PM520: Lecture 8

Further explorations in MCMC

MCMC: Metropolis-Hastings

1. If at *x*, propose move to *x*′ according to transition kernel q(*x* -> *x*′).
2. Calculate

$$h = \min\left\{1, \frac{f(x')q(x' \to x)}{f(x)q(x \to x')}\right\}$$

3. Move to *x’* with prob. h, else remain at *x*.
4. Return to 1.

(Metropolis et al. 1953, Hastings 1970)

The Markov chain has stationary distribution f.

Metropolis-Hastings: Features

- Pros:
  - The algorithm ‘learns’.
  - Comparison is ‘local’.
  - MCMC package in R, Gelman or CODA convergence diagnostics, WinBUGS, Many others…

- Cons:
  - Need to be able to calculate f(x) (may be non-trivial xin some cases - we will see an example of this today).
  - Need to sample from stationary distribution.
  - Consecutive outputs are correlated.
  - May get stuck in local minima

- Book chapter on Blackboard “Chapter on MCMC” - read up to section 7.4.3 (inclusive)

Sampling from Posterior Distributions

- Data, D; model parameter(s) θ. Density f(D|θ).
- Bayes theorem:

$$f(\theta|D)= f(D|\theta)\pi(\theta)/f(D)$$

Normalizing constant

Likelihood

Prior distribution

MCMC: Metropolis-Hastings

1. If at *θ*, propose move to *θ*' according to transition kernel q(*θ* -> *θ*').
2. Calculate

$$h = \min\left\{1, \frac{[P(\theta'|D)q(\theta' \rightarrow \theta)}{[P(\theta|D)q(\theta \rightarrow \theta')}\right\}$$

3. Move to *θ*' with prob. h, else remain at *θ*.
4. Return to 1.

The stationary distribution of this chain will be P(*θ* | D)

MCMC: Metropolis-Hastings

1. If at θ, propose move to θ' according to transition kernel q(θ -> θ').
2. Calculate

$$h = \min\left\{1, \frac{[P(D|\theta')\pi(\theta')/P(D)]q(\theta' \to \theta)}{[P(D|\theta)\pi(\theta)/P(D)]q(\theta \to \theta')}\right\}$$

3. Move to θ' with prob. h, else remain at θ.
4. Return to 1.

The stationary distribution of this chain will be P(θ | D)

MCMC: Metropolis-Hastings

1. If at *θ*, propose move to *θ′* according to transition kernel q(*θ* -> *θ′*).
2. Calculate

$$h = \min\left\{1, \frac{P(D|\theta')\pi(\theta')q(\theta' \to \theta)}{P(D|\theta)\pi(\theta)q(\theta \to \theta')}\right\}$$

3. Move to *θ′* with prob. h, else remain at *θ*.
4. Return to 1.

The stationary distribution of this chain will be P(*θ* | D).

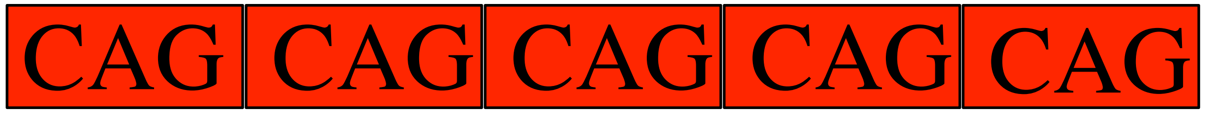
P(D) disappears, **so this can be used even when *P(D)* cannot be calculated!**

Example MCMC Application

- Wilson and Balding: Genealogical inference from microsatellite data, Genetics 150:499-510, 1998.

- Outline:
  - Use micro-satellite (short tandem repeats) data for pop. gen. evolution
  - Good: easy to collect
  - Bad: hard to interpret (*back mutation*)
  - Use MCMC on coalescent (ancestral) trees ***with an augmented state-space.***

Micro-satellite

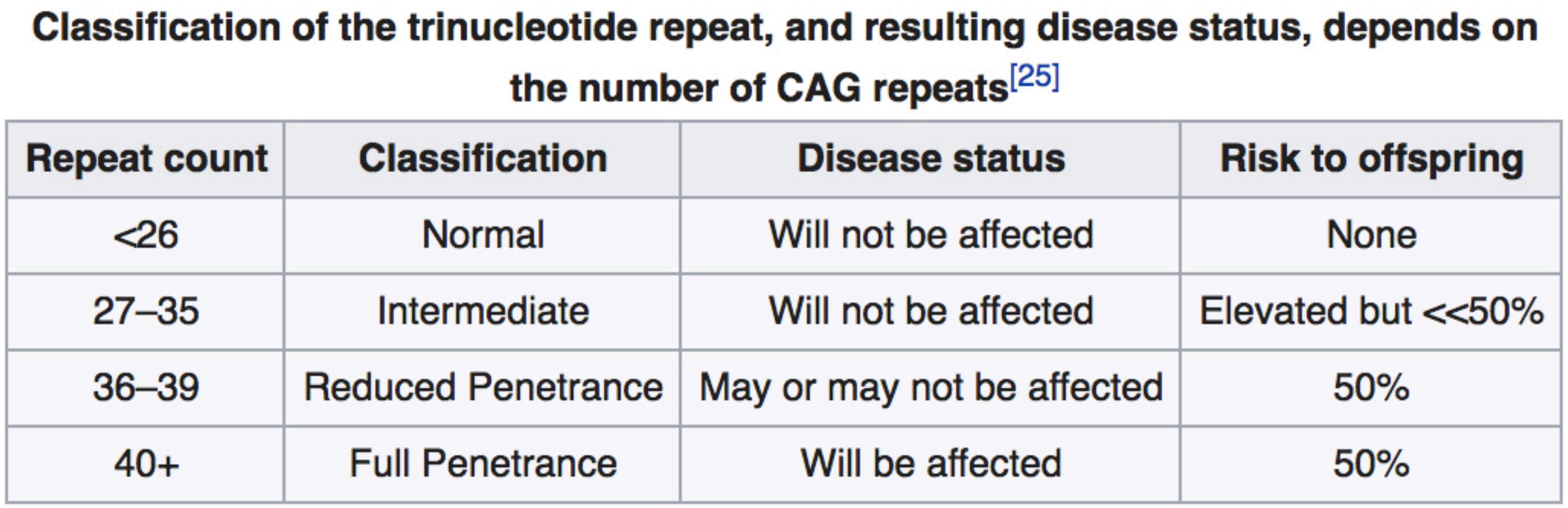


CAG CAG CAG CAG CAG

- Number of repeats varies from person-to-person
- Used as markers in population studies
- Very common; mostly neutral but some associated with disease (Trinucleotide repeat disorders, e.g. Fragile X syndrome, Huntington’s disease)
- Mutation rate relatively high (slippage)

