the probability of producing the input data S.

ML, if solved exactly, is statistically consistent under Cavender-Farris (and under the DNA sequence models, and more complex models as well).

The problem is that **ML is hard to solve.**

“Solving ML”

- Technique 1: compute the probability of the data under each model tree, and return the best solution.
- Problem: Exponentially many trees on n sequences, and uncountably infinite number of ways of setting the parameters on each of these trees!

“Solving ML”

- Technique 2: For each of the tree topologies, find the best parameter settings.
- Problem: Exponentially many trees on n sequences, and calculating the best setting of the parameters on any given tree is hard!

Even so, there are hill-climbing heuristics for both of these calculations (finding parameter settings, and finding trees).

Bayesian MCMC analyses

- Algorithm is a **random walk** through space of all possible model trees (trees with substitution matrices on edges, etc.).
- From your current model tree, you perturb the tree topology and numerical parameters to obtain a new model tree.
- Compute the probability of the data (character states at the leaves) for the new model tree.
  - If the probability increases, accept the new model tree.
  - If the probability is lower, then accept with some probability (that depends upon the algorithm design and the new probability).
- Run for a long time…

Bayesian MCMC estimation

After the random walk has been run for a very long time…

- Gather a random sample of the trees you visit
- Return:
  - Statistics about the random sample (e.g., how many trees have a particular bipartition), OR
  - Consensus tree of the random sample, OR
  - The tree that is visited most frequently

Bayesian methods, if run *long enough*, are statistically consistent methods (the tree that appears the most often will be the true tree with high probability).

MrBayes is standard software for Bayesian analyses in biology.

Summary for CF (and GTR)

- Maximum Likelihood is statistically consistent if solved exactly
- Bayesian MCMC methods, if run long enough
- Distance-based methods (like Neighbor Joining and the Naïve Quartet Method)

But not maximum parsimony, not maximum compatibility, and not UPGMA

Software

- MrBayes is the most popular Bayesian methods (but there are others)
- RAxML is the most popular software for ML estimation on large datasets, but other software may be almost as accurate and faster (in particular FastTree)
- Protein sequence data presents additional challenges, due to model selection
- Issues: running time, memory, and models…

Phylogeny estimation statistical issues

- Is the phylogeny estimation method statistically consistent under the given model?
- How much data does the method need need to produce a correct tree?
- Is the method robust to model violations?
- *Is the character evolution model reasonable?*