The Story So Far

- Parsimony: relatively simple and intuitive, but:
  - Requires tree search, which is expensive
  - Throws away a lot of data (informative sites only)
  - No explicit model of sequence change
  - Long-branch attraction

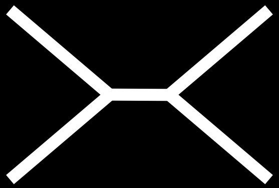
- What other options do we have?

Distance Methods

Overview

```
acca
gcca
gcct
tgca
```

	1	2	3	4
1				
2				
3				
4				



Step 1:
Construct distance matrix

Step 2:
Build tree

1: Sequences to Distances

Can use a model (e.g., PAM) to compute evolutionary distances

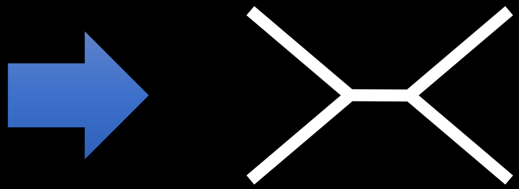
```
acca
gcca
gcct
tgca
```

	1	2	3	4
1				
2				
3				
4				

2. Distances to Trees

- Many different approaches:
  - Iterative/greedy (UPGMA, neighbour-joining)
  - Optimization (Fitch, minimum evolution)

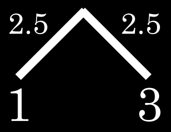
	1	2	3	4
1				
2				
3				
4				



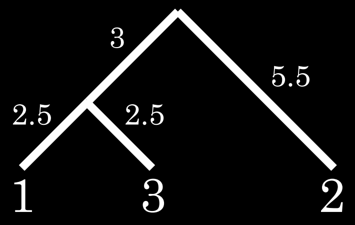
UPGMA again

Unweighted Pair Grouping with Arithmetic Mean

	1	2	3
1			
2	10		
3	5	12	



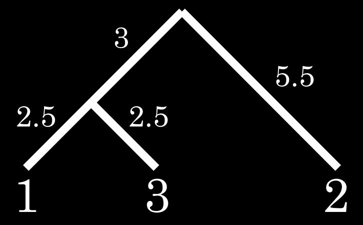
	1+3	2
1+3		
2	11	



distances from the root to all leaves will be EQUAL

A big problem with UPGMA

Distances from the root to all leaves will be EQUAL



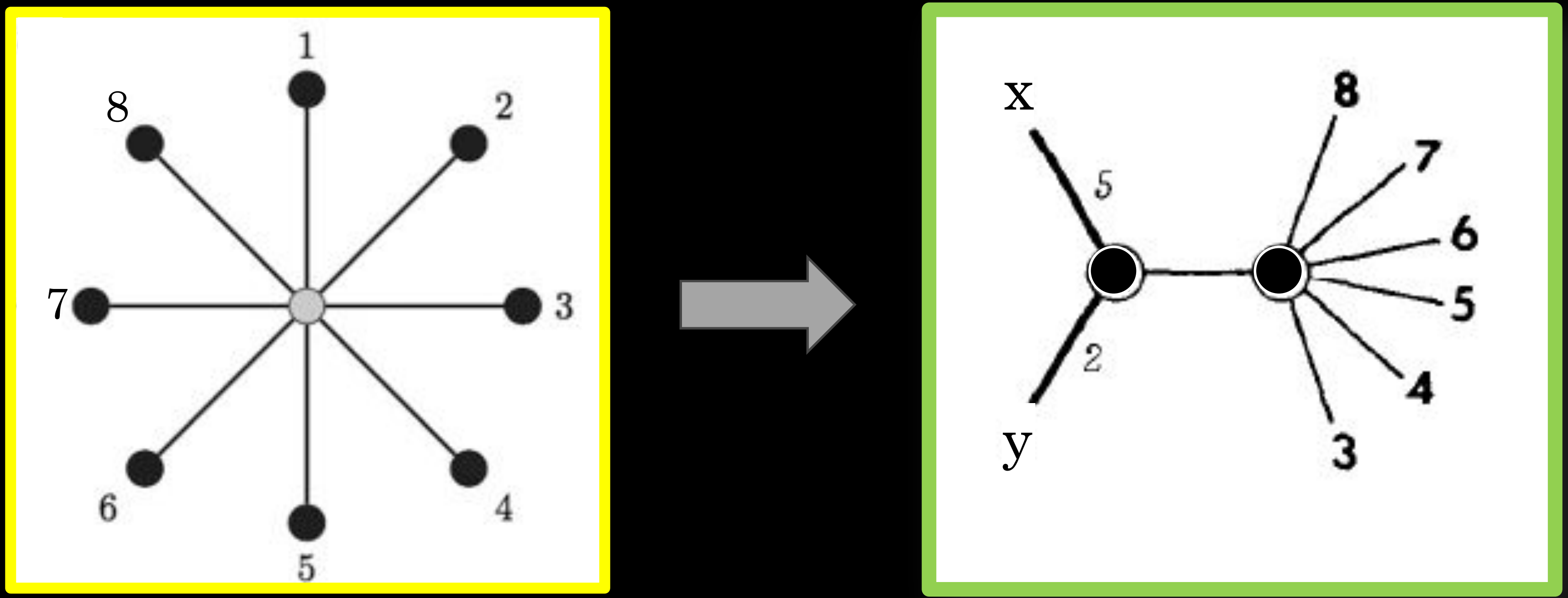
A molecular clock assumption – all sequences evolve at the same rate

Violations of this assumption can really mess up UPGMA, so do not use

Neighbor-joining

Start with a ‘star’ tree

At each iteration, split off the pair of taxa that minimizes the total sum of branch lengths in the tree

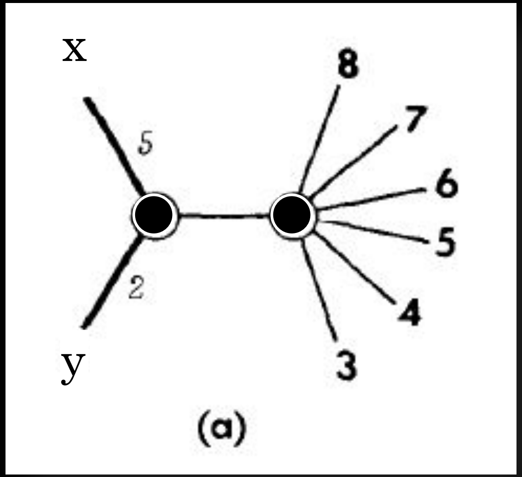


Saitou and Nei (1987) *Molecular Biology and Evolution*
Figure: Telles et al. (2018) *BMC Bioinformatics*

Neighbor-joining

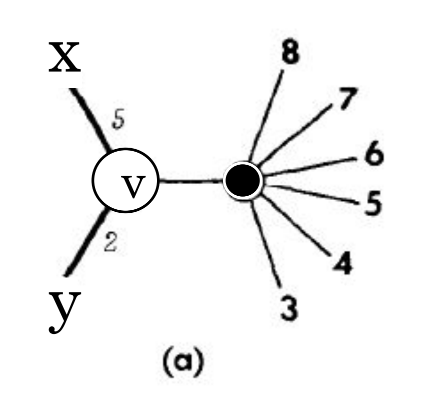
Choose groups x and y to minimize the Q-criterion:

$$\underbrace{\delta(x,y)}_{\text{Distance matrix entry for } (x,y)} - \underbrace{\frac{1}{(n-2)}\sum_z \delta(x,z) - \frac{1}{(n-2)}\sum_z \delta(y,z)}_{\text{Weighted distance from } x \text{ and } y \text{ to each other leaf } z}$$



Figures from Saitou and Nei (1987) *Molecular Biology and Evolution*
Formulas from Bryant (2005) *Journal of Classification*

This splitting creates a new internal node v, and assigns x and y as sisters in the growing tree



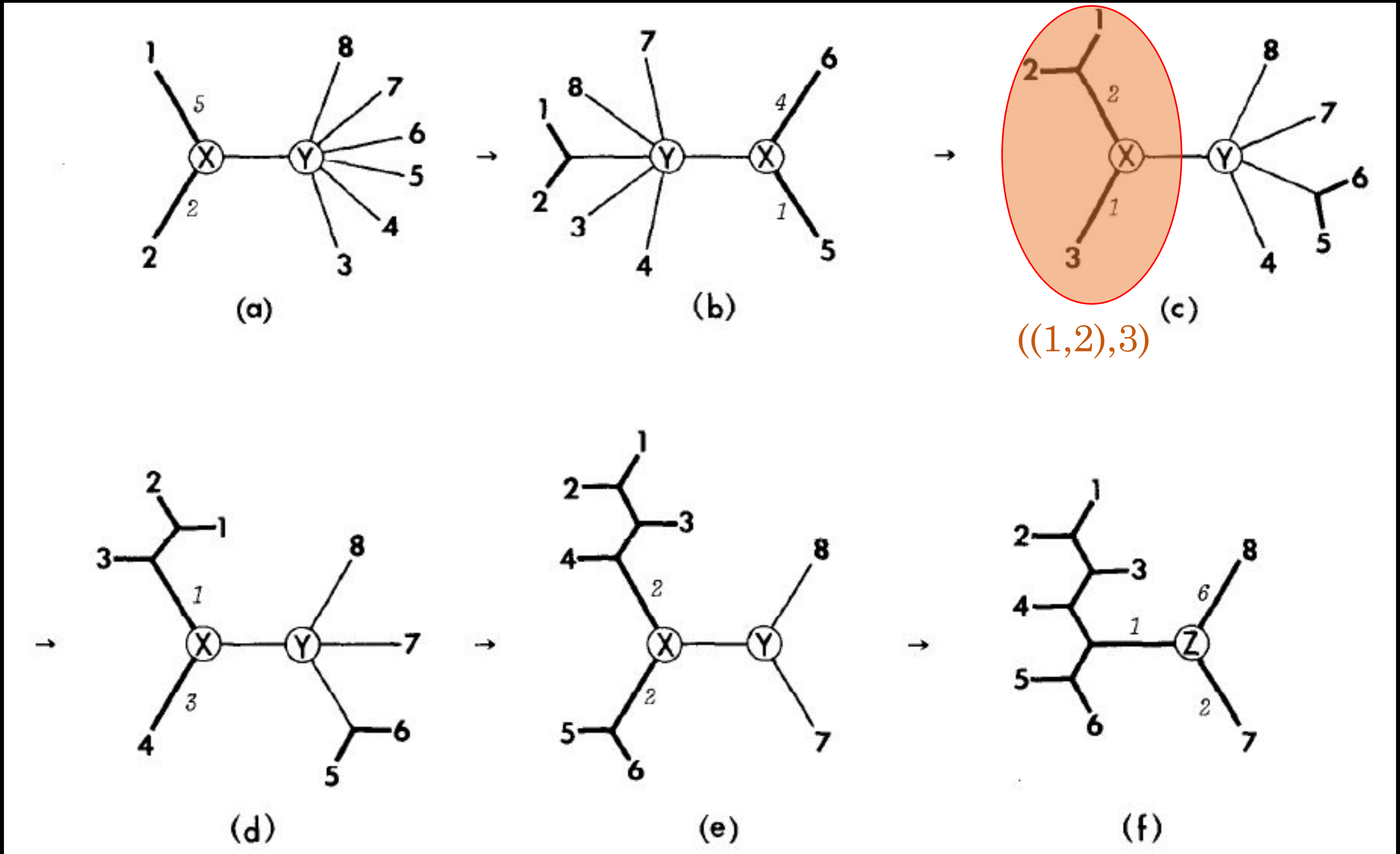
REDUCTION: Recompute distances from all leaves *u* to node *v* to allow subsequent computations of the Q criterion

$$\delta'(u, v_{xy}) = \frac{1}{2}(\delta(u,x) + \delta(u,y) - \delta(x,y))$$

And assign branch lengths *x*-*v* and *y*-*v*

$$b_x = \frac{1}{n-2} \sum_{z \neq x,y} (\delta(x,z) + \delta(x,y) - \delta(y,z))$$