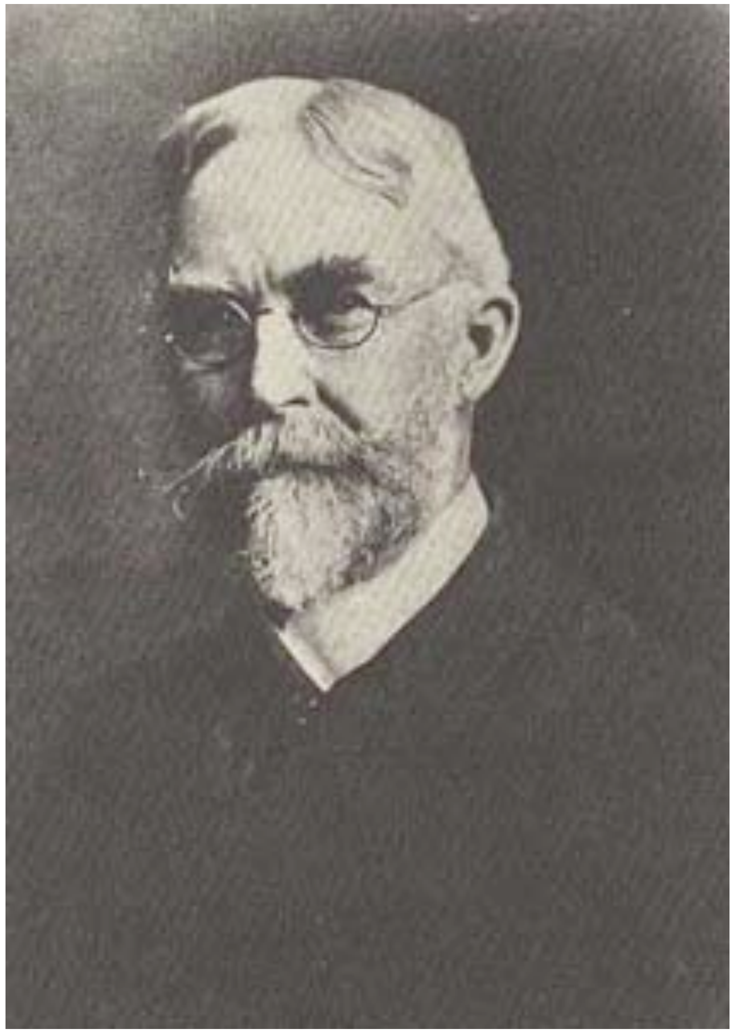
Huntington’s disease

- Described by George Huntington in 1872
- CAG repeat on short arm of chr 4
- Neurodegenerative
- More repeats -> earlier onset
- Number of repeats is most likely to increase via paternal inheritance



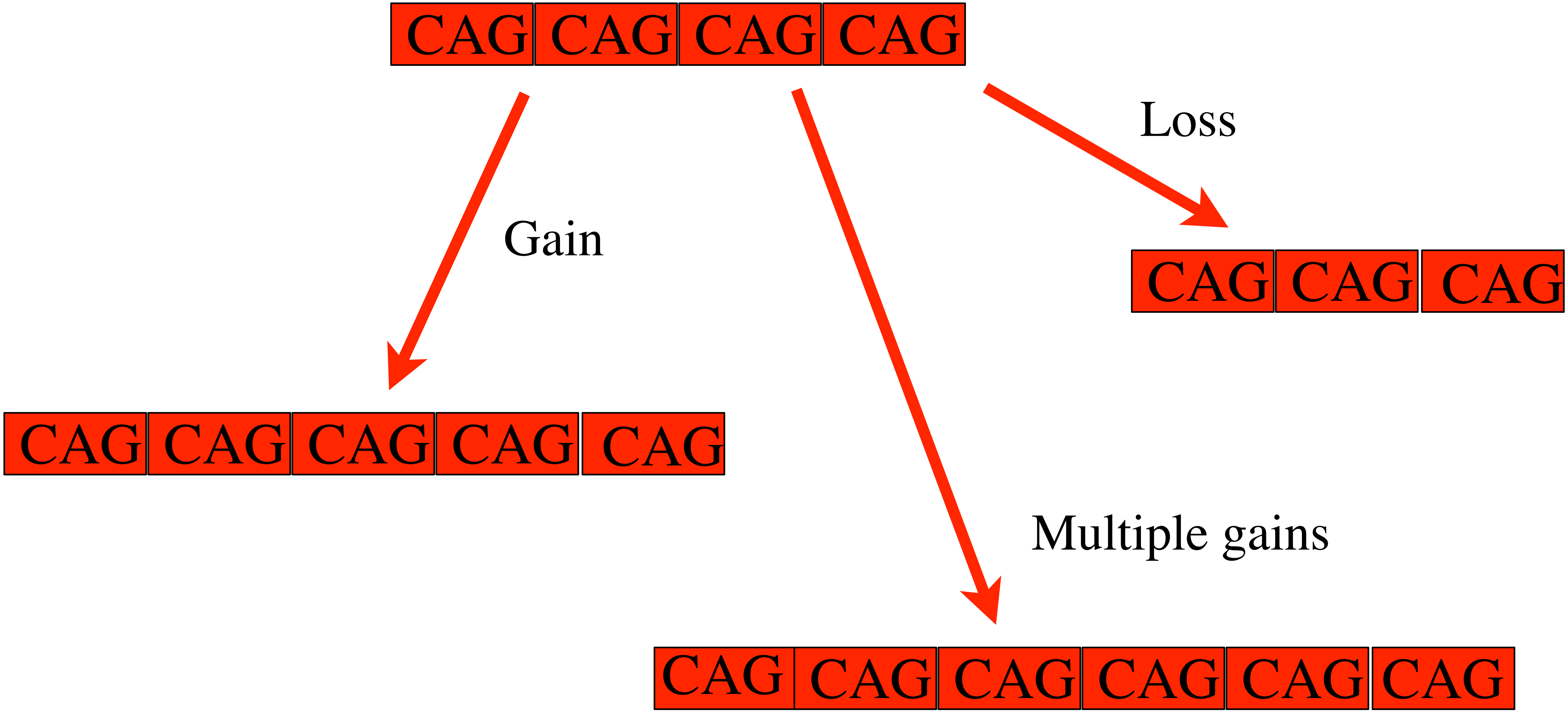
Classification of the trinucleotide repeat, and resulting disease status, depends on the number of CAG repeats[25]

Repeat count	Classification	Disease status	Risk to offspring
<26	Normal	Will not be affected	None
27–35	Intermediate	Will not be affected	Elevated but <<50%
36–39	Reduced Penetrance	May or may not be affected	50%
40+	Full Penetrance	Will be affected	50%



