Lecture 17
Molecular Evolution and Phylogenetics

Challenges in Computational Biology

Module 4: Population / Evolution / Phylogeny

- L15/16: Association mapping for disease and molecular traits
  - Statistical genetics: disease mapping in populations (Mark Daly)
  - Quantitative traits and molecular variation: eQTLs, cQTLs
- L17/18: Phylogenetics / Phylogenomics
  - Phylogenetics: Evolutionary models, Tree building, Phylo inference
  - Phylogenomics: gene/species trees, coalescent models, populations
- L19/20: Human history, Missing heritability
  - Measuring natural selection in human populations
  - The missing heritability in genome-wide associations
- And done! Last pset Nov 11 (no lab), In-class quiz on Nov 20
  - No lab 4! Then entire focus shifts to projects, Thanksgiving, Frontiers
Extinctions part of life

Phylogenetics

General Problem:
Infer complete ancestry of a set of ‘objects’ based on knowledge of their ‘traits’

‘Objects’ can be: Species, Genes, Cell types, Diseases, Cancers, Languages, Faiths, Cars, Architectural Styles

‘Traits’ can be: Morphological, molecular, gene expression, TF binding, motifs, words…

Historical record varies: Fossils, imprints, timing of geological events, ‘living fossils’, sequencing of extinct species, paintings, stories.

Today: Phylogenies using only extant species data ➔ gene trees (paralog / ortholog / homolog trees)

Goals for today: Phylogenetics

- Basics of phylogeny: Introduction and definitions
  - Characters, traits, nodes, branches, lineages, topology, lengths
  - Gene trees, species trees, cladograms, chronograms, phylograms

1. From alignments to distances: Modeling sequence evolution
   - Turning pairwise sequence alignment data into pairwise distances
   - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy

2. From distances to trees: Tree-building algorithms
   - Tree types: Ultrametric, Additive, General Distances
   - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
   - Optimality: Least-squared error, minimum evolution (require search)

3. From alignments to trees: Alignment scoring given a tree
   - Parsimony: greedy (union/intersection) vs. DP (summing cost)
   - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)

4. Tree exploration: Markov Chain Monte Carlo tree search
   - Proposal operations: NNI neighbor interchange, SPR prune/regraft
   - MCMC: Metropolis-Hastings, sample posterior P(B,T|D) without P(D)

Introduction: Basics and Definitions

Characters, traits, gene/species trees

Inferring Phylogenies: Traits and Characters

Trees can be inferred by several criteria:
- Traditional traits: Morphology data
- Modern traits: Molecular data

\[
\begin{align*}
\text{Kangaroo} & : \text{ACAGTGACGCCCCAAACGT} \\
\text{Elephant} & : \text{ACAGTGACGCATCAAAACGT} \\
\text{Dog} & : \text{CCTGTGACGTAACAAACGA} \\
\text{Mouse} & : \text{CCTGTGACGTAGCAAACGA} \\
\text{Human} & : \text{CCTGTGACGTAGCAAACGA}
\end{align*}
\]

From physiological traits to DNA characters

- Traditional phylogenetics
  - Building species trees
  - Small number of traits
    - Hoofs, nails, teeth, horns
  - Well-behaved traits, each arose once
    - Parsimony principle, Occam’s razor

- Modern phylogenetics
  - Building gene trees and species trees
  - Very large number of traits
    - Every DNA base and every protein residue
  - Frequently ill-behaved traits
    - Back-mutations are frequent (convergent evolution)
    - Small number of letters, arise many times independently
Common Phylogenetic Tree Terminology

- Ancestral Node or ROOT of the Tree
- Internal Nodes or Divergence Points (represent hypothetical ancestors of the taxa)
- Branches or Lineages
- Terminal Nodes

Represent the TAXA (genes, populations, species, etc.) used to infer the phylogeny

Three types of trees

- Cladogram
  - Topology only
  - Taxon A, Taxon B, Taxon C, Taxon D

- Chronogram
  - Topology + Divergence times
  - t1, t2, t3

- Phylogram
  - Topology + Divergence times + Divergence rates

Inferring a tree from nucleotides/peptides

- Molecular phylogenetic methods
- Sequence data:
  - Nucleotide alignments
  - Peptide alignments
- Evolutionary history represented as a binary tree

Two basic approaches for phylogenetic inference

- Distance based
  - From alignments to distances: Modeling sequence evolution
  - From distances to trees: Tree-building algorithms