Continue until binary tree is obtained



Figures from Saitou and Nei (1987) *Molecular Biology and Evolution*
Formulas from Bryant (2005) *Journal of Classification*

Neighbor-joining vs. UPGMA

- Neighbor-joining uses a somewhat less intuitive optimality criterion Q

- However, it is still iterative and still fast

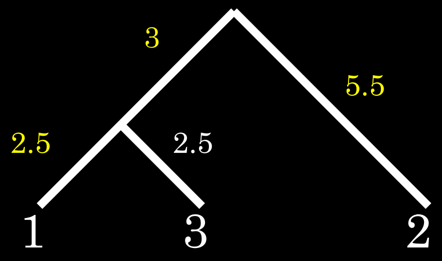
- Another advantage is that it does not assume a molecular clock – branch lengths are assigned based on all distances in the matrix

Not All Distance Methods are Greedy!

- Least Squares methods: Minimize the difference between pairwise taxon distances in the matrix, and the distances in the tree

$d_{1,2} = 10$

	1	2	3
1			
2	10		
3	5	12	



$d_{1,2} = 11$

- These methods give more-accurate trees and branch lengths in general, but require optimization rather than using a greedy approach

Good overview: Bryant and Waddell (1998) *Mol Biol Evol*

Summary:
Advantages of Distance Methods

- Explicit modelling of residue changes
- Can be very FAST – neighbour-joining can build trees with thousands of leaves
- Can be applied to a distance matrix computed from any input data (indeed UPGMA is widely used outside of phylogenetics)

Disadvantages of Distance Methods

- A considerable amount of information is lost when sequence pairs are replaced with a single distance

- Greedy methods may perform poorly for some problems

Conclusion

- Parsimony: Character-based, model-free
  - tree search required
- Distance: Pairwise distances, can use a model
  - Greedy approaches or iterative searches
- Is there a way to use models without collapsing each pair of sequences to a single distance value? yes

Maximum Likelihood

Parsimony is inconsistent

- Statistical consistency: as we add data, a method should converge on the correct answer

- With parsimony, more data can often reinforce an incorrect conclusion

- The long-branch attraction problem is an example of this

Likelihood

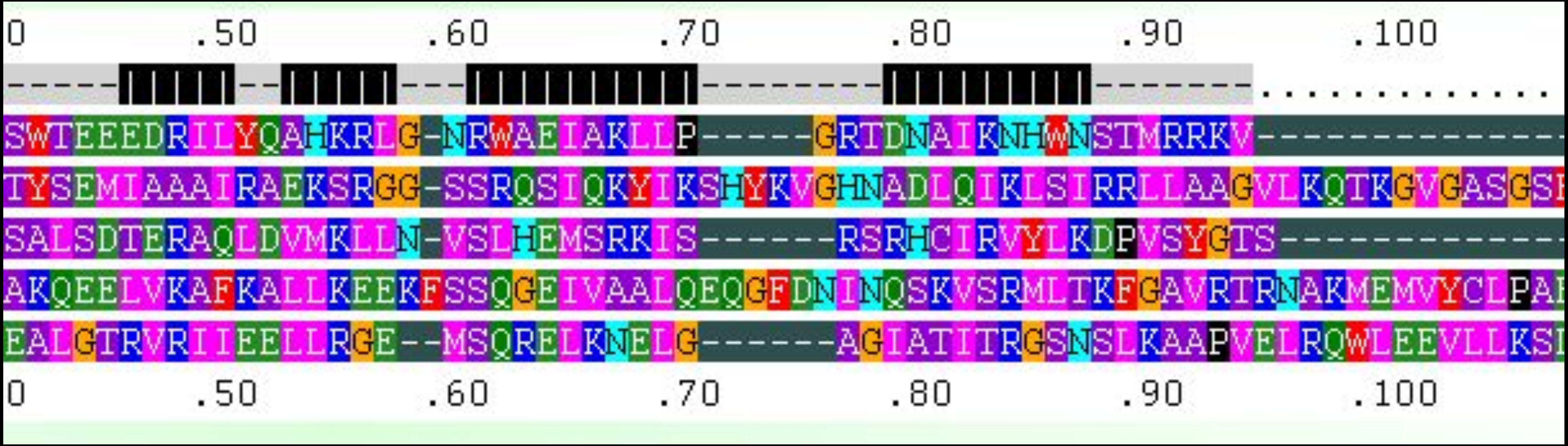
- Likelihood: the probability of observing the data under a given model

- If we can specify a model ϒ of evolution, then we can calculate the likelihood of the data, given ϒ

- The probability of the data, given the model, is the likelihood

What Data?

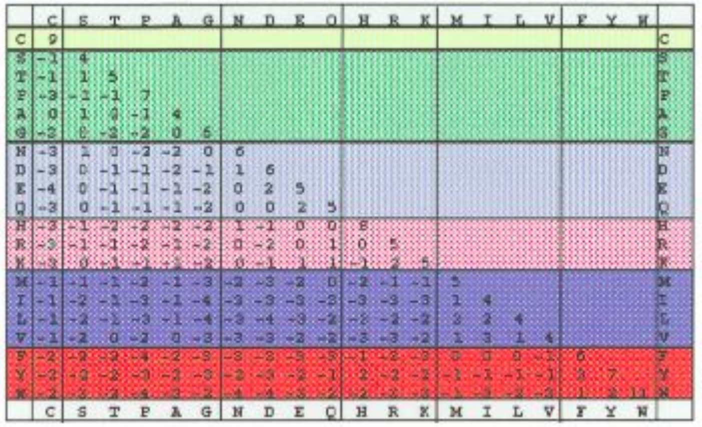
The sequence alignment (our genes or proteins of interest)



```
0       .50       .60       .70       .80       .90       .100
-----|||||--|||||---||||||||||--------|||||||||-------..............
SWTEEEDRILYQAHKRLG-NRWAEIAKLLP-----GRTDNAIKNHWNSTMRRKV--------------
TYSEMIAAAIRAEKSRGG-SSRQSIQKYIKSHYKVGHNADLQIKLSIRRLLAAGVLKQTKGVGASGS
SALSDTERAQLDVMKLLN-VSLHEMSRKIS------RSRHCIRVYLKDPVSYGTS-------------
AKQEELVKAFKALLKEEKFSSQGEIVAALQEQGFDNINQSKVSRMLTKFGAVRTRNAKMEMVYCLPA
EALGTRVRIIEELLRGE--MSQRELKNELG------AGIATITRGSNSLKAAPVELRQWLEEVLLKS
0       .50       .60       .70       .80       .90       .100
```

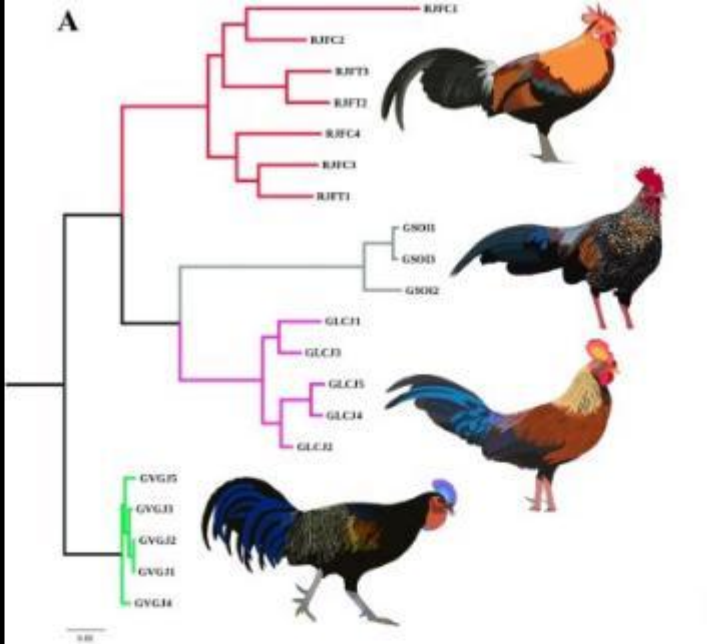
What model?

- A substitution model



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

- Branching order and branch lengths in a tree



Sharma et al. (2023) *Animal Gene*

A Simple Example

Coin-toss likelihoods

One free parameter (probability of ship)
= 1 – (probability of Queen Elizabeth)

We need data (proportion of throws that came up ship)

What is the p(ship)?

Formula

$$L = \underbrace{p(D \mid p(ship) = x)}_{\text{Probability of the data, if the probability of a ship is } x} = \underbrace{\binom{\#trials}{\#ships}}_{\text{Number of ways we can observe } k \text{ ships in } t \text{ trials}} \times \underbrace{p(ship)^{\#ships} \times p(queen)^{\#queens}}_{\text{probability}^{\text{observations}}}$$

Example

Data: 10 throws, 6 ships, 4 Queens

what is L(p(ship) = 0.4)?

$$L = p(D \mid p(ship) = 0.4) = \binom{10}{4} \times 0.4^{6} \times 0.6^{4} = 0.1115$$

what is L(p(ship) = 0.6)?