https://en.wikipedia.org/wiki/Huntington%27s_disease

Micro-satellite mutation



CAG CAG CAG CAG

Gain

CAG CAG CAG CAG CAG

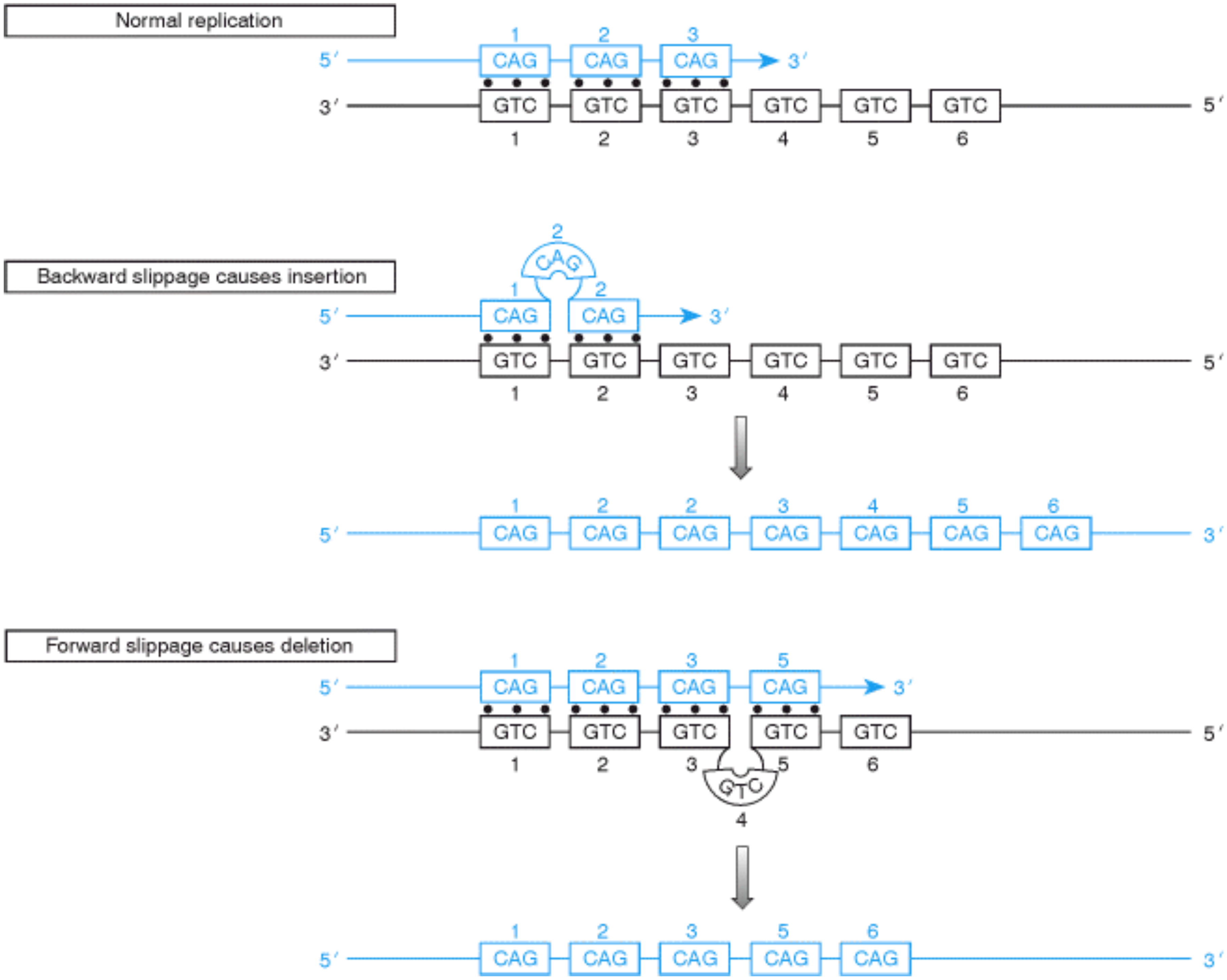
Multiple gains

CAG CAG CAG CAG CAG CAG

Loss

CAG CAG CAG

Slipped-strand mispairing


Normal replication
5′ 1 CAG 2 CAG 3 CAG 3′
3′ GTC 1 GTC 2 GTC 3 GTC 4 GTC 5 GTC 6 5′
Backward slippage causes insertion
2 CAG
5′ 1 CAG 2 CAG 3′
3′ GTC 1 GTC 2 GTC 3 GTC 4 GTC 5 GTC 6 5′
5′ 1 CAG 2 CAG 2 CAG 3 CAG 4 CAG 5 CAG 6 CAG 3′
Forward slippage causes deletion
5′ 1 CAG 2 CAG 3 CAG 5 CAG 3′
3′ GTC 1 GTC 2 GTC 3 GTC 5 GTC 6 5′
GTC 4
5′ 1 CAG 2 CAG 3 CAG 5 CAG 6 CAG 3′

http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=hmg.figgrp.1097

- Back-mutation is very possible (i.e., not every ‘mutation’ will lead to a state never seen before)

- Step-wise (‘ladder’) mutation models:
  - #repeats changes by +/- 1. (Weber and Wong 1993, Heyer et al., 1997, observed 11 mutations, all of which were +/-1)

- Mutation rate may depend on #repeats (but not in this paper)

Background

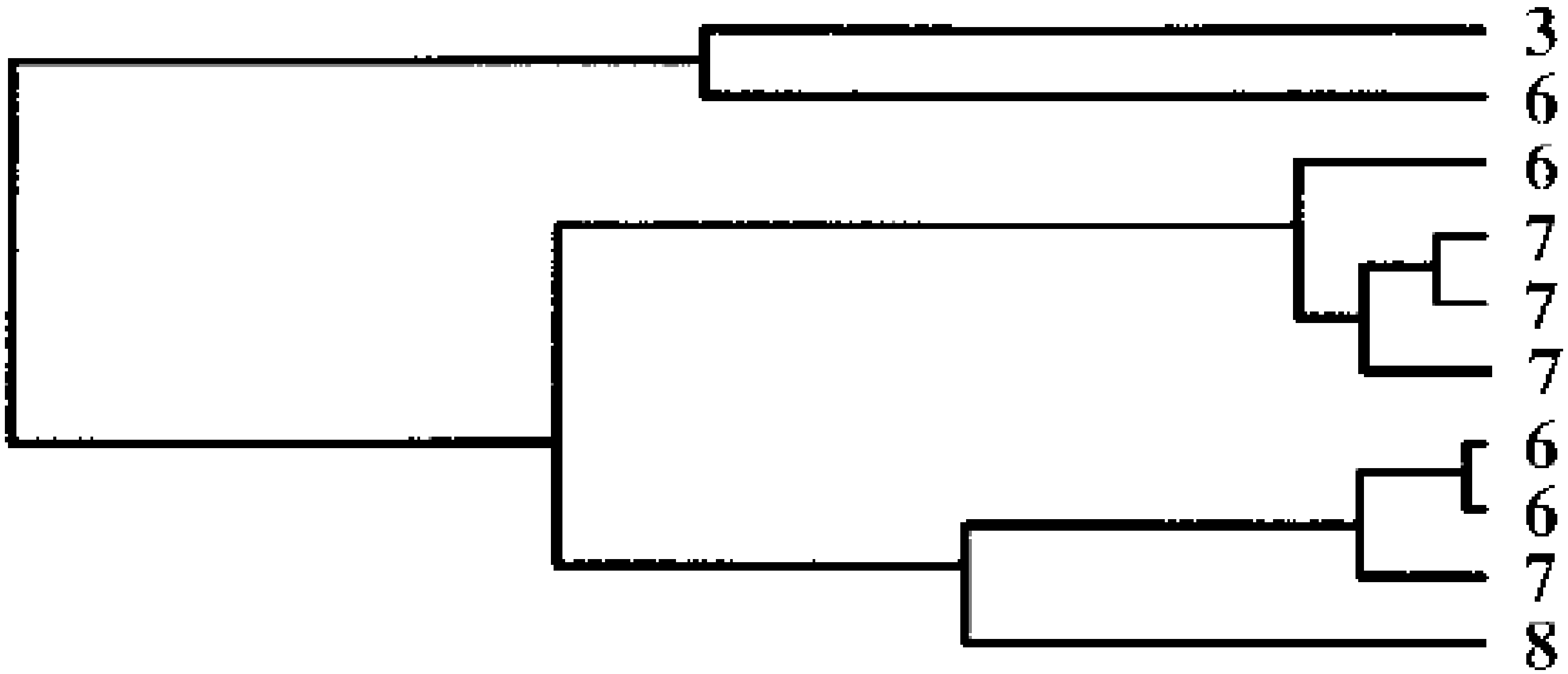
- Traditionally, pop. gen. inference had been based on things such as pairwise differences (Manhattan distance) - i.e., simple things.
- But, pairwise summaries of the data miss much of the info (e.g., genealogy)
- Methods based on ancestral trees were appearing (Felsenstein et al., Tavaré and Griffiths), but these methods ignored back-mutation.

- Data, D:
  - collection of samples from individuals.
  - each sample consists of an observation of the number of repeats at a set of L micro-satellite loci.
- Parameter:
  - Mutation parameter, θ.
  - Model: (Unobserved) ancestral tree showing how the samples are related to each other + action of mutation.