- Character based
  - From alignments to distances: Modeling evolutionary rates

Goals for today: Phylogenetics

- Basics of phylogeny: Introduction and definitions
  - Characters, traits, nodes, branches, lineages, topology, lengths
  - Gene trees, species trees, cladograms, chronograms, phylograms

1. From alignments to distances: Modeling sequence evolution
   - Turning pairwise sequence alignment data into pairwise distances
   - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy

2. From distances to trees: Tree-building algorithms
   - Tree types: Ultrametric, Additive, General Distances
   - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
   - Optimality: Least-squared error, minimum evolution (require search)

3. From alignments to trees: Alignment scoring given a tree
   - Parsimony: greedy (union/intersection) vs. DP (summing cost)
   - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)

4. Tree exploration: Markov Chain Monte Carlo tree search
   - Proposal operations: NNI neighbor interchange, SPR prune/regraft
   - MCMC: Metropolis-Hastings, sample posterior P(B,T|D) without P(D)
### Measuring evolutionary rates

- **Nucleotide divergence**
  - Uniform rate. Overall percent identity.
- **Transitions and transversions**
- **Synonymous and non-synonymous substitutions**
  - Ka/Ks rates. Amino-acid changing substitutions

\[ N_{\text{actual mutations}} > N_{\text{observed substitutions}} \]
- Some fraction of “conserved” positions mutated twice

\[
\begin{align*}
A & \quad C & \quad G & \quad T \\
A & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 \\
C & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 \\
G & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 \\
T & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 & \quad 0.1 \\
\end{align*}
\]

### Evolving a nucleotide under random model

- **At time step 0,** start with letter **A**
- **At time step 1:**
  - Remain **A** with probability 0.7
  - Change to **C,G,T** with prob. 0.1 each
- **At time step 2:**
  - In state **A** with probability 0.52
    - Remain **A** with probability 0.7 * 0.7
    - Go back to **A** from **C,G,T** with 0.1*0.1 each
  - In states **C,G,T** with prob. 0.16 each

<table>
<thead>
<tr>
<th>Time</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>t=1</td>
<td>1</td>
<td>0.7</td>
<td>0.52</td>
<td>0.412</td>
</tr>
<tr>
<td>t=2</td>
<td>0</td>
<td>0.1</td>
<td>0.16</td>
<td>0.196</td>
</tr>
<tr>
<td>t=3</td>
<td>0</td>
<td>0.1</td>
<td>0.16</td>
<td>0.196</td>
</tr>
<tr>
<td>t=4</td>
<td>0</td>
<td>0.1</td>
<td>0.16</td>
<td>0.196</td>
</tr>
<tr>
<td>t=5</td>
<td>0</td>
<td>0.1</td>
<td>0.16</td>
<td>0.196</td>
</tr>
</tbody>
</table>

### Modeling Nucleotide Evolution

- **During infinitesimal time \( \Delta t \),** there is not enough time for two substitutions to happen on the same nucleotide
- So we can estimate \( P(x \mid y, \Delta t) \), for \( x, y \in \{A, C, G, T\} \)
- Then let
  \[
  S(\Delta t) = \begin{pmatrix}
  P(A \mid A, \Delta t) & \cdots & P(A \mid T, \Delta t) \\
  \vdots & \ddots & \vdots \\
  P(T \mid A, \Delta t) & \cdots & P(T \mid T, \Delta t)
  \end{pmatrix}
  \]

### Modeling Nucleotide Evolution

- **Jukes-Cantor:**
  - Transitions: A/G, C/T
  - Transversions: A/T, A/C, G/T, C/G
  - Transitions (rate \( \alpha \)) are much more likely than transversions (rate \( \beta \))

\[
S(t) = \begin{pmatrix}
1 - 3\alpha & \alpha & \alpha & \alpha \\
\alpha & 1 - 3\alpha & \alpha & \alpha \\
\alpha & \alpha & 1 - 3\alpha & \alpha \\
\alpha & \alpha & \alpha & 1 - 3\alpha
\end{pmatrix}
\]

For short time \( \varepsilon \), \( S(\varepsilon) = \frac{1}{4} (1 - 3\varepsilon) + \frac{3}{4} \varepsilon \)

\[
\begin{align*}
\alpha & = 1 - 2s(t) - u(t) \\
\beta & = 1 - 3\alpha
\end{align*}
\]