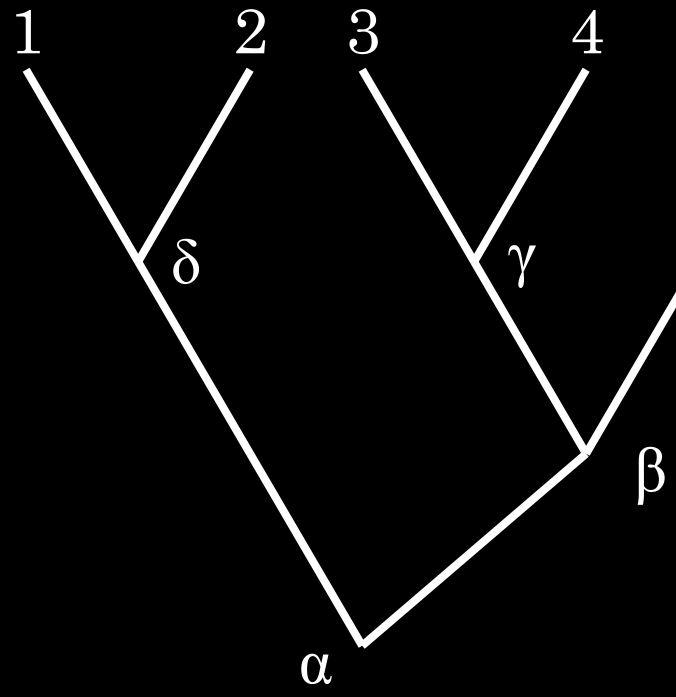
$$L = p(D \mid p(ship) = 0.6) = \binom{10}{6} \times 0.6^{6} \times 0.4^{4} = 0.2508$$

0.6 is the maximum likelihood estimate of p(ship), given these data
(a better explanation than p(ship) = 0.4)

Likelihood of an alignment, given the model Υ

- If we assume independence of each character (alignment column), then we can compute the likelihood separately for each column and multiply the results together
- So column order doesn't really matter (kinda like in the language example)
- People have developed models that consider interactions among sites. But how would you do it?

Computing the likelihood for an alignment column



1, 2, 3, 4, 5 are known states

1	A
2	A
3	C
4	C
5	G

α, β, γ, δ, are internal states in the tree

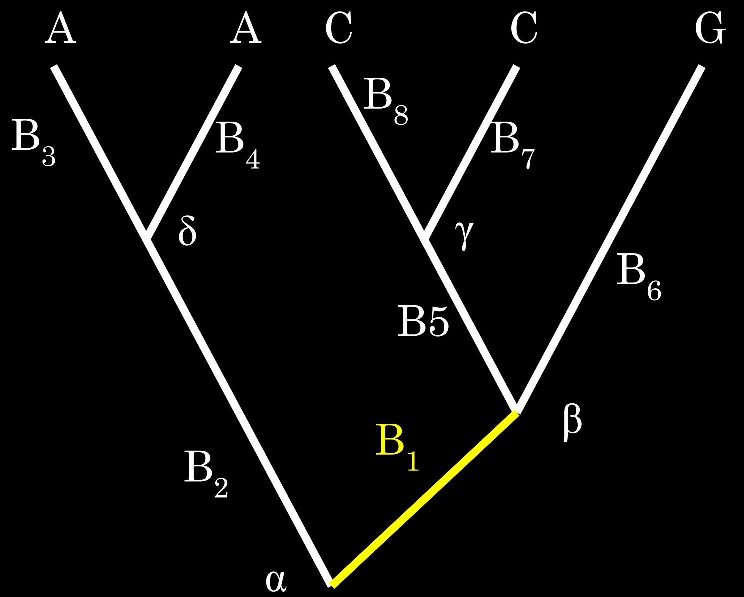
$$P(Data\,|\,T) = \sum_{\alpha} \sum_{\beta} \sum_{\gamma} \sum_{\delta} P(A,A,C,C,G,\alpha,\beta,\gamma,\delta\,|\,T)$$

Huh?

Computing the likelihood for a given column

Sum over all probabilities (4 nucleotides or 20 amino acids) at every internal node

$$\sum_{\alpha} \sum_{\beta} \sum_{\gamma} \sum_{\delta} P(A, A, C, C, G, \alpha, \beta, \gamma, \delta \mid T)$$



$$= P(\alpha = A) \text{ x } P(\beta = A \mid \alpha = A, B_1) \text{ x } \ldots$$

$$+ P(\alpha = C) \text{ x } P(\beta = A \mid \alpha = C, B_1) \text{ x } \ldots$$

…

4^4 terms!

What is $P(\beta = C \mid \alpha = A, B_1)$???

- B_1 is the branch length (in substitutions per site)

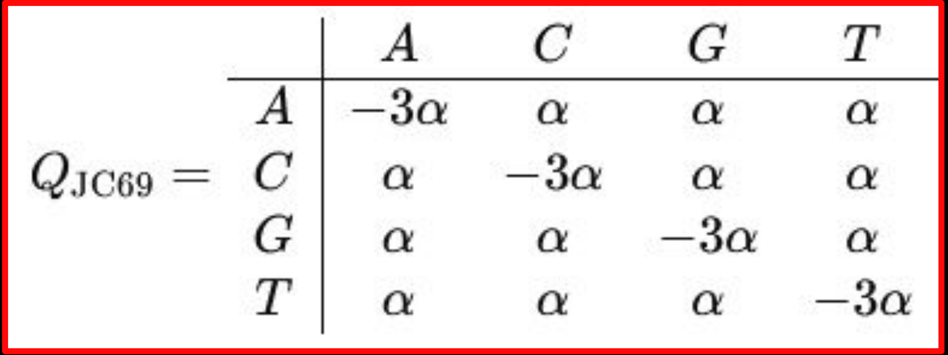
- Our substitution model defines the probability of observing a substitution from A to C over a branch of a given length

- A matrix like PAM needs to be converted into an instantaneous rate matrix Q, which accounts for residue frequencies and rows sum to 0

$$P(\beta = C \mid \alpha = A, B_1) = \left(e^{QB_1}\right)_{A,C}$$

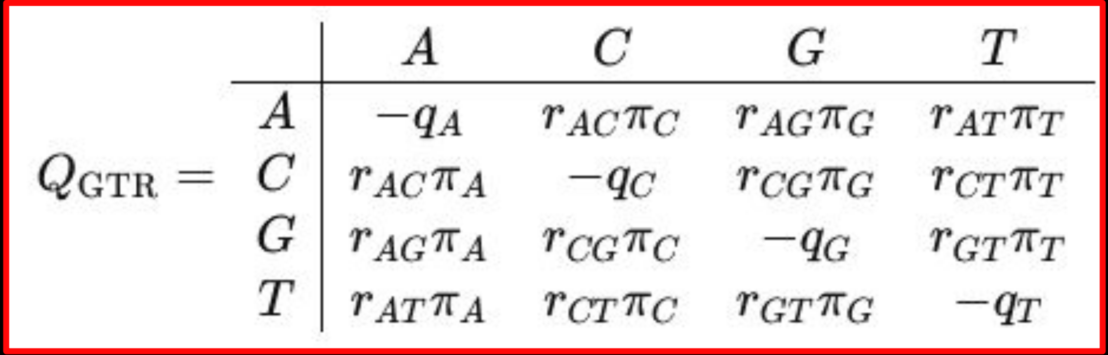
The Instantaneous Rate Matrix

- Off diagonals: rates of change from each nucleotide to each other nucleotide
- Rows sum to zero
- Different numbers of parameters:



$$Q_{\mathrm{JC69}} = \begin{array}{c|cccc} & A & C & G & T \\ \hline A & -3\alpha & \alpha & \alpha & \alpha \\ C & \alpha & -3\alpha & \alpha & \alpha \\ G & \alpha & \alpha & -3\alpha & \alpha \\ T & \alpha & \alpha & \alpha & -3\alpha \end{array}$$

Jukes-Cantor: all substitution rates (α) and nucleotide frequencies are the same



$$Q_{\mathrm{GTR}} = \begin{array}{c|cccc} & A & C & G & T \\ \hline A & -q_A & r_{AC}\pi_C & r_{AG}\pi_G & r_{AT}\pi_T \\ C & r_{AC}\pi_A & -q_C & r_{CG}\pi_G & r_{CT}\pi_T \\ G & r_{AG}\pi_A & r_{CG}\pi_C & -q_G & r_{GT}\pi_T \\ T & r_{AT}\pi_A & r_{CT}\pi_C & r_{GT}\pi_G & -q_T \end{array}$$

General Time Reversible (GTR): different rates of change, and nucleotide frequencies

Longer B_1 leads to larger probabilities of change

$$(e^{QB_1})$$

Jukes and Cantor (1969) *Mammalian Protein Metabolism*

Tavaré (1986). *Some Mathematical Questions in Biology - DNA Sequence Analysis*

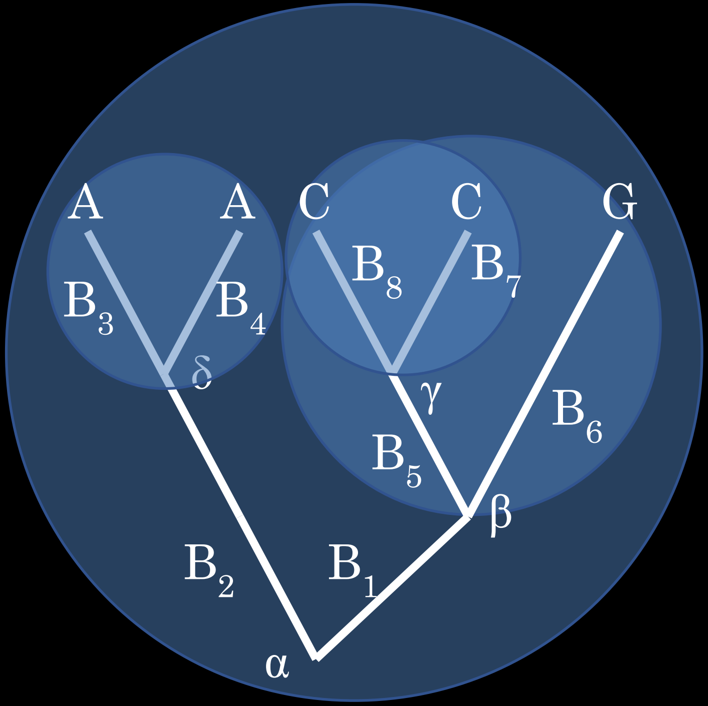
Amino Acid Rate Matrices

20 x 20 amino acid matrices are usually predefined (*empirical* substitution matrices)

Examples: PAM, JTT, BLOSUM, VT, WAG, LG – different source datasets and counting techniques

Why don’t we do amino acid GTR?