- Goal: to infer the mutation rate.
- Problem: Direct calculation of f(θ|D) is impossible, as is calculation of f(D).
- Solution: Use MCMC.

$$\boxed{f(\theta|D)= f(D|\theta)\pi(\theta)/f(D)}$$

- MCMC algorithm: explores the space of trees that explain the data



(From Figure 1 of the paper)

MCMC

$$\boxed{f(\theta|D)= f(D|\theta)\pi(\theta)/f(D)}$$

Ufeus felsensteini

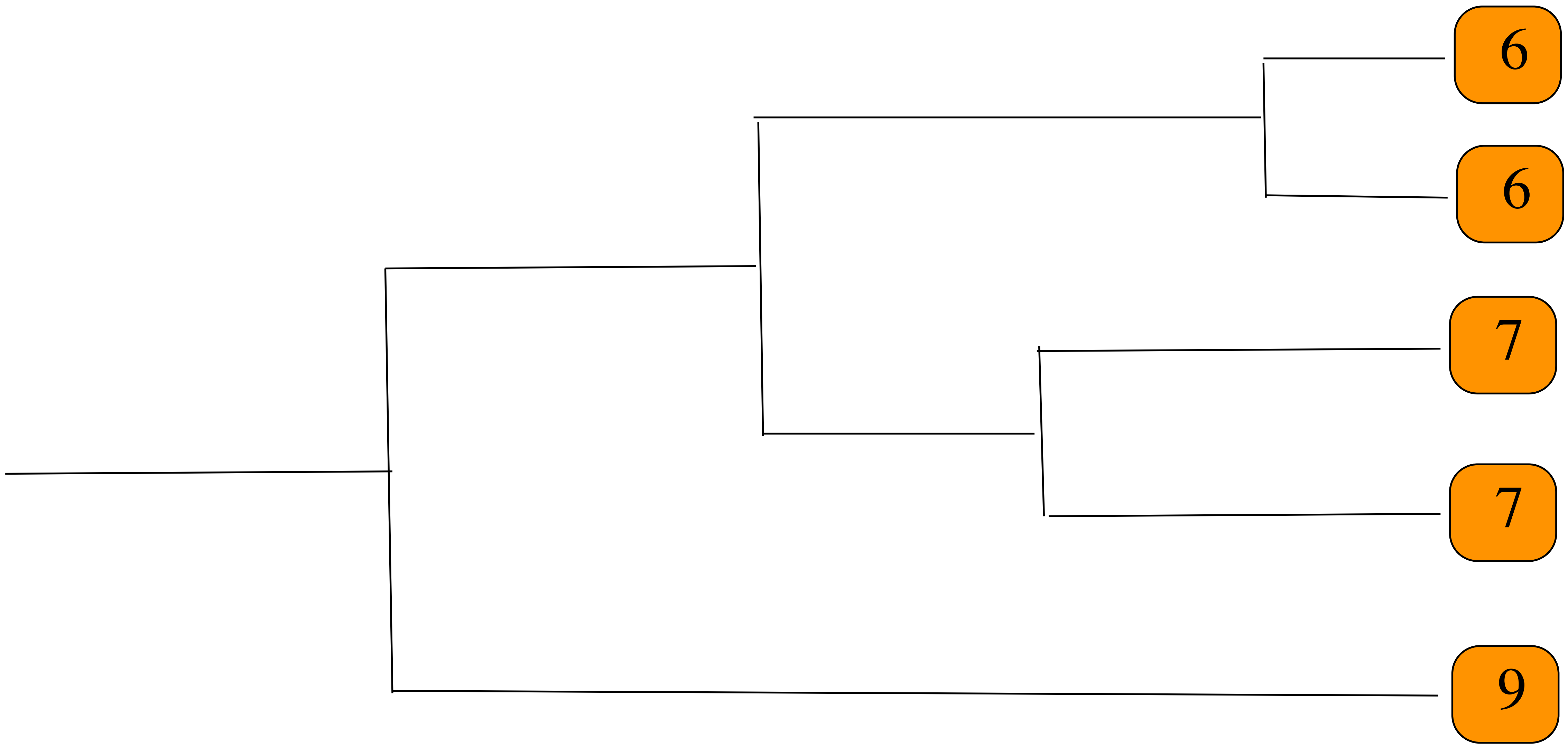
- Use of MCMC requires calculation of $P(D|\theta)$, the prob. of the data given the mutation rate, and a good way of moving around the parameter-space (i.e., a good proposal kernel, q).

- But, calculation of $P(D|\theta)$ directly is impossible, so use an *augmented state-space* that also includes the unobserved coalescent tree of the data: let Y denote the current tree.

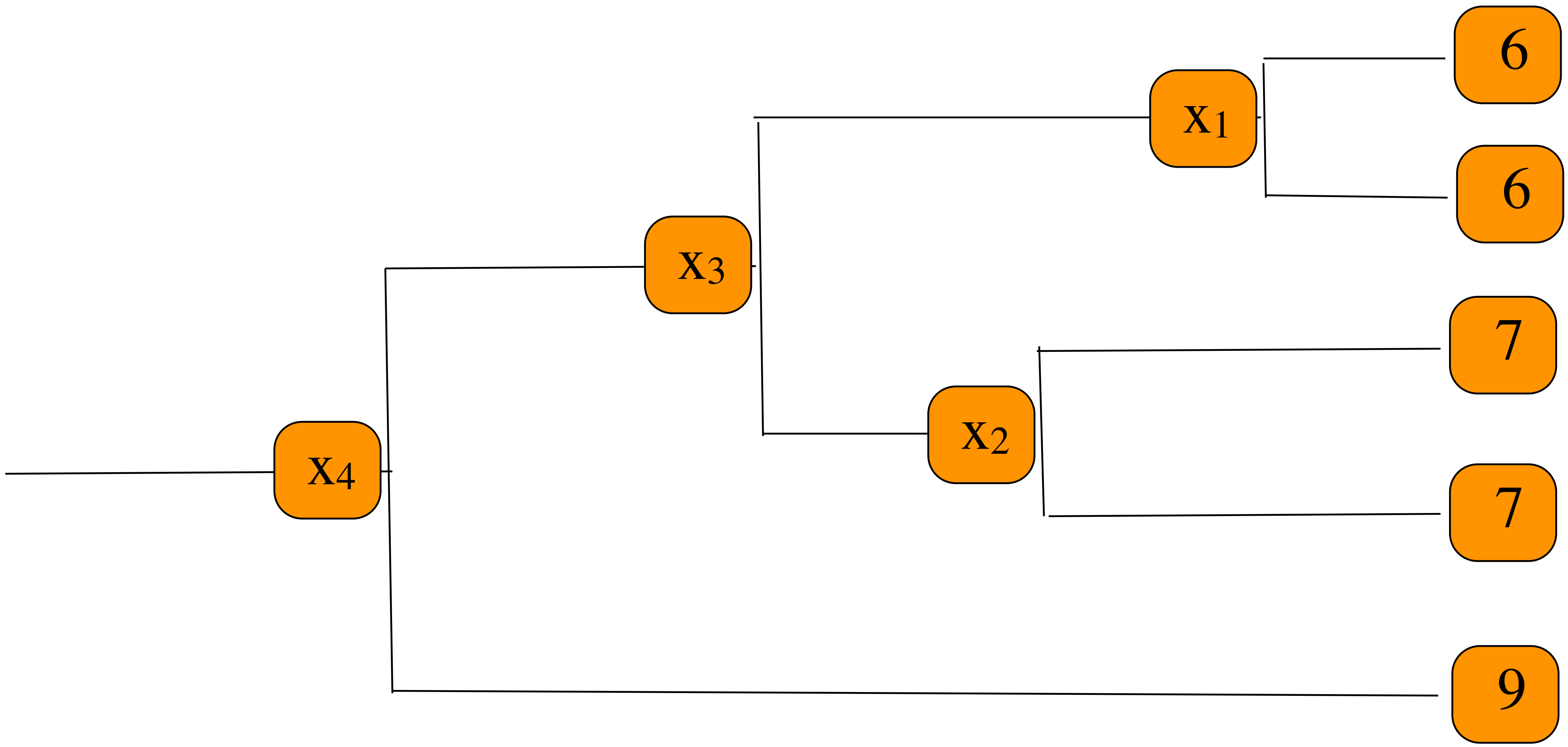
- Now, calculation of $P(D|\theta,Y)$ is possible, but is a difficult (i.e., very time-consuming) calculation [‘Peeling’ algorithm of Felsenstein]

- So, further augment tree Y to also include the state of every node....