### Modeling Nucleotide Evolution

- **Kimura:**
  - Transitions: A/G, C/T
  - Transversions: A/T, A/C, G/T, C/G
  - Transitions (rate \( \alpha \)) are much more likely than transversions (rate \( \beta \))

\[
S(t) = \begin{pmatrix}
\alpha & \alpha & \alpha & \alpha \\
\alpha & \alpha & \alpha & \alpha \\
\alpha & \alpha & \alpha & \alpha \\
\alpha & \alpha & \alpha & \alpha
\end{pmatrix}
\]

Where
\[
\begin{align*}
\alpha & = 1 - 2s(t) - u(t) \\
\beta & = 1 - 3\alpha
\end{align*}
\]
Distance between two sequences

Given (well-aligned portion of) sequences \( x^i, x^j \),

Define
\[
d_{ij} = \text{distance between the two sequences}
\]

One possible definition:
\[
d_{ij} = \text{fraction } f \text{ of sites } u \text{ where } x^i[u] \neq x^j[u]
\]

Better model (Jukes-Cantor):
\[
d_{ij} = -\frac{4}{3} \log(1 - \frac{4f}{3})
\]

Better model (Jukes-Cantor):
\[
r(t) = \frac{1}{4} \left(1 + 3e^{-4t}\right)
\]
\[
s(t) = \frac{1}{4} \left(1 - e^{-4t}\right)
\]

Observed \( F = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7] \)
Actual \( D = [0.11, 0.23, 0.38, 0.57, 0.82, 1.21, 2.03] \)

Many nucleotide models have been developed
Varying levels of complexity (parameters)

\[
\begin{array}{c|c|c}
\text{4 substitution types} & \text{3 substitution types} & \text{1 substitution type} \\
\hline
\text{GTR (general time reversible)} & \text{TM (Tamura-Nei)} & \text{K81 (Kimura 2 parameter)} \\
\text{2 transitions} & \text{1 transition} & \text{1 substitution type} \\
\text{1 transversion} & \text{1 transversion} & \\
\text{1 transversion} & \\
\hline
\end{array}
\]

Models also exist for peptides and codons

Goals for today: Phylogenetics
- Basics of phylogeny: Introduction and definitions
  - Characters, traits, nodes, branches, lineages, topology, lengths
  - Gene trees, species trees, cladograms, chronograms, phylogenies
- From alignments to distances: Modeling sequence evolution
  - Turning pairwise sequence alignment data into pairwise distances
  - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy
- From distances to trees: Tree-building algorithms
  - Tree types: Ultrametric, Additive, General Distances
  - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
  - Optimality: Least-squared error, minimum evolution (require search)
- From alignments to trees: Alignment scoring given a tree
  - Parsimony: greedy (union/intersection) vs. DP (summing cost)
  - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)
- Tree exploration: Markov Chain Monte Carlo tree search
  - Proposal operations: NNI neighbor interchange, SPR prune/regraft
  - MCMC: Metropolis-Hastings, sample posterior P(B,T|D) without P(D)

Distance matrix ↔ Phylogenetic tree

<table>
<thead>
<tr>
<th></th>
<th>Hum</th>
<th>Mou</th>
<th>Rat</th>
<th>Dog</th>
<th>Cat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>Mouse</td>
<td>h.y.m</td>
<td>0</td>
<td>3</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Rat</td>
<td>h.y.r</td>
<td>m.r</td>
<td>0</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>Dog</td>
<td>h.z.x.d</td>
<td>m.y.z.x.d</td>
<td>r.y.z.x.d</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Cat</td>
<td>h.z.x.c</td>
<td>m.y.z.x.c</td>
<td>r.y.z.x.c</td>
<td>d.c</td>
<td>0</td>
</tr>
</tbody>
</table>

Tree implies a distance matrix
\( M_{ij} \)

Goal:
Minimize discrepancy between observed distances and tree-based distances

Distance matrix ↔ Phylogenetic tree

\[
\begin{align*}
\text{Human} & \quad h \\
\text{Mouse} & \quad m \\
\text{Rat} & \quad r \\
\text{Dog} & \quad y \\
\text{Cat} & \quad z
\end{align*}
\]

Map distances \( D_{ij} \) to a tree

\[
\min \sum_i (D_{ij} - M_{ij})^2
\]