Felsenstein's likelihood algorithm



Dynamic Programming yet again
Start at the tips, and work backward through the tree

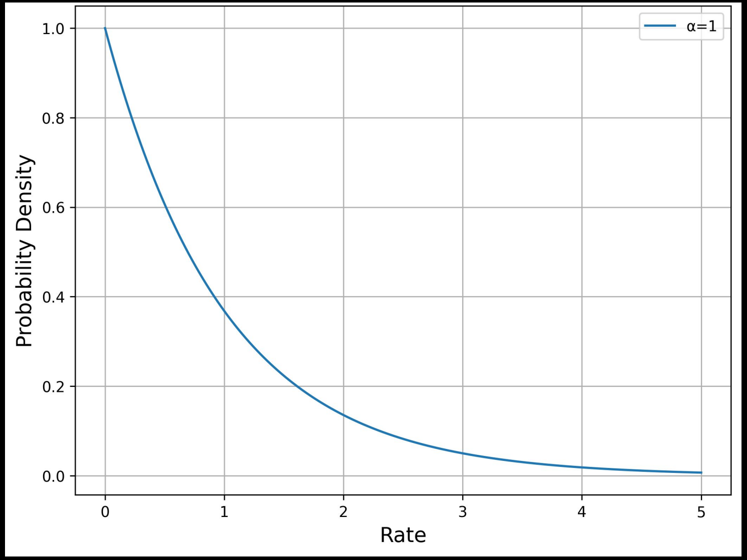
Previous method was b^{n-1} operations
    b = # of bases (alphabet size)
    n = # of taxa
DP method requires $(n - 1)b^2$ operations

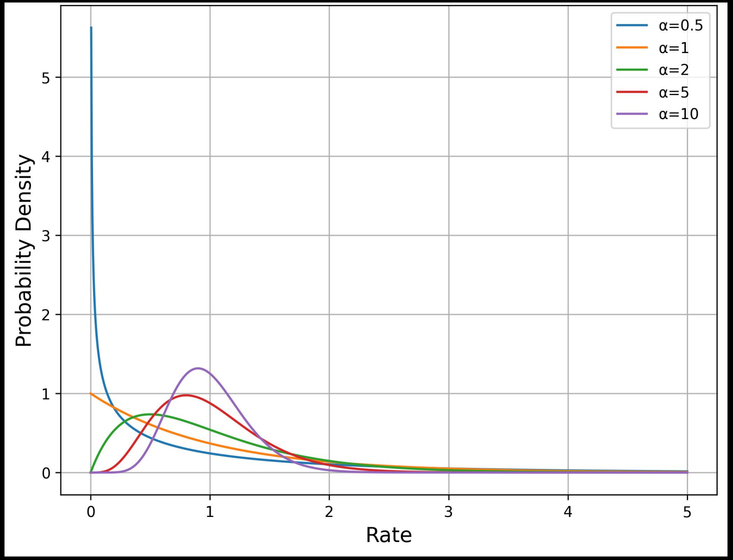
Reuse computed likelihoods on each branch, rather than recomputing them every time

Rate Heterogeneity

- Different sites evolve at different rates, but the basic substitution model violates this assumption

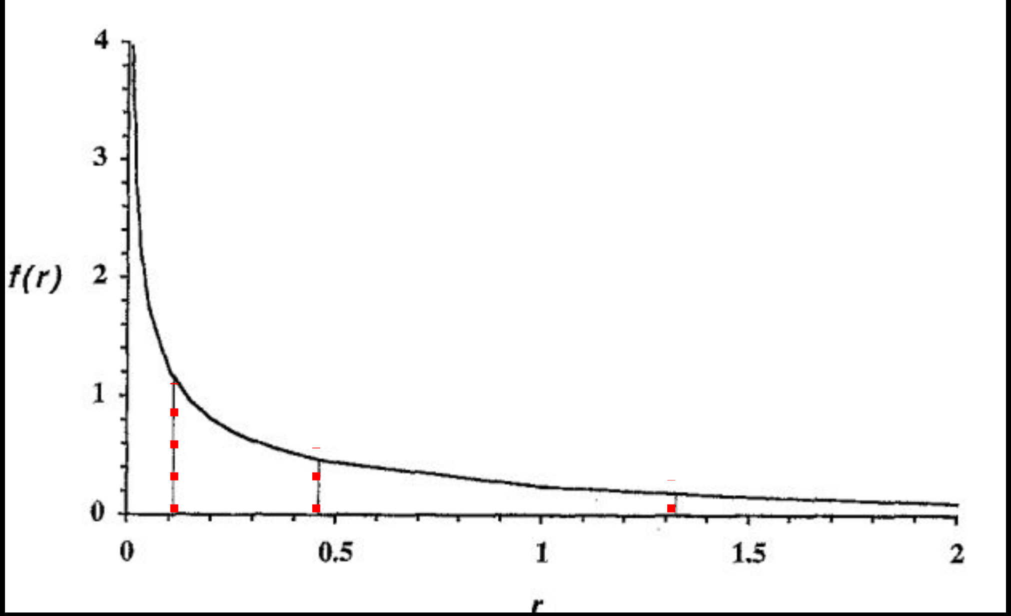
- We can model this with a one-parameter gamma distribution:





Rate Heterogeneity

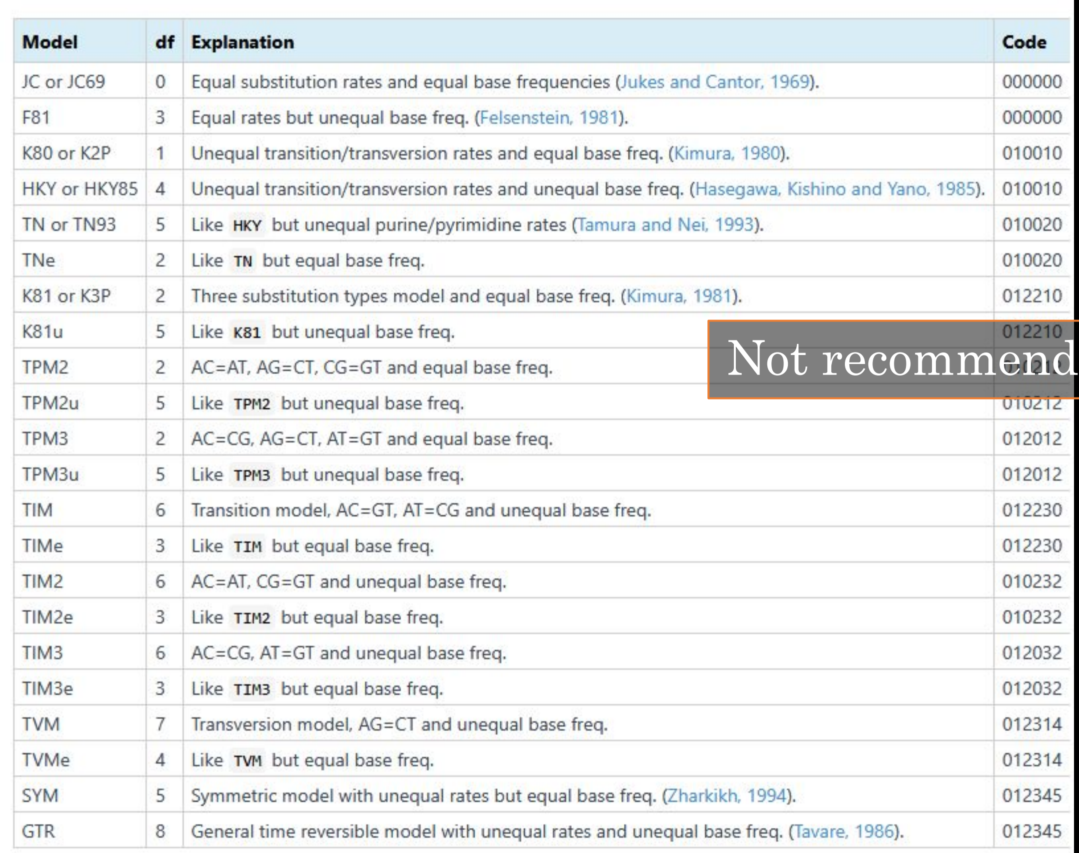
- Modeling a continuous gamma distribution is computationally intensive, so we use a discretized (binned) set of probabilities instead
- The number of categories is typically fixed at the start of the run, while the value of α is a parameter optimized by the model



Yang (1994) *Journal of Molecular Evolution*

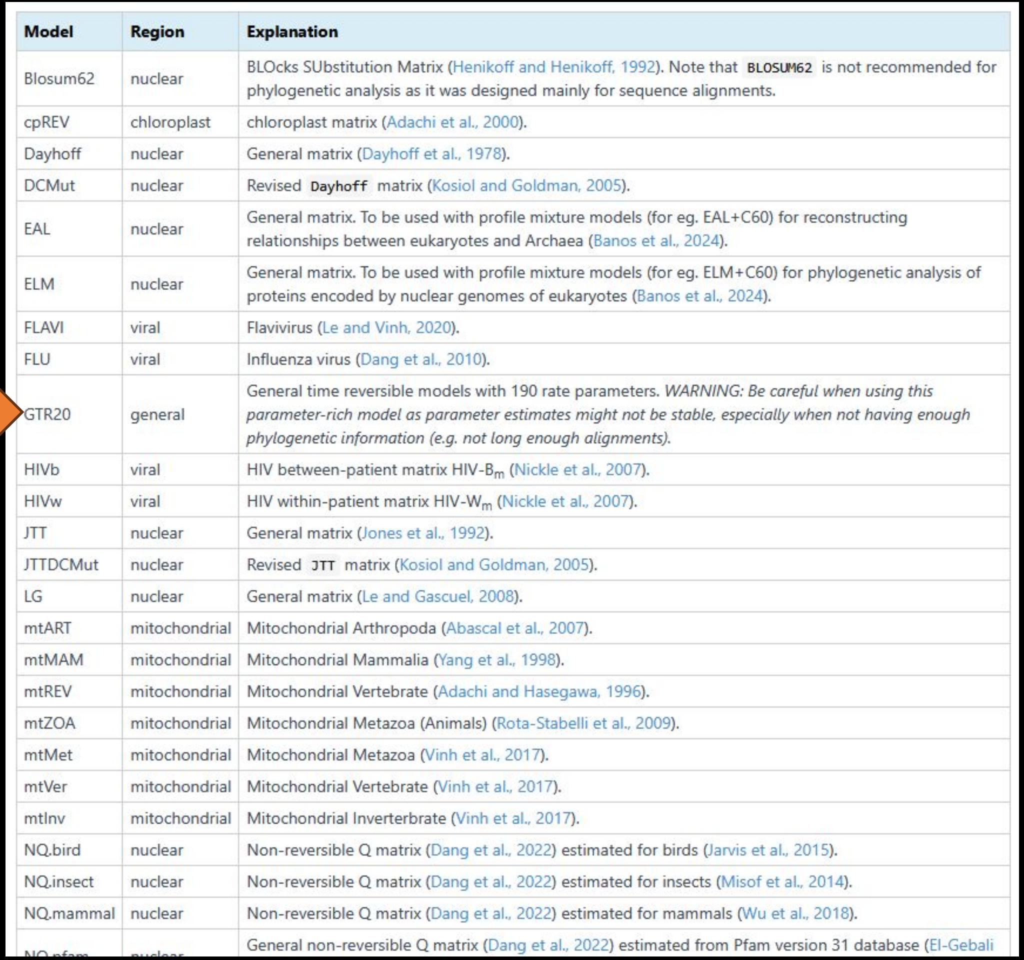
So Many Models…

Nucleotide



Model	df	Explanation	Code
JC or JC69	0	Equal substitution rates and equal base frequencies (Jukes and Cantor, 1969).	000000
F81	3	Equal rates but unequal base freq. (Felsenstein, 1981).	000000
K80 or K2P	1	Unequal transition/transversion rates and equal base freq. (Kimura, 1980).	010010
HKY or HKY85	4	Unequal transition/transversion rates and unequal base freq. (Hasegawa, Kishino and Yano, 1985).	010010
TN or TN93	5	Like `HKY` but unequal purine/pyrimidine rates (Tamura and Nei, 1993).	010020
TNe	2	Like `TN` but equal base freq.	010020
K81 or K3P	2	Three substitution types model and equal base freq. (Kimura, 1981).	012210
K81u	5	Like `K81` but unequal base freq.	012210
TPM2	2	AC=AT, AG=CT, CG=GT and equal base freq.	010212
TPM2u	5	Like `TPM2` but unequal base freq.	010212
TPM3	2	AC=CG, AG=CT, AT=GT and equal base freq.	012012
TPM3u	5	Like `TPM3` but unequal base freq.	012012
TIM	6	Transition model, AC=GT, AT=CG and unequal base freq.	012230
TIMe	3	Like `TIM` but equal base freq.	012230
TIM2	6	AC=AT, CG=GT and unequal base freq.	010232
TIM2e	3	Like `TIM2` but equal base freq.	010232
TIM3	6	AC=CG, AT=GT and unequal base freq.	012032
TIM3e	3	Like `TIM3` but equal base freq.	012032
TVM	7	Transversion model, AG=CT and unequal base freq.	012314
TVMe	4	Like `TVM` but equal base freq.	012314
SYM	5	Symmetric model with unequal rates but equal base freq. (Zharkikh, 1994).	012345
GTR	8	General time reversible model with unequal rates and unequal base freq. (Tavare, 1986).	012345

Not recommended!