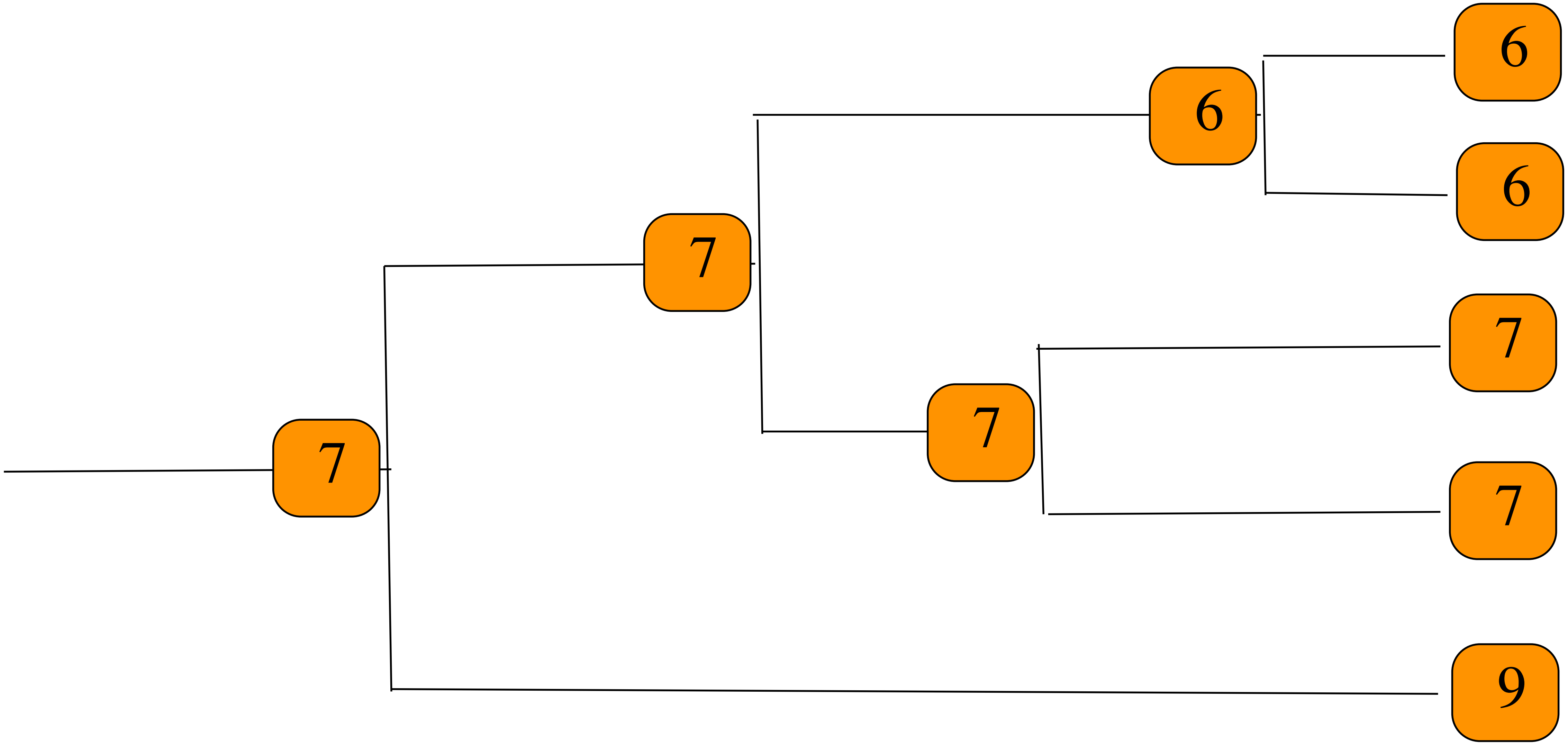
Augmented-tree



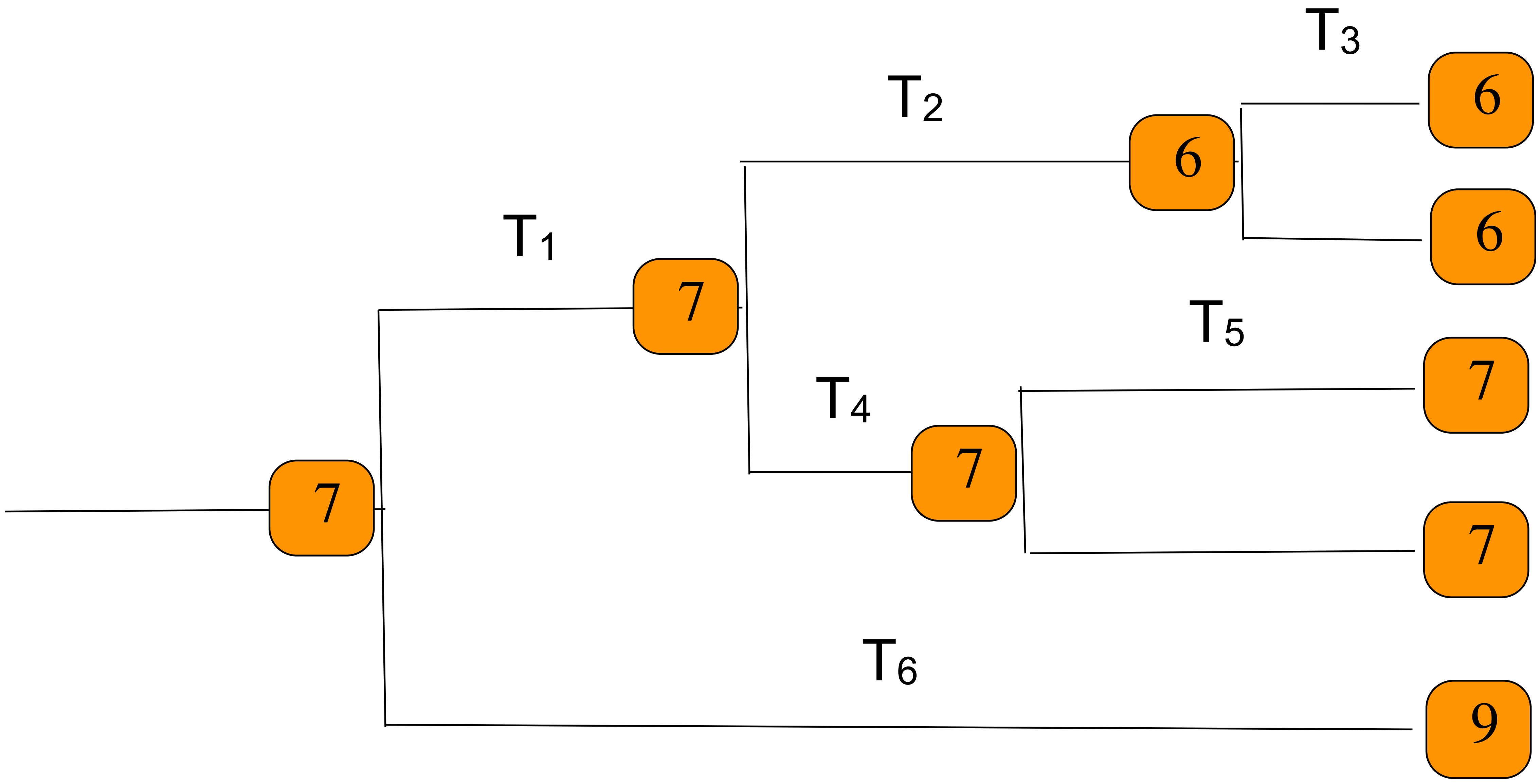
Augmented-tree



Augmented-tree



Augmented-tree



$P(\text{Data}|\text{Tree}) = P(\text{MRCA}=7)P(7\text{->}7|T_1)P(7\text{->}6|T_2)$
$P(6\text{->}6|T_3)^2\, P(7\text{->}7|T_4)P(7\text{->}7|T_5)^2\, P(7\text{->}9|T_6)$