Distances: (a) Ultrametric distances
- For all points \( i, j, k \)
  - two distances are equal and third is smaller
  \[ d(i,j) = d(i,k) = d(j,k) \]
  \[ a + a = a + b = a + b \]

Where \( a \leq b \)

Result:
- All paths from leaves are equidistant to the root
- Rooted tree with uniform rates of evolution
Distances: (b) Additive distances

- All distances satisfy the four-point condition
  - Any quartet can be labeled i,j,k,l such that:
    \[ d(i,j) + d(k,l) \leq d(i,k) + d(j,l) = d(i,l) + d(j,k) \]
    \[ (a+b)+(c+d) \leq (a+m+c)+(b+m+d) = (a+m+d)+(b+m+c) \]

- Result:
  - All pairwise distances obtained by traversing a tree

Algorithms: (a) UPGMA (aka Hierarchical Clustering)

(Undirected Pair Group Method with Arithmetic mean)

**Initialization:**
- Assign each \( x_i \) into its own cluster \( C_i \)
- Define one leaf per sequence, height 0

**Iteration:**
- Find two clusters \( C_i, C_j \) s.t. \( d_{ij} \) is min
- Let \( C_k = C_i \cup C_j \)
- Define node connecting \( C_i, C_j \)
- & place it at height \( d_{ij}/2 \)
- Delete \( C_i, C_j \)

**Termination:**
- When two clusters \( i, j \) remain, place root at height \( d_{ij}/2 \)

Ultrametric Distances & UPGMA

UPGMA is guaranteed to build the correct tree if distance is ultrametric

**Proof:**
1. The tree topology is unique, given that the tree is binary
2. UPGMA constructs a tree obeying the pairwise distances

Weakness of UPGMA

**Molecular clock assumption:**
- Implies time is constant for all species

However, certain species (e.g., mouse, rat) evolve much faster

Example where UPGMA messes up:

Algorithms: (b) Neighbor-Joining

- Guaranteed to produce the correct tree if distance is additive
- May produce a good tree even when distance is not additive

**Step 1:** Finding neighboring leaves

Define

\[ D_i = d_i - (r_i + r) \]

Where

\[ r_i = \frac{1}{|L| - 2} \sum_{k} d_{ik} \]

Claim: The above "magic trick" ensures that \( D_i \) is minimal iff \( i, j \) are neighbors

**Proof:** Beyond the scope of this lecture (Durbin book, p. 189)
**Algorithm: Neighbor-joining**

**Initialization:**
- Define $T$ to be the set of leaf nodes, one per sequence
  - Let $L = T$

**Iteration:**
- Pick $i, j$ s.t. $D_{ij}$ is minimal
  - Define a new node $k$, and set $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$ for all $m \in L$
  - Add $k$ to $T$, with edges of lengths $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$
  - Remove $i, j$ from $L$
  - Add $k$ to $L$

**Termination:**
- When $L$ consists of two nodes, $i, j$, and the edge between them of length $d_{ij}$

---

**Distance matrix ↔ Phylogenetic tree**

<table>
<thead>
<tr>
<th></th>
<th>Hum</th>
<th>Mou</th>
<th>Rat</th>
<th>Dog</th>
<th>Cat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>Mouse</td>
<td>h.y.m</td>
<td>0</td>
<td>3</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Rat</td>
<td>h.y.r</td>
<td>m.r</td>
<td>0</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>Dog</td>
<td>h.z.x.d</td>
<td>m.y.z.x.d</td>
<td>r.y.z.x.d</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Cat</td>
<td>h.z.x.c</td>
<td>m.y.z.x.c</td>
<td>r.y.z.x.c</td>
<td>d.c</td>
<td>0</td>
</tr>
</tbody>
</table>

Tree implies a distance matrix $M_{ij}$

Map distances $D_{ij}$ to a tree

Goal: Minimize discrepancy between observed distances and tree-based distances

---

**Algorithms: (c) Distance-fitting algorithms**

- With distance-based algorithms, we can also aim to directly minimize discrepancy between original distance matrix and tree-based distance matrix

**COMPUTATIONAL METHOD**

**Optimality criterion Clustering algorithm**

- Parsimony
- Maximum Likelihood
- Minimum Evolution
- UPGMA
- Least Squares
- Neighbor-Joining