Model	Region	Explanation
Blosum62	nuclear	BLOcks SUbstitution Matrix (Henikoff and Henikoff, 1992). Note that `BLOSUM62` is not recommended for phylogenetic analysis as it was designed mainly for sequence alignments.
cpREV	chloroplast	chloroplast matrix (Adachi et al., 2000).
Dayhoff	nuclear	General matrix (Dayhoff et al., 1978).
DCMut	nuclear	Revised `Dayhoff` matrix (Kosiol and Goldman, 2005).
EAL	nuclear	General matrix. To be used with profile mixture models (for eg. EAL+C60) for reconstructing relationships between eukaryotes and Archaea (Banos et al., 2024).
ELM	nuclear	General matrix. To be used with profile mixture models (for eg. ELM+C60) for phylogenetic analysis of proteins encoded by nuclear genomes of eukaryotes (Banos et al., 2024).
FLAVI	viral	Flavivirus (Le and Vinh, 2020).
FLU	viral	Influenza virus (Dang et al., 2010).
GTR20	general	General time reversible models with 190 rate parameters. *WARNING: Be careful when using this parameter-rich model as parameter estimates might not be stable, especially when not having enough phylogenetic information (e.g. not long enough alignments).*
HIVb	viral	HIV between-patient matrix HIV-B$_m$ (Nickle et al., 2007).
HIVw	viral	HIV within-patient matrix HIV-W$_m$ (Nickle et al., 2007).
JTT	nuclear	General matrix (Jones et al., 1992).
JTTDCMut	nuclear	Revised `JTT` matrix (Kosiol and Goldman, 2005).
LG	nuclear	General matrix (Le and Gascuel, 2008).
mtART	mitochondrial	Mitochondrial Arthropoda (Abascal et al., 2007).
mtMAM	mitochondrial	Mitochondrial Mammalia (Yang et al., 1998).
mtREV	mitochondrial	Mitochondrial Vertebrate (Adachi and Hasegawa, 1996).
mtZOA	mitochondrial	Mitochondrial Metazoa (Animals) (Rota-Stabelli et al., 2009).
mtMet	mitochondrial	Mitochondrial Metazoa (Vinh et al., 2017).
mtVer	mitochondrial	Mitochondrial Vertebrate (Vinh et al., 2017).
mtInv	mitochondrial	Mitochondrial Inverterbrate (Vinh et al., 2017).
NQ.bird	nuclear	Non-reversible Q matrix (Dang et al., 2022) estimated for birds (Jarvis et al., 2015).
NQ.insect	nuclear	Non-reversible Q matrix (Dang et al., 2022) estimated for insects (Misof et al., 2014).
NQ.mammal	nuclear	Non-reversible Q matrix (Dang et al., 2022) estimated for mammals (Wu et al., 2018).
NQ.pfam	nuclear	General non-reversible Q matrix (Dang et al., 2022) estimated from Pfam version 31 database (El-Gebali

Amino Acid

http://www.iqtree.org/doc/Substitution-Models

Maximum Likelihood

- Given an alignment, find the set of parameter values that maximize L

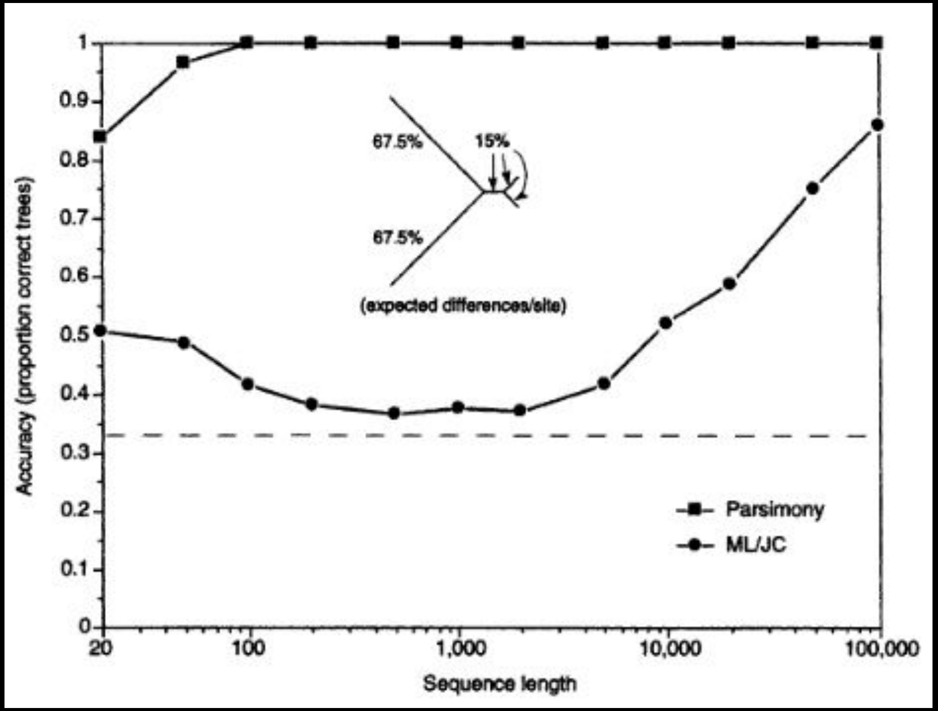
- As with parsimony, we need to perform a search through tree space

- But now, in addition to considering the tree shape, we must add branch lengths and substitution probabilities to the model

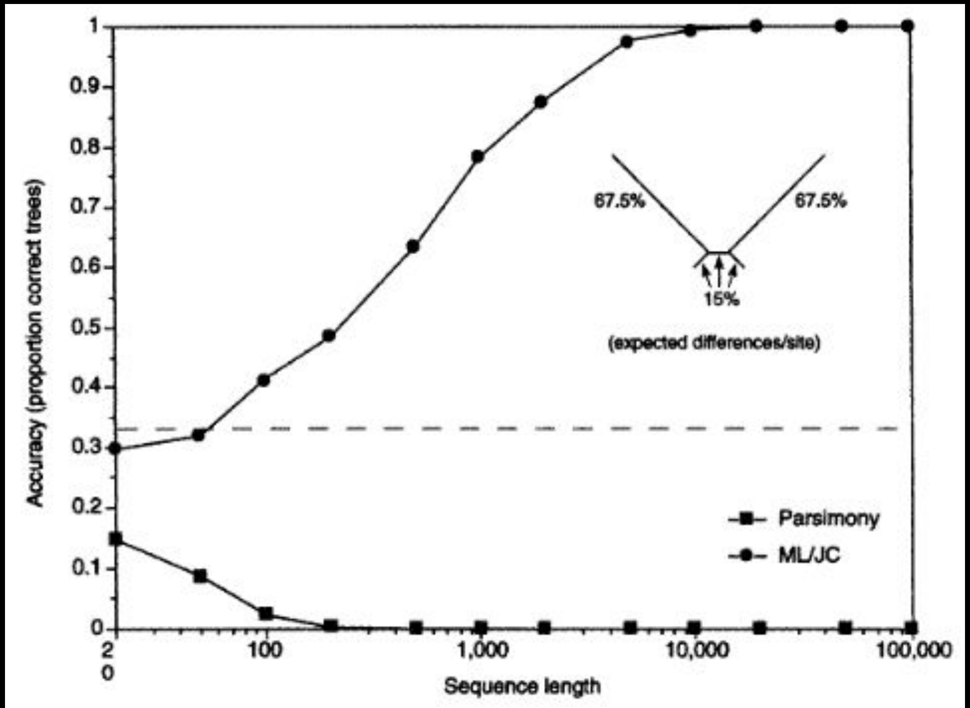
How do those distance methods work again?

Likelihood vs. Parsimony

Accuracy under two different tree shapes (simulated data)



Parsimony does really well when long branches are adjacent in the tree



Parsimony is awful when long branches are separate in the tree

Swofford et al (2001) *Systematic Biology*

What’s going on?

- Convergent substitutions:
    - Long branches will have many changes
    - Some of these changes will converge by chance!
    - Parsimony consequently sees these sequences as being more similar than they really are

= Long-branch attraction

The key difference…

- In parsimony we consider only the best internal states of the tree (Fitch’s algorithm!)