- It remains to calculate P(i -> j | t)...
- Mutations are assumed to be symmetric [P(gain a repeat)=P(lose a repeat)]
- There are an infinite number of possible paths, each of which involves j-i directed mutations (assuming j>i) and d pairs of opposite mutations

e.g. for 5->7 we have paths such as

5->6->7

5->6->5->6->7

5->4->5->6->7->8->7

Equation 3

- Notation: t=time, θ/2= mutation rate
- (#mutns | t) ~ Poisson(tθ/2=λ)
- P(*m* mutns)= $\dfrac{\lambda^m e^{-\lambda}}{m!} = \dfrac{(\theta t/2)^m e^{-\theta t/2}}{m!}$
- P(i -> j | t,θ) = P(|i-j| |t,θ) [they assume symmetry]
- So, define d = |i-j|

Equation 3

$$
\begin{aligned}
P(d \mid t) &= \sum_{k=0}^{\infty} P(2k+d \text{ mutns}|\theta, t) \\
&\qquad \times P(k+d \text{ go `up'}; k \text{ go `down'})(\#\text{choices for the } k \text{ `downs'}) \\
&= \sum_{k=0}^{\infty} \frac{e^{\frac{-\theta t}{2}} \left(\frac{\theta t}{2}\right)^{2k+d}}{(2k+d)!} \left[\left(\frac{1}{2}\right)^{k+d} \left(\frac{1}{2}\right)^{k}\right] \binom{2k+d}{k} \\
&= e^{\frac{-\theta t}{2}} \sum_{k=0}^{\infty} \left(\frac{\theta t}{4}\right)^{2k+d} \frac{1}{(k+d)!d!}
\end{aligned}
$$

Equation 3

$$P(d \mid t) \;=\; \sum_{k=0}^{\infty} P(2k+d \text{ mutns}|\theta, t)$$

$$\times P(k+d \text{ go `up'}; k \text{ go `down'})(\#\text{choices for the } k \text{ `downs'})$$

$$= \sum_{k=0}^{\infty} \frac{e^{\frac{-\theta t}{2}} \left(\frac{\theta t}{2}\right)^{2k+d}}{(2k+d)!} \left[\left(\frac{1}{2}\right)^{k+d} \left(\frac{1}{2}\right)^{k}\right] \binom{2k+d}{k}$$

$$= e^{\frac{-\theta t}{2}} \sum_{k=0}^{\infty} \left(\frac{\theta t}{4}\right)^{2k+d} \frac{1}{(k+d)!d!}$$

Claim: only first few terms matter

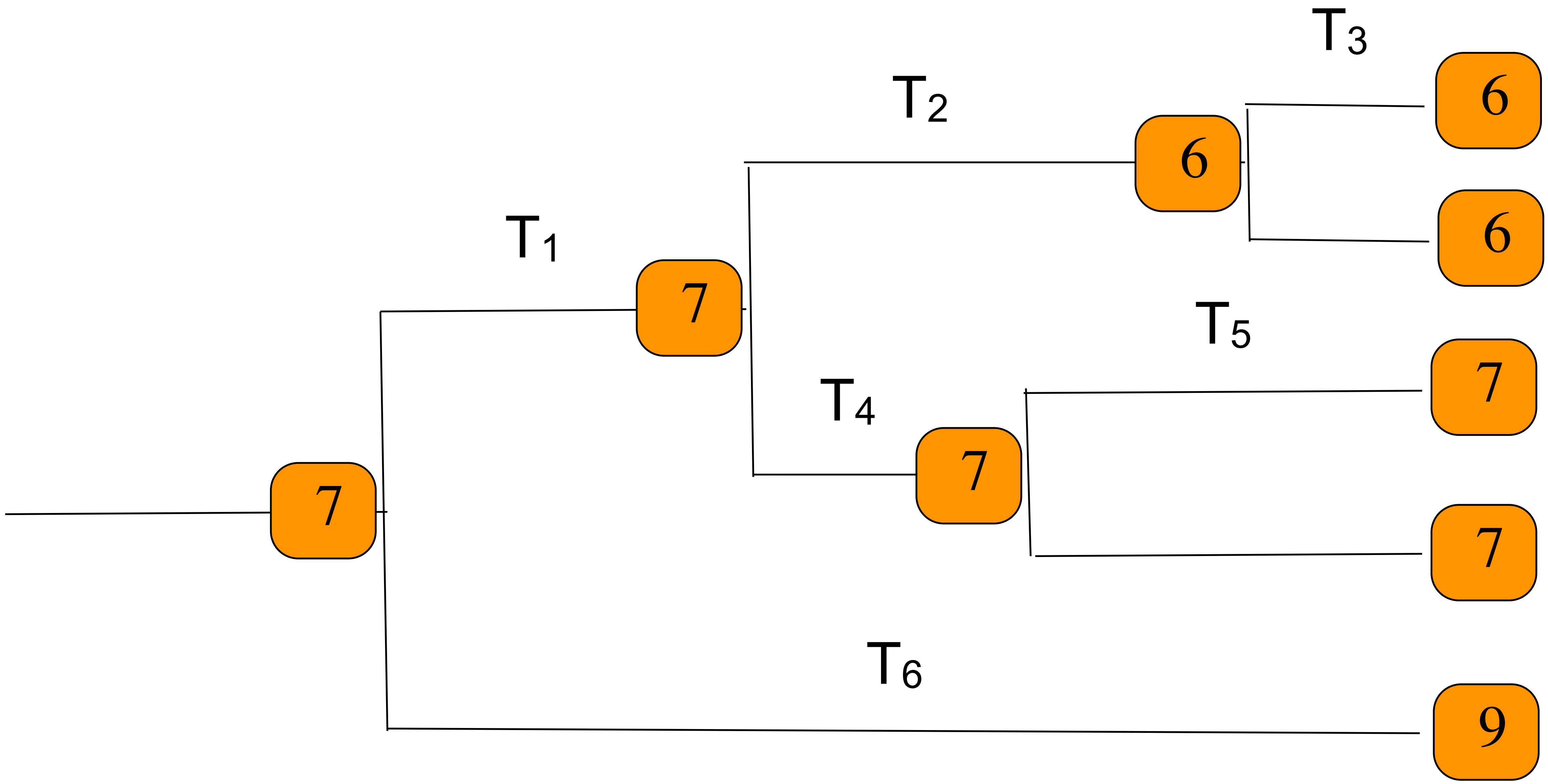
“modified d*th* order Bessel function of the first kind”