---

**Goals for today: Phylogenetics**

- **Basics of phylogeny: Introduction and definitions**
  - Characters, traits, nodes, branches, lineages, topology, lengths
  - Gene trees, species trees, cladograms, chronograms, phylograms

1. **From alignments to distances: Modeling sequence evolution**
   - Turning pairwise sequence alignment data into pairwise distances
   - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy

2. **From distances to trees: Tree-building algorithms**
   - Tree types: Ultrametric, Additive, General Distances
   - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
   - Optimality: Least-squared error, minimum evolution (require search)

3. **From alignments to trees: Alignment scoring given a tree**
   - Parsimony: greedy (union/intersection) vs. DP (summing cost)
   - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)

4. **Tree exploration: Markov Chain Monte Carlo tree search**
   - Proposal operations: NNI neighbor interchange, SPR prune/regraft
   - MCMC: Metropolis-Hastings, sample posterior $P(B,T|D)$ without $P(D)$

---

**Basic algorithms of phylogenetic methods**

**Distance based**

1. Sequence alignment
2. Pair-wise distance matrix
3. Output tree

**Character based**

1. Sequence alignment
2. Couplet to tree proposal and scoring
3. Output tree
Character-based phylogenetic inference

- Really about tree **scoring** techniques, not tree finding techniques
  - Couple them with tree proposal and update and you have an algorithm (part 4 of the lecture)
- Two approaches exist, all use same architecture:
  - Minimize events: Parsimony (union/intersection)
  - Probabilistic: Max Likelihood / MAP

**Parsimony scoring (a): Union and intersection**

Given a tree, and an alignment column
Label internal nodes to minimize the number of required substitutions

**Initialization:**
Set cost $C = 0$; $k = 2N - 1$

**Iteration:**
If $k$ is a leaf, set $R_k = \{ x^u \}$
If $k$ is not a leaf,
Let $i, j$ be the daughter nodes;
Set $R_k = R_i \cap R_j$ if intersection is nonempty
Set $R_k = R_i \cup R_j$ and $C += 1$, if intersection is empty

**Termination:**
Minimal cost of tree for column $u$, $C$

---

**Parsimony traceback to find ancestral nucleotides**

Traceback:
1. Choose an arbitrary nucleotide from $R_{2N-1}$ for the root
2. Having chosen nucleotide $r$ for parent $k$,
   - If $r \in R_i$, choose $r$ for daughter $i$
   - Else, choose arbitrary nucleotide from $R_i$

Easy to see that this traceback produces some assignment of cost $C$

---

**Goals for today: Phylogenetics**

- **Basics of phylogeny**: Introduction and definitions
  - Characters, traits, nodes, branches, lineages, topology, lengths
  - Gene trees, species trees, cladograms, chronograms, phylograms
- **From alignments to distances**: Modeling sequence evolution
  - Turning pairwise sequence alignment data into pairwise distances
  - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy
- **From distances to trees**: Tree-building algorithms
  - Tree types: Ultrametric, Additive, General Distances
  - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
  - Optimality: Least-squared error, minimum evolution (require search)
- **From alignments to trees**: Alignment scoring given a tree
  - Parsimony: greedy (union/intersection) vs. DP (summing cost)
    - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)
- **Tree exploration**: Markov Chain Monte Carlo tree search
  - Proposal operations: NNI neighbor interchange, SPR prune/regraft
  - MCMC: Metropolis-Hastings, sample posterior $P(B,T|D)$ without $P(D)$

---

**Scoring (c) Maximum Likelihood & Max-a-Posteriori**

**Input:** Sequence alignment
**Output:** tree with maximum likelihood / max a posteriori prob.
**Search:** Heuristic search for max likelihood tree.

**Maximum Likelihood (ML)**

$$B^*, T^* = \arg \max_{B,T} P(D|B,T)$$

**Maximum a Posteriori (MAP)**

$$B^*, T^* = \arg \max_{B,T} P(B,T|D) = \arg \max_{B,T} P(B,T,D) / P(D) = \arg \max_{B,T} P(D|B,T)P(B,T)$$

$P(D|B,T)$ is the likelihood of data given model
- Use seq evolution model: JC,K2P,HKY.