- Whereas in likelihood calculations, all possible internal states are modeled

$$= P(\alpha = A) \times P(\beta = A \mid \alpha = A, B_1) \times \ldots$$
$$+ P(\alpha = C) \times P(\beta = A \mid \alpha = C, B_1) \times \ldots$$

Maximum Likelihood in practice

- Not only do we need to find the best tree shape, we must also optimize the branch lengths

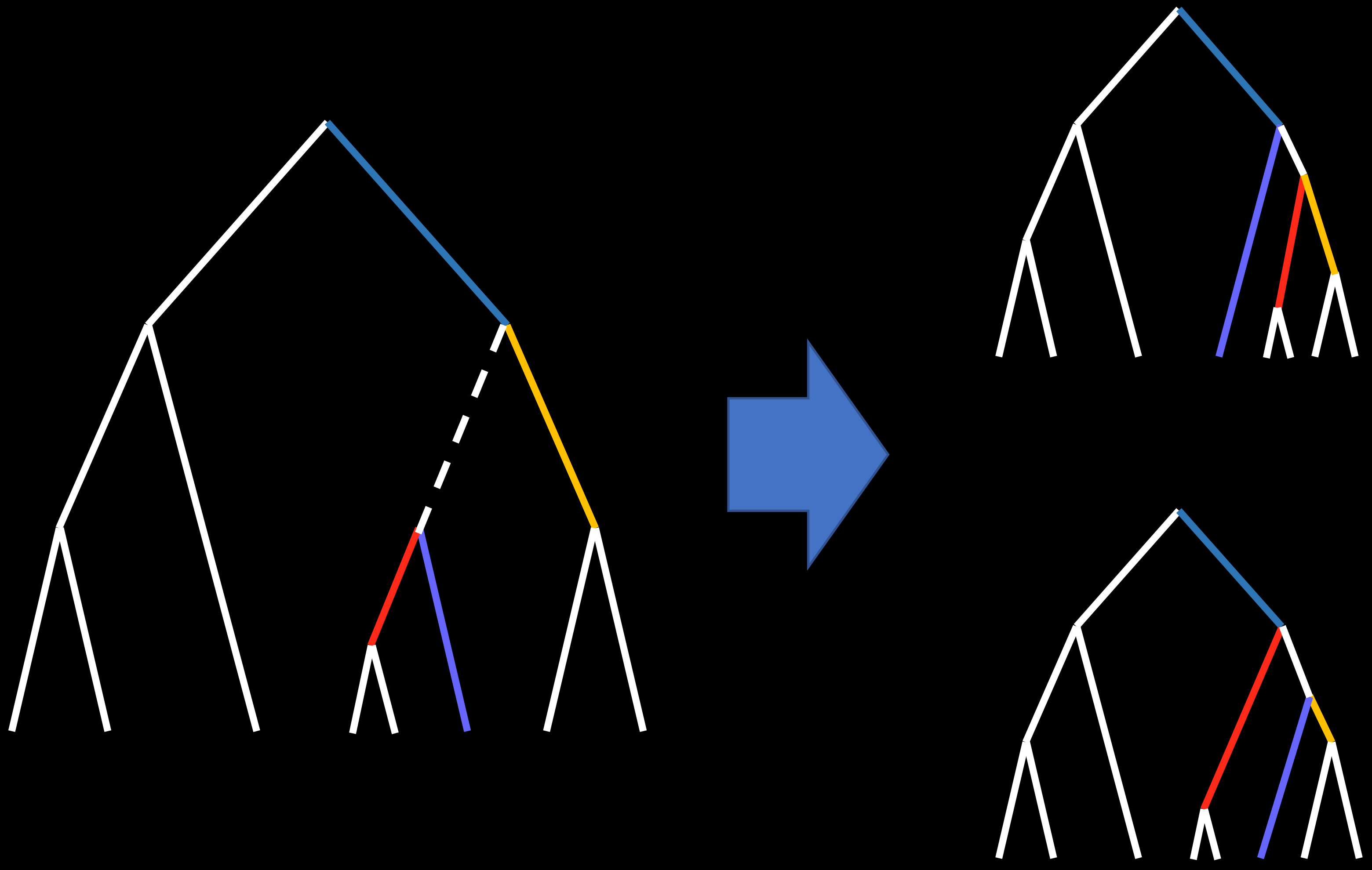
- Heuristics are desperately needed!

Searching through tree space

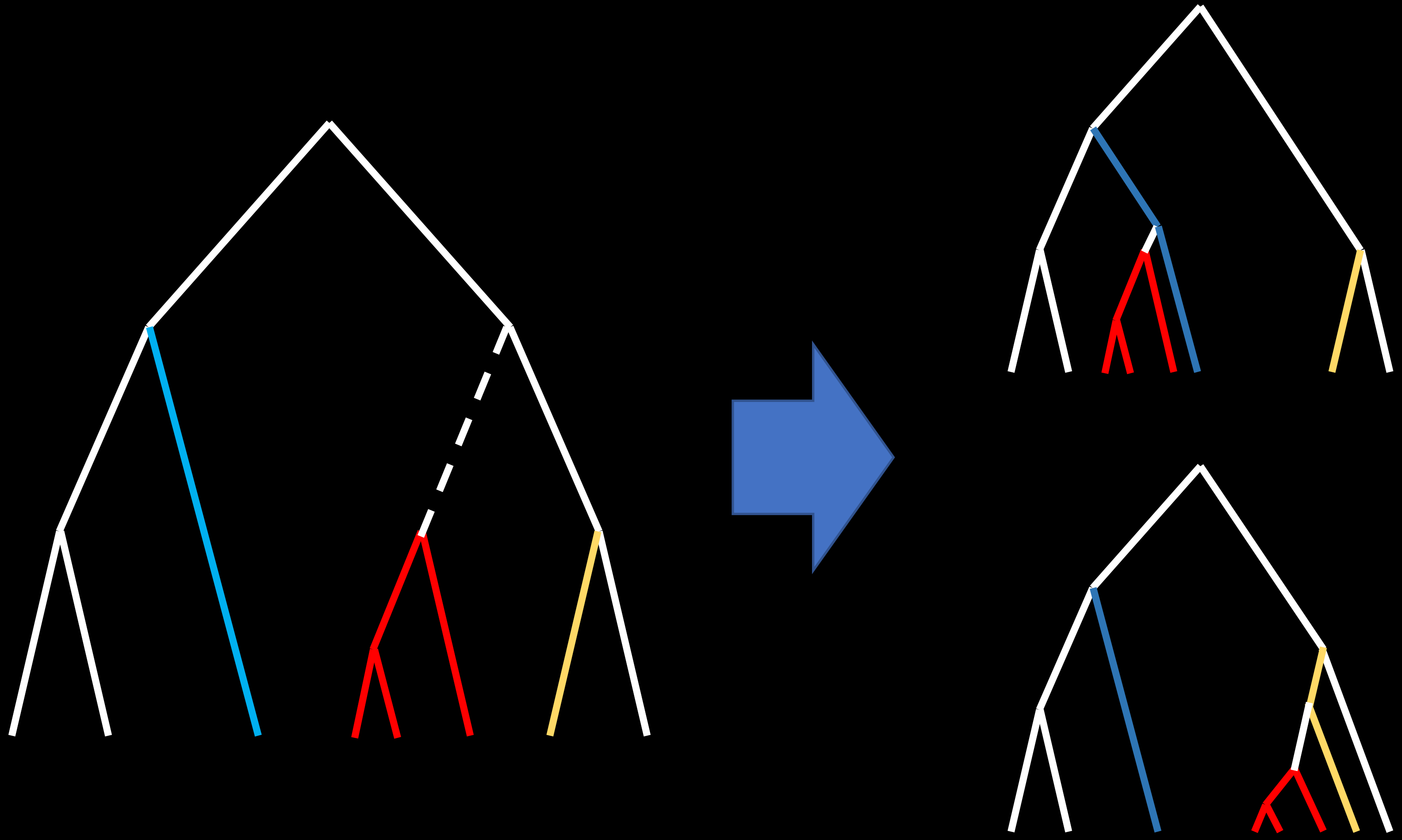
- We need techniques to permute the tree at every step

- Different permutations can induce smaller or larger changes in the tree topology

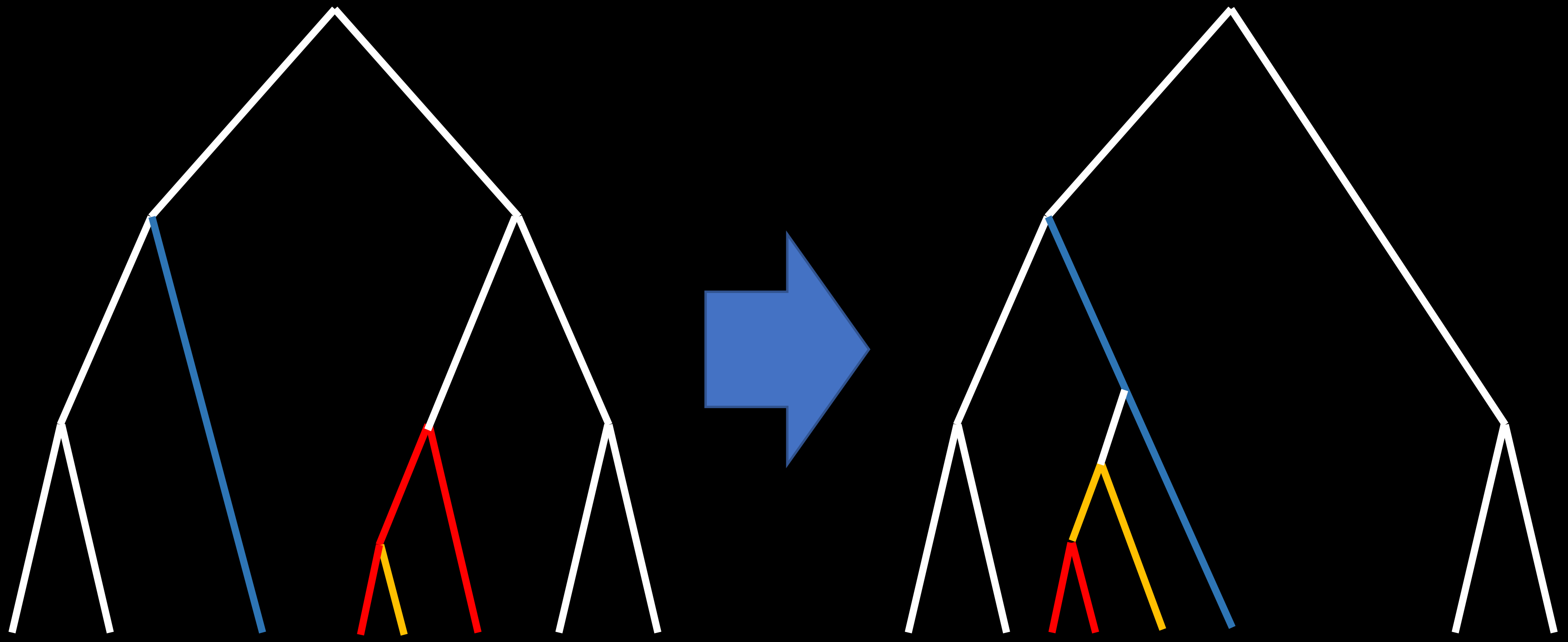
Nearest-neighbour interchange (NNI)



Subtree Prune and Regraft (SPR)



Tree bisection and reconnection



Thoughts on which is best for searching tree space?