Proposal kernel

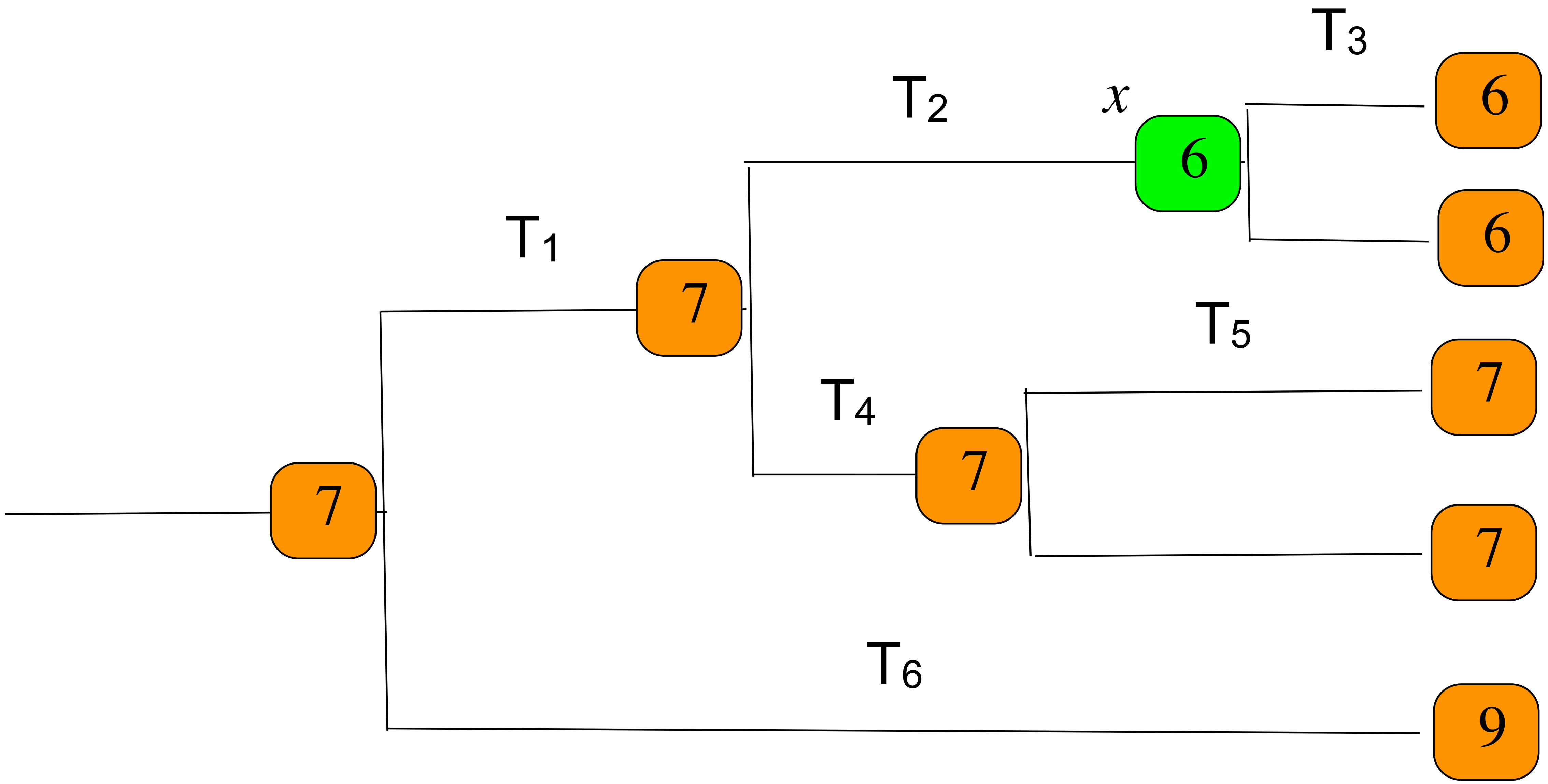
- State-space now consists of:

1. Tree topology
2. Times of events (i.e. nodes) on tree
3. State of ancestor at every node

Proposal kernel q, in pictures...

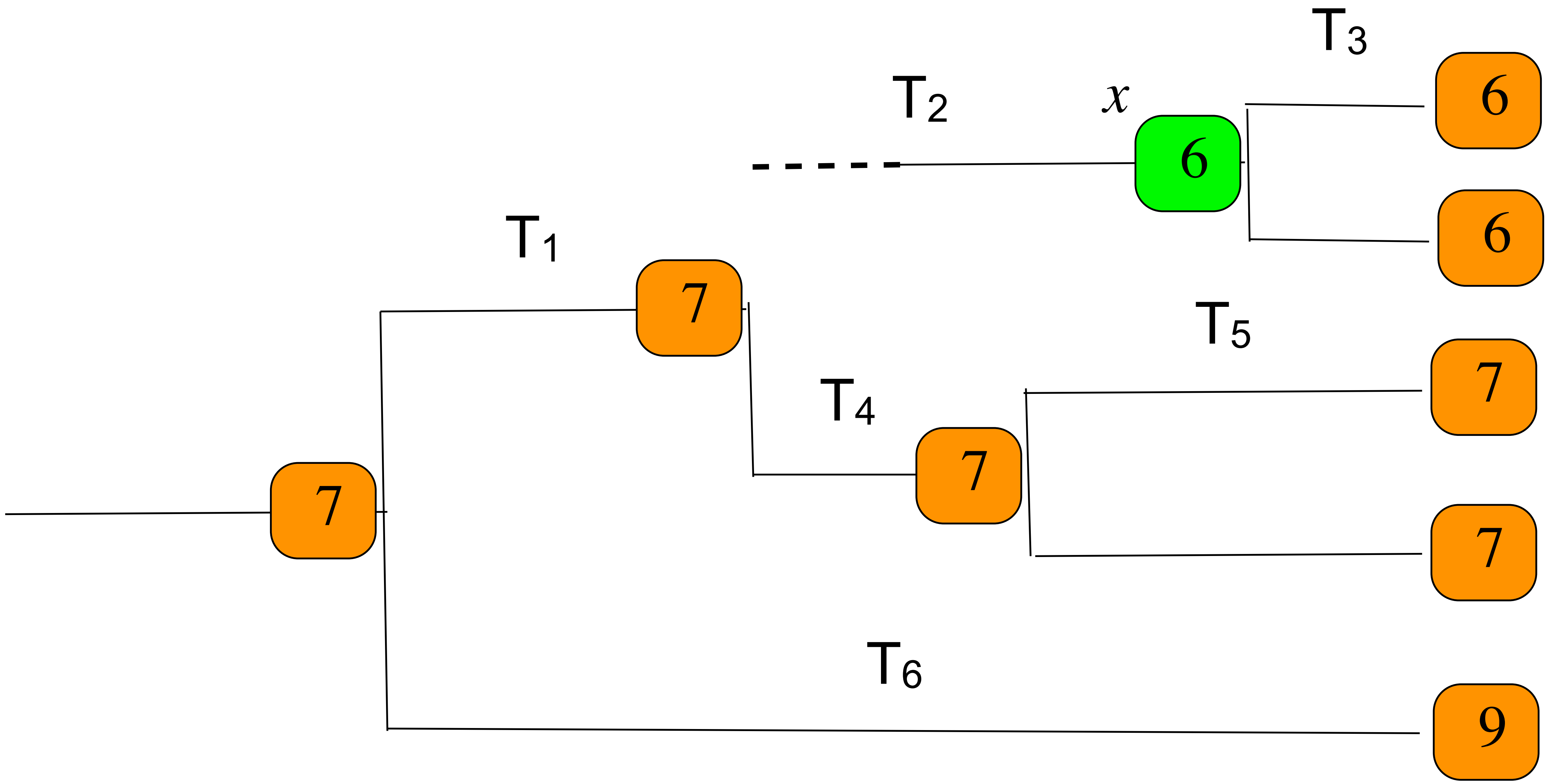


Proposal kernel q, in pictures...