$P(B,T)$ is a prior on trees/branch lengths
- Use Yule process, Birth-Death process to model

---

**Compute recursively using DP**
`Peeling` algorithm for \( P(D|B,T) \) term

1. Assume sites \( j \) evolve independently
   - Treat each column of the alignment in isolation
2. Assume branch independence, conditioned on parent
   - Expand total joint probability into prod of \( P(x_i|x_{\text{parent}(i)}, t_i) \)
   - Only \( P(x_{2n-1}) \) remains, root prior, background nucl. freq.
3. We know how to compute \( P(x_i|x_{\text{parent}(i)}, t_i) \) for fixed pair
   - Defined by our sequence model (JC, K2P, HKY, etc)
   - Easily calculate for any given assignment of internal nodes
4. As internal node values are not known \( \rightarrow \) marginalize
   - Sum over all possible values of all internal/root nodes
   - Let \( x_{n+1}, \ldots, x_{2n-1} \) represent seqs of \( n-1 \) internal nodes

1. Site evolution over single branch

   **Remember: Jukes-Cantor (JC)**

   - JC is a Continuous-Time Markov Chain (CTMC)
   - Defines instantaneous rates of transition between states (bases)
   - Use JC to define single site evolution:
     \[ P(a = "C" | b = "A", t) = S(t)ba \]

2. Sequence evolution over single branch

   - Assume site independence
     \[ P(x_i | x_{i'}, t_i) = \prod_j P(b=x_{ij} | a=x_{kj}, t_j) \]
   - Use product to define sequence evolution:
     \[ x_9 = "AAACTG" \]
     \[ x_1 = "AAACTG" \]

3. Sequence evolution over entire tree

   - Assume branch independence
     \[ P(x_1, \ldots, x_n, \ldots, x_{2n-1} | T, t) = P(x_{2n-1}) \prod_i P(x_i|x_{\text{parent}(i)}, t_i) \]
   - Assume prior on root sequence, e.g.
     \[ P(x_{2n-1}) = (1/4)^m \text{ for sequence length } m \]
   - Use product and prior to define sequence evolution over tree:
     \[ x_9 = "AAACTG" \]
     \[ x_1 x_2 x_3 x_4 x_5 \]
     \[ x_7 x_6 \]
     \[ x_8 \]
     \[ t_1 \]
     \[ t_2 t_3 \]
     \[ t_6 t_7 \]
     \[ t_4 t_5 \]
     \[ t_8 \]

4. Integrate (marginalize) over hidden ancestral seqs!

   - Notice, all sequences are needed, both internal nodes and leaves
     \[ P(x_1, \ldots, x_n, \ldots, x_{2n-1} | T, t) \]
   - But, only leaves are given: \( x_1, \ldots, x_n \)
   - Therefore, need to marginalize (sum) over unknowns: \( x_{n+1}, \ldots, x_{2n-1} \)
   - This looks expensive!
     \[ P(x_1, \ldots, x_n | T, t) = \sum_{x_{n+1}} \ldots \sum_{x_{2n-1}} P(x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n-1} | T, t) \]
   - Don’t worry, dynamic programming can do it efficiently.

**Basic trick to efficient marginalization**

- \( L(i,j,a) \) is the DP table.
- Each entry contains the probability of seeing the leaf data below node \( i \), given that node \( i \) has base \( a \) at site \( j \).
- The leaves of the table are initialized based on the observed sequence.
- Entries populated in post-order traversal.
- Runtime: \( O(2n \cdot k^2) \)
Use DP to compute argmax \( P(D|B,T) \) efficiently

If we know the branch lengths \( t_{\text{left}} \) & \( t_{\text{right}} \).
And we already have the likelihood tables \( L_j \) & \( L_k \) of left and right subtrees
(for each possible ending character at \( b, c \))

\[
L_i[a] = \sum_{b \in \{ACGT\}} \sum_{c \in \{ACGT\}} \left( P(b|a,t_{\text{left}}) L_{\text{left}}[b] \times P(c|a,t_{\text{right}}) L_{\text{right}}[c] \right) \frac{\text{Prob}(a \rightarrow b)}{\text{Prob}(a \rightarrow c)}
\]

Initialization and Termination

- Characters at the leaves are already known
  - Their likelihood is 1 or 0, indicating the known char
- Fill in internal node likelihood vectors iteratively
- Once we reach the root, multiply by the base freqs
- Maximization over Topologies and Lengths
  ➔ Numerical: gradient descent, Newton’s method

Advantages/disadvantages of ML/MAP methods

- **Advantages:**
  - Inherently statistical and evolutionary model-based.
  - Usually the most ‘consistent’ of the methods available.
  - Used for both character and rate analyses
  - Can be used to infer the sequences of the extinct ancestors.
  - Account for branch-length effects in unbalanced trees.
  - Nucleotide or amino acid sequences, other types of data.