Key questions in ML tree finding

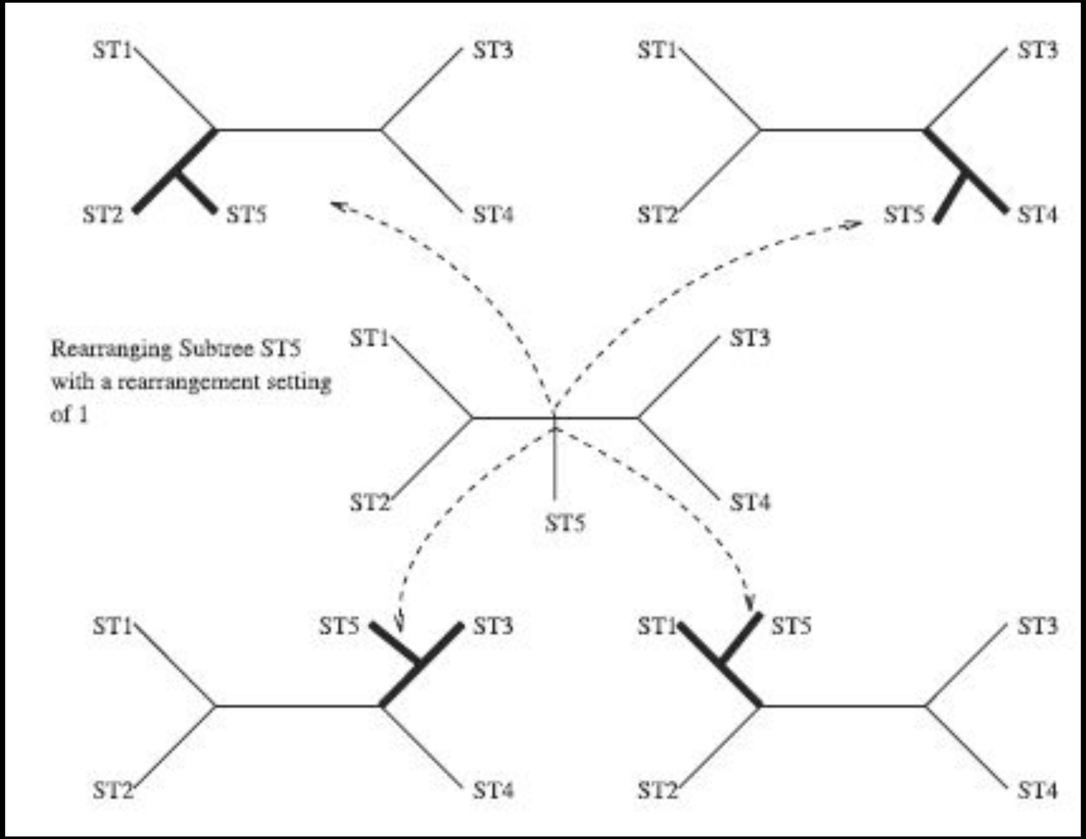
- Where do we start?
- What search strategy do we use?
- When do we optimize branch lengths?
- When do we stop?

RAxML: Fancy Tree Searching

Starting tree: stepwise addition, maximum parsimony (fast!)

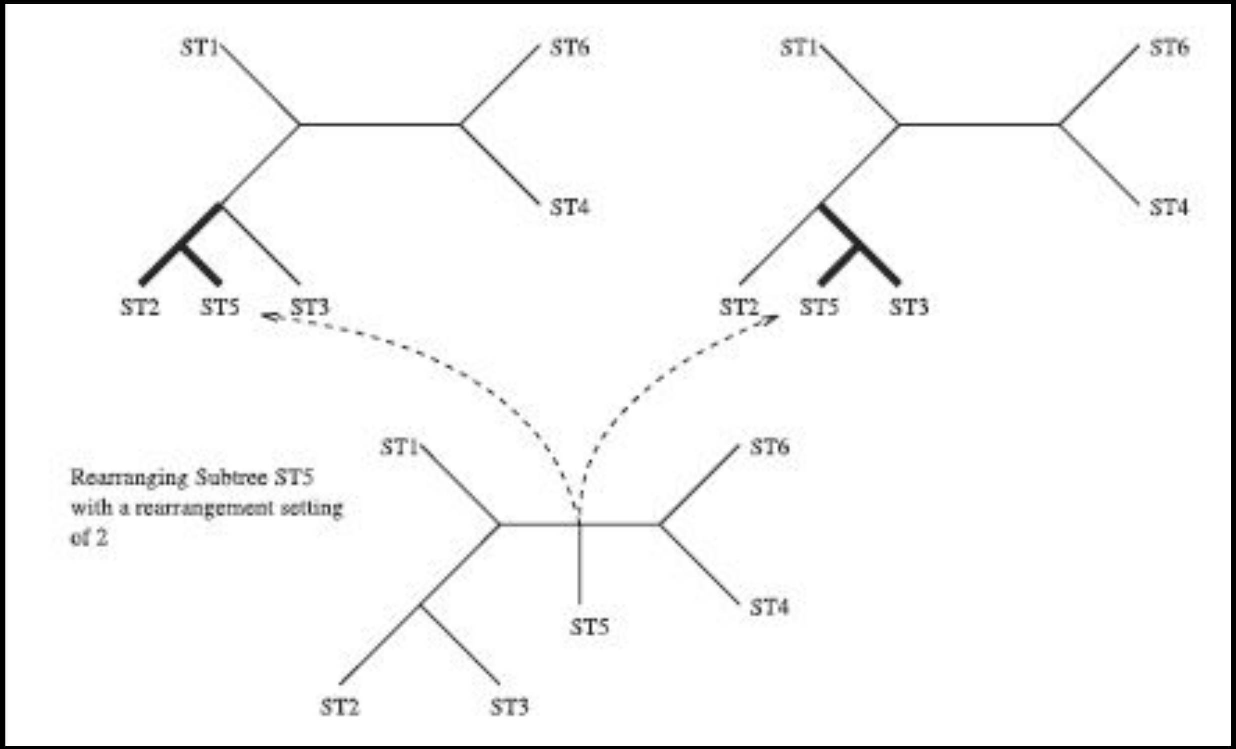
Tree search using constrained SPR, where each subtree is moved between *Rmin* and *Rmax* steps along the tree

1



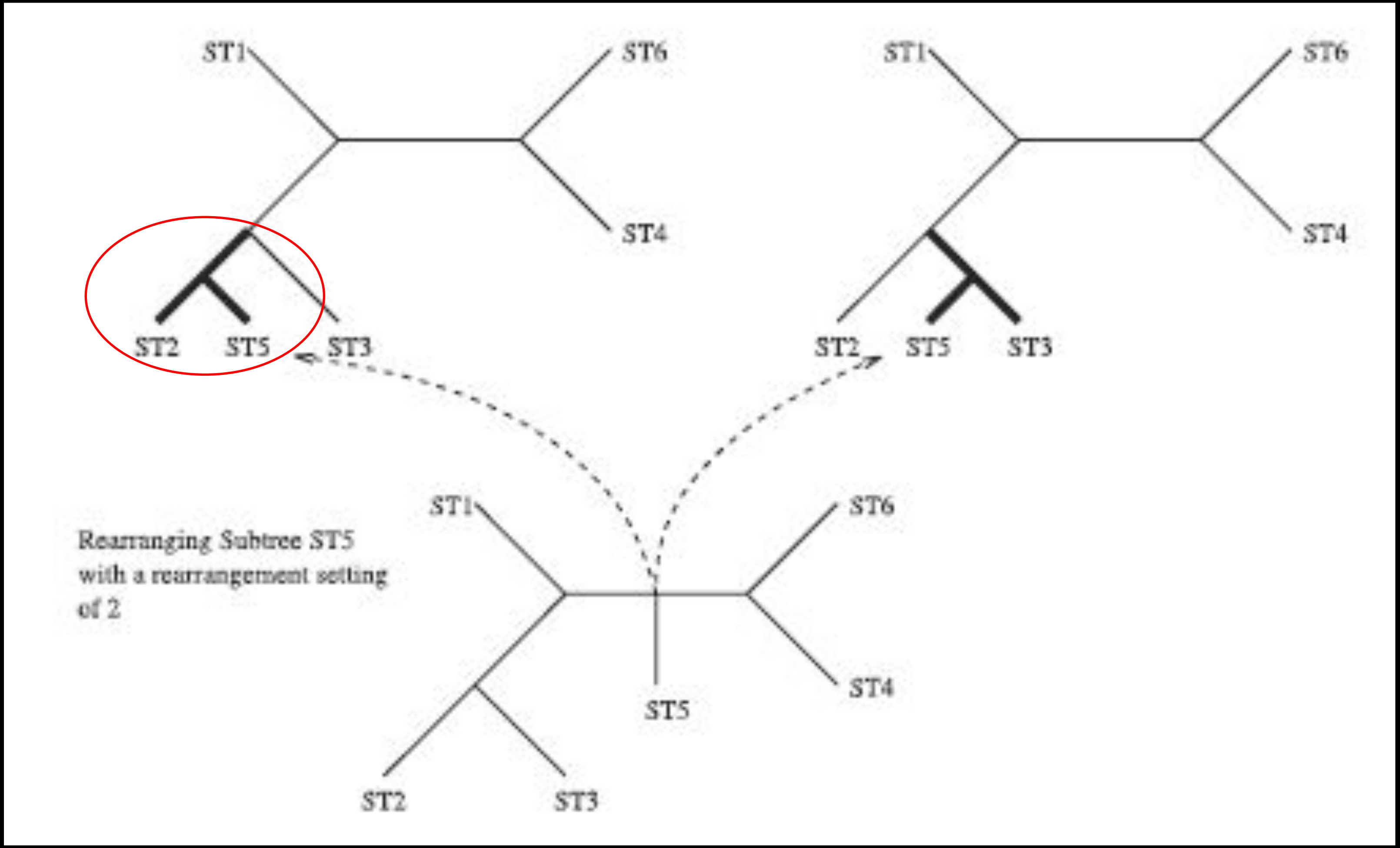
Rearranging Subtree ST5 with a rearrangement setting of 1

2



Rearranging Subtree ST5 with a rearrangement setting of 2

During the complete subtree search, only optimize the branch lengths that are directly implicated in the swap



- Rank all the resulting trees based on their likelihood
- Choose the top 20 for full branch length optimization