- **Disadvantages:**
  - Not as intuitive as parsimony (e.g. may choose more events if they’re more likely in our probabilistic model)
  - Computationally intense (limits num taxa, sequence length).
  - Like parsimony, can be fooled by high levels of homoplasy.
  - Violations of model assumptions can lead to incorrect trees.

Goals for today: Phylogenetics

- **Basics of phylogeny: Introduction and definitions**
  - Characters, traits, nodes, branches, lineages, topology, lengths
  - Gene trees, species trees, cladograms, chronograms, phylogenograms

1. **From alignments to distances: Modeling sequence evolution**
   - Turning pairwise sequence alignment data into pairwise distances
   - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy

2. **From distances to trees: Tree-building algorithms**
   - Tree types: Ultrametric, Additive, General Distances
   - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
   - Optimality: Least-squared error, minimum evolution (require search)

3. **From alignments to trees: Alignment scoring given a tree**
   - Parsimony: greedy (union/intersection) vs. DP (summing cost)
   - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)

4. **Tree exploration: Markov Chain Monte Carlo tree search**
   - Proposal operations: NNI neighbor interchange, SPR prune/regraft
   - MCMC: Metropolis-Hastings, sample posterior \( P(B,T|D) \) without \( P(D) \)

Basic algorithms of phylogenetic methods

**Distance based**

1. Sequence alignment
2. Pair-wise distance matrix
3. Output tree

**Character based**

3. Sequence alignment
4. Coupled to tree proposal and scoring
5. Output tree

Heuristic tree search in character-based reconstruction

- Propose many trees using rearrangements; “score” each one.
- Score can be: parsimony cost, likelihood, posterior probability.
- Algorithms differ on how “score” is used to guide the tree search
Searching through topologies: operations

Nearest neighbor interchange

Subtree pruning and regrafting

Other methods include: tree bisection and join (TBJ).

Note: NNIs alone can explore entire tree space

Tree space is big:

Number of unrooted topologies: \( N_u = 3 \times 5 \times 7 \times \cdots \times (2n - 5) = (2n - 5)!! \)

Number of rooted topologies: \( N_r = (2n - 3)!! \times N_u = (2n - 3) \times (2n - 5)!! \)

\( N(3) = 1, \ N(4) = 3, \ N(5) = 15, \ N(10) = 2 \times 10^6, \ N(20) = 2 \times 10^{20}, \ N(64) > \text{googol} \)

Markov Chain Monte Carlo (MCMC)

Exploring your state space in a way that makes it a sample from the posterior (well, after skipping burn-in while likelihood is climbing and only taking 1 in 100)

Metropolis-Hastings

- Lets you convert any proposal rule that is a connected one (can reach every state) into one that visits the states according to their posterior
- The rule is:
  - If better: always take it
  - If worse:
    - Compute likelihood ratio \( p = \frac{P(\text{current})}{P(\text{proposed})} \) (if no priors)
    - With \( p \): next becomes current
    - Why ratio: cuz we don’t need to compute evidence \( P(D) \) (see next slide)
  - While any connected proposal rule is okay, in practice proposal strategy is an art.
    - Need good mixing \( \rightarrow \) Big steps good
    - BUT: Don’t fall off cliffs \( \rightarrow \) Small steps good
    - Ensure low rejection rate, i.e. your step size is not sooo big that you typically fall off the cliffs
    - Good rule of thumb is try it, if low rejection rate \( \rightarrow \) good

Metropolis Hastings details

- The ratio you compute in the Metropolis-Hastings rule is actually more complicated than what previous slide describes. Say current state is \((B_1, T_1)\) and proposed next state is \((B_2, T_2)\). Then the ratio you look at is

\[
\alpha = \frac{P(B_2, T_2 | D) P(B_1, T_1 | B_2, T_2)}{P(B_1, T_1 | D) P(B_2, T_2 | B_1, T_1)}
\]

- If \( \alpha > 1 \) then accept \((B_2, T_2)\) no matter what, and if it’s less than 1, accept with probability \( \alpha \). The Metropolis-Hastings algorithm is a generalization of an early algorithm called the Metropolis algorithm that required your proposal rule be symmetric. Namely,

\[
P(B_1, T_1 | B_2, T_2) = P(B_2, T_2 | B_1, T_1)
\]

which means the Metropolis algorithm (no Hastings) is a special case of MH where we have

\[
\alpha = \frac{P(B_2, T_2 | D)}{P(B_1, T_1 | D)}
\]

- Now notice this is just ratio of posterior probabilities. If the prior is flat \( P(B_1, T_1) = P(B_2, T_2) \), then it simplifies even further to a ratio of likelihoods:

\[
\alpha = \frac{P(D | B_2, T_2) P(B_2, T_2)}{P(D | B_1, T_1) P(B_1, T_1)}
\]

- Assumptions needed:

  - symmetric proposal rule and flat prior. Whether or not these assumptions are added, the data probability \( P(D) \) (often called the “evidence”) is still canceled in the ratio; and that’s the true advantage of MCMC, because \( P(D) \) is usually tough to calculate. In fact, MCMC is only used when \( P(D) \) can’t be computed efficiently. If \( P(D) \) can be computed efficiently you would could just do \( P(D | B, T) P(B, T) / P(D) = P(B, T | D) \) and compute posterior probabilities directly.

Tree reliability: Bootstrapping

1. Re-sample alignments:
   - Randomly sample alignment columns with replacement
   - Create many alignments of equal size.
2. Build a phylogenetic tree for each sample
3. Repeat (1) and (2) many times
   - 1000s of times
4. Output summary tree
   - Tree constructed most frequently
   - Consensus tree (even if not most freq)
   - Other options
5. Report observation frequency of each branch
   - Each branch is a binary split

Basic algorithms of phylogenetic methods

Distance based

- From alignments
- To distances
- Pair-wise distance matrix
- Output tree

Character based

- From alignments
- To phylogenies
- Couple to tree proposal and scoring
- Output tree
Algorithms for reconstructing trees

- **UPGMA** (Sokal 1958) [distance]
- **Least Square Error** (Cavalli-Sforza & Edwards 1963) [dist]
- **Maximum Parsimony** (Fitch 1977) [character]
  - PHYLIP
- **Maximum Likelihood** (Felsenstein 1981) [character]
  - RAxML, PHYML, PHYLIP, PAUP
- **Neighbor Joining** (Saitou 1987) [distance]
  - BIONJ, QuickTree, RapidTree, FNJ
- **Maximum a posteriori** (Rannala & Yang 1996) [character]
  - MrBayes, BEAST

  - BEST, CoalMCMC, CoalHMM, PhyloNet, SPIMAP, SYNERGY, PriME-GSR

### Goals for today: Phylogenetics

1. **From alignments to distances: Modeling sequence evolution**
   - Turning pairwise sequence alignment data into pairwise distances
   - Probabilistic models of divergence: Jukes Cantor/Kimura/hierarchy

2. **From distances to trees: Tree-building algorithms**
   - Tree types: Ultrametric, Additive, General Distances
   - Algorithms: UPGMA, Neighbor Joining, guarantees and limitations
   - Optimality: Least-squared error, minimum evolution (require search)

3. **From alignments to trees: Alignment scoring given a tree**
   - Parsimony: greedy (union/intersection) vs. DP (summing cost)
   - ML/MAP (includes back-mutations, lengths): peeling algorithm (DP)

4. **Tree exploration: Markov Chain Monte Carlo tree search**
   - Proposal operations: NNI neighbor interchange, SPR prune/regraft
   - MCMC: Metropolis-Hastings, sample posterior P(B,T|D) without P(D)

---

**Module V: Evolution/phylogeny/populations**

- **Phylogenetics / Phylogenomics**
  - Phylogenetics: Evolutionary models, Tree building, Phylo inference
  - Phylogenomics: gene/species trees, reconciliation, coalescent, pops

- **Population genomics**
  - Learning population history from genetic data (David Reich)
  - Statistical genetics: disease mapping in populations (Mark Daly)
  - Measuring natural selection in human populations (Pardis Sabeti)
  - The missing heritability in genome-wide associations (Yaniv Erlich)

- **And we’re done! Last pset Nov 21st, In-class quiz on Nov 22nd**
  - No lab 4! Then entire focus shifts to projects, Thanksgiving, Frontiers