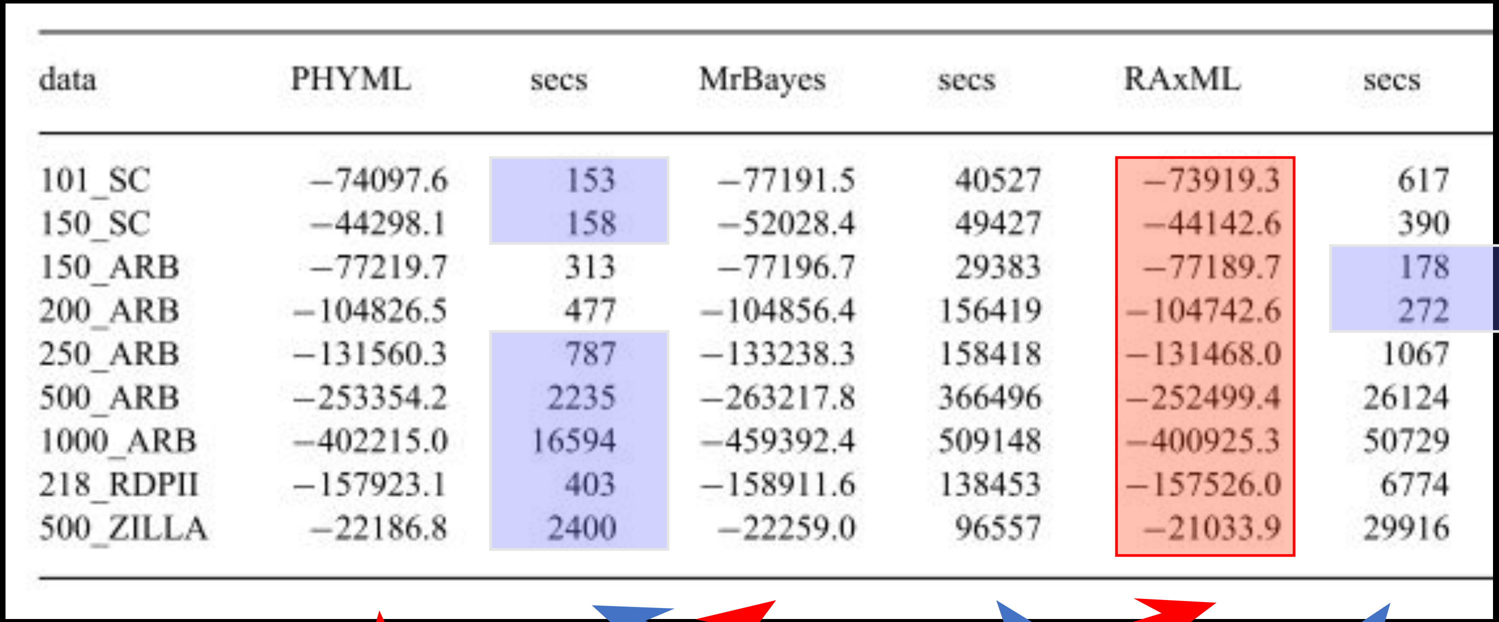
Stopping conditions

- Set a maximum value for $Rmax$

- If the tree does not improve during an iteration, increment $Rmin$ and $Rmax$

- When $Rmax = \max(Rmax)$, stop!

Performance comparison



data	PHYML	secs	MrBayes	secs	RAxML	secs
101_SC	−74097.6	153	−77191.5	40527	−73919.3	617
150_SC	−44298.1	158	−52028.4	49427	−44142.6	390
150_ARB	−77219.7	313	−77196.7	29383	−77189.7	178
200_ARB	−104826.5	477	−104856.4	156419	−104742.6	272
250_ARB	−131560.3	787	−133238.3	158418	−131468.0	1067
500_ARB	−253354.2	2235	−263217.8	366496	−252499.4	26124
1000_ARB	−402215.0	16594	−459392.4	509148	−400925.3	50729
218_RDPII	−157923.1	403	−158911.6	138453	−157526.0	6774
500_ZILLA	−22186.8	2400	−22259.0	96557	−21033.9	29916

Log-likelihoods
(closer to 0 = better)

Running times

Stamatakis et al. (2005) *Bioinformatics*

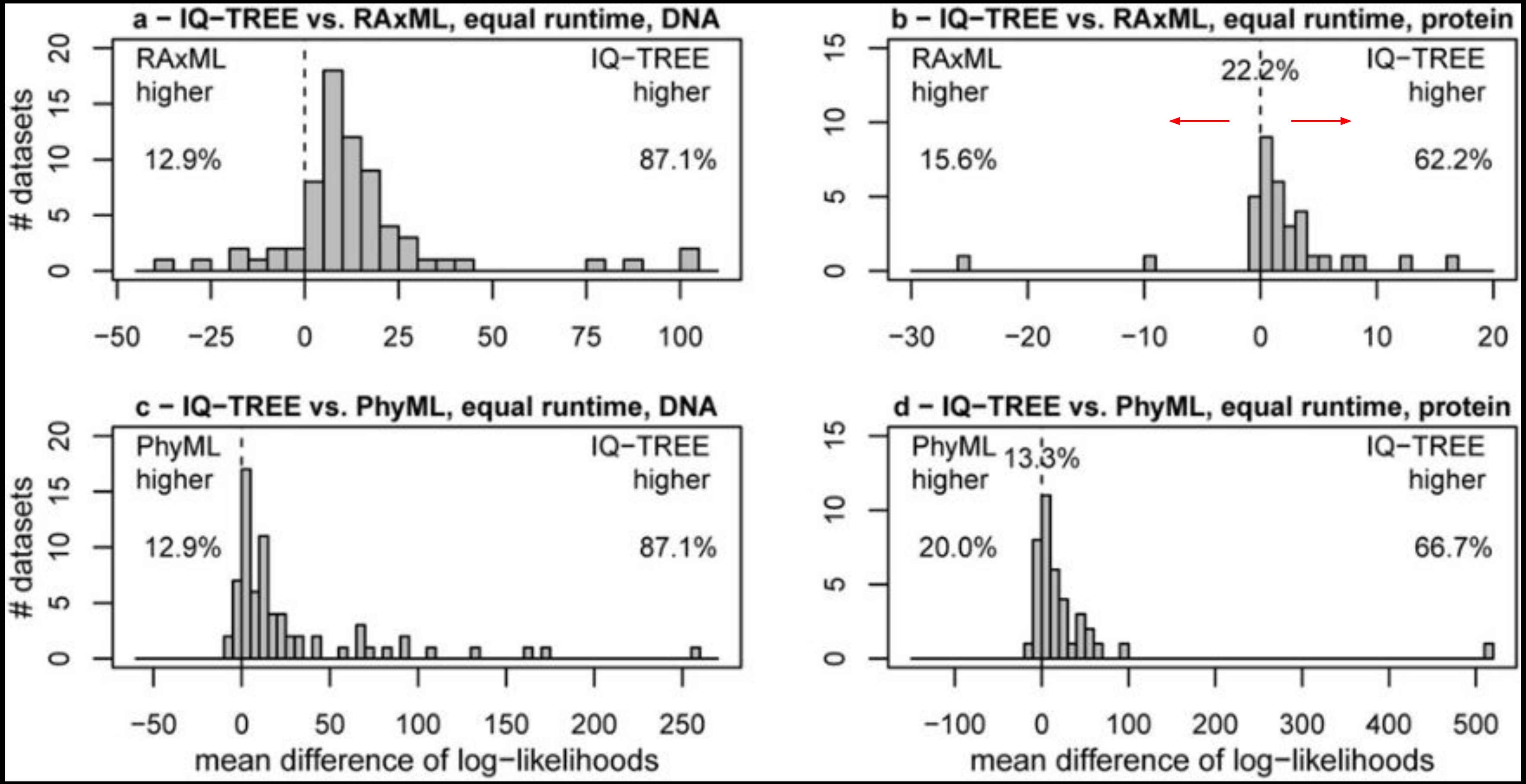
Why RAxML works

- The tree search is a compromise between a narrow, precise search and a broader search
- Only optimize when you need to
- Other stuff: different available models, parallelization, etc.

IQ-TREE

- Key differences with RAxML:
  - Use 100 starting parsimony trees (rapidly inferred, avoid local optima)
  - Filter filter filter!! Optimize branch lengths using ML, purge, then *really* optimize the top 5 trees
  - Perturb these trees with a bunch of random NNIs, re-optimize
  - Stop if 100 rounds of this yield no improvement

Nguyen et al. (2015) *Mol. Biol. Evol.*


a – IQ–TREE vs. RAxML, equal runtime, DNA
RAxML higher
IQ–TREE higher
12.9%
87.1%
# datasets
0
5
10
15
20
−50
−25
0
25
50
75
100
b – IQ–TREE vs. RAxML, equal runtime, protein
RAxML higher
22.2%
IQ–TREE higher
15.6%
62.2%
0
5
10
15
−30
−20
−10
0
10
20
c – IQ–TREE vs. PhyML, equal runtime, DNA
PhyML higher
IQ–TREE higher
12.9%
87.1%
# datasets
0
5
10
15
20
−50
0
50
100
150
200
250
mean difference of log−likelihoods
d – IQ–TREE vs. PhyML, equal runtime, protein
PhyML higher
13.3%
IQ–TREE higher
20.0%
66.7%
0
5
10
15
−100
0
100
200
300
400
500
mean difference of log−likelihoods

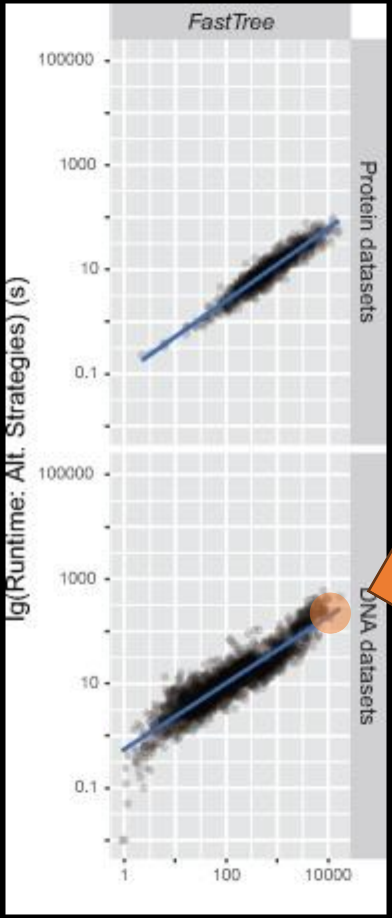
Fasttree2: Approximate Likelihood

3 steps:

1. (not likelihood): use neighbor-joining to build the initial tree
2. (not likelihood): use minimum evolution to locally permute the tree to minimize its length
3. (likelihood):
   - Repeated rounds of NNIs (capped in proportion to the size of the tree)
   - Freeze parts of the tree that show no improvement
   - Delay calculation of branch lengths, substitution rates

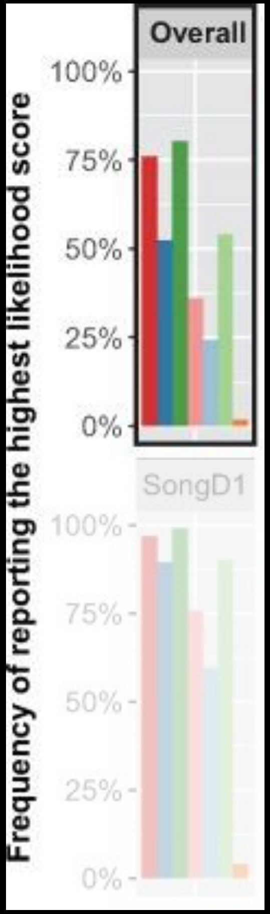
Price et al. (2010) *PLoS ONE*

Fastree: Speed vs Accuracy



Fasttree can be > 100x faster than RAxML and other tools

RAxML ~ 10,000 s
Fasttree ~ 100 s



But it's not great at finding the best tree

Summary

- Likelihood gives you the best of both worlds: model-based tree construction, and consideration of every character

- Likelihood-based methods are very time consuming, and imperfect heuristics are needed

- IQ-TREE, RAxML: heuristic
- Fasttree: very heuristic