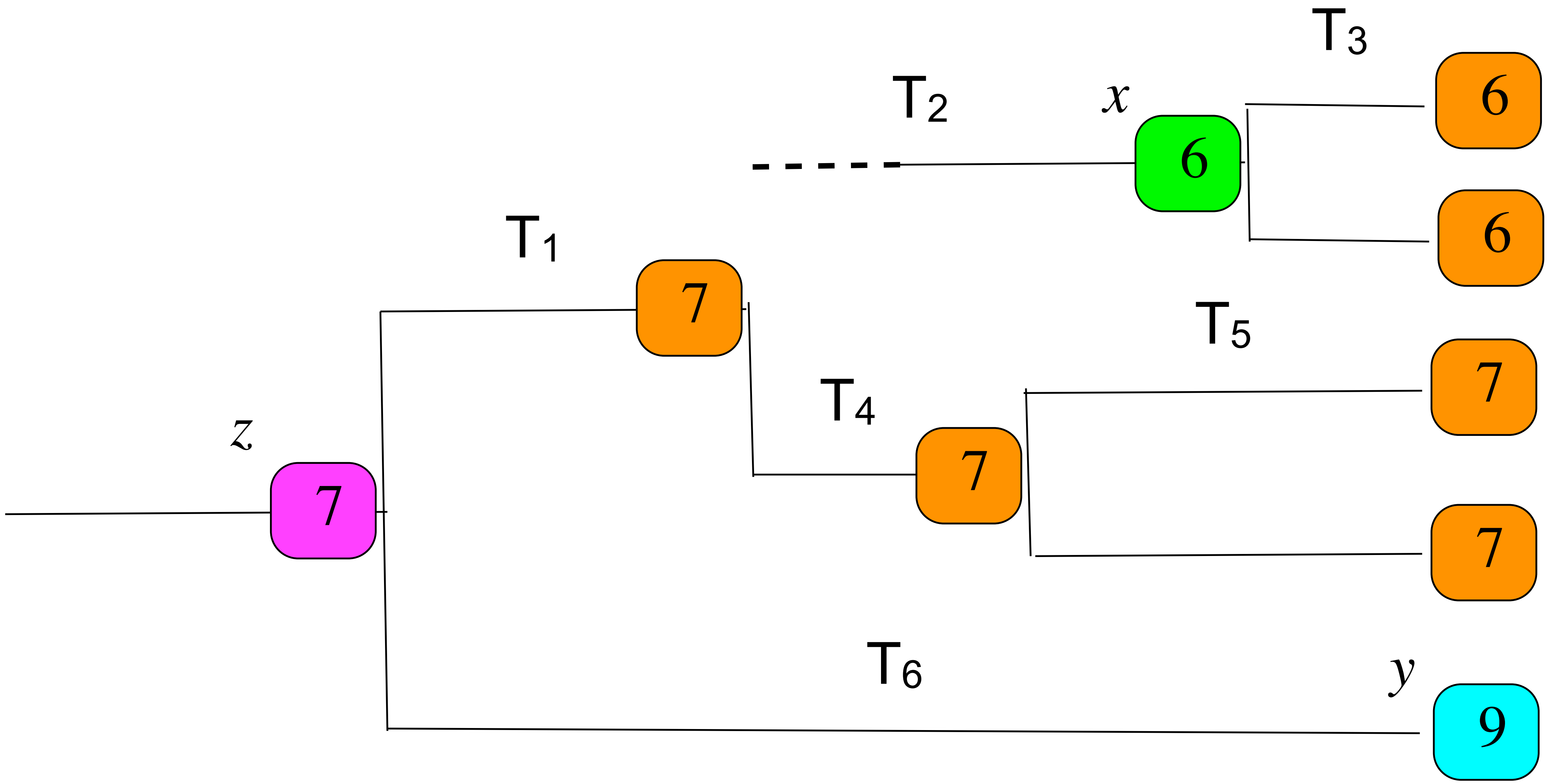
Pick an internal point (i.e. not a 'leaf') at random (call it *x*)

Proposal kernel q, in pictures...



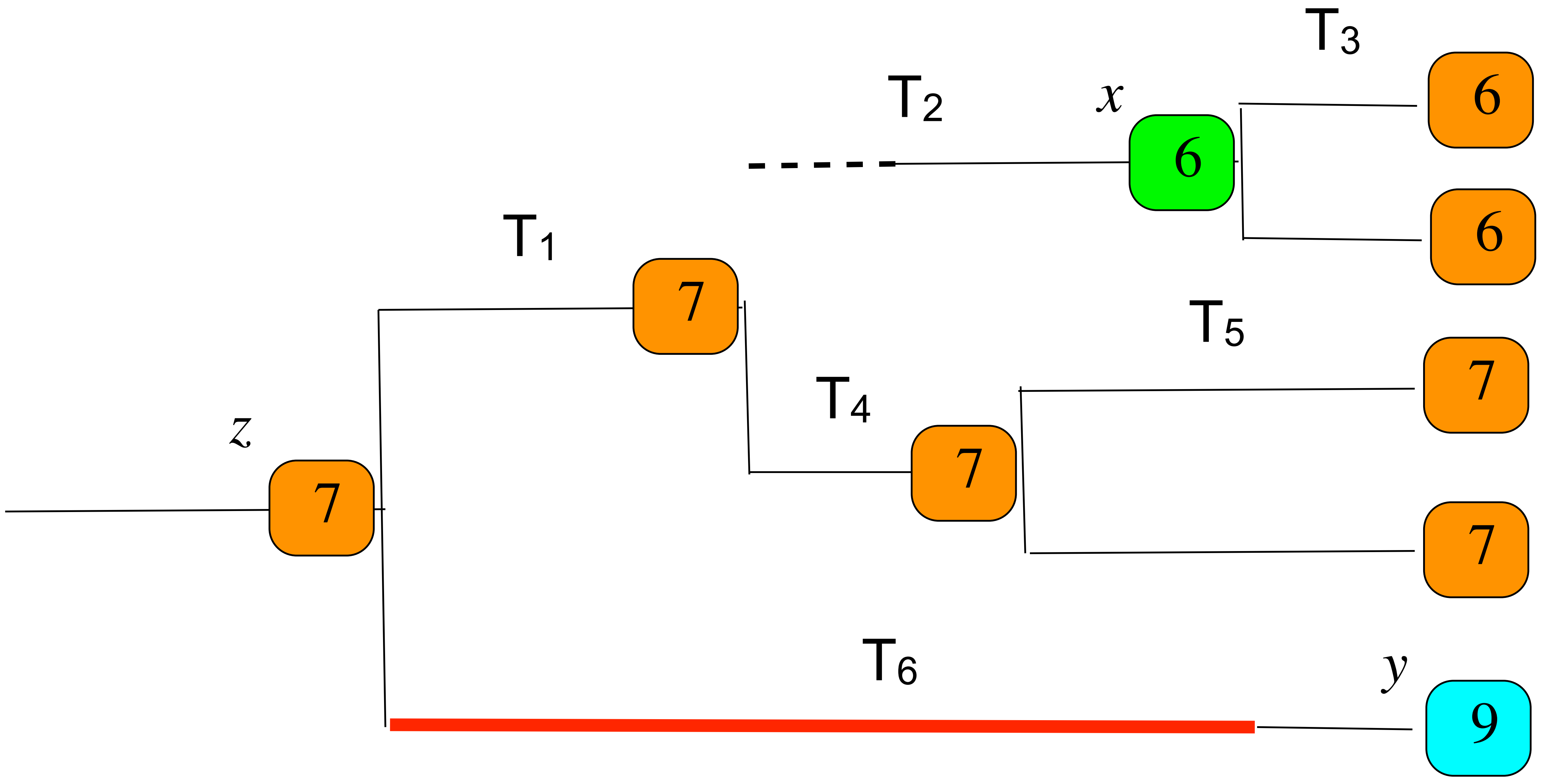
Detach *x* and its subtree

Proposal kernel q, in pictures...



Pick another random node, y, from the remaining portion of the tree, such that its parent, z, is higher up the tree than *x* (*i.e.* *t(z)>t(x)*). [*y* can be the root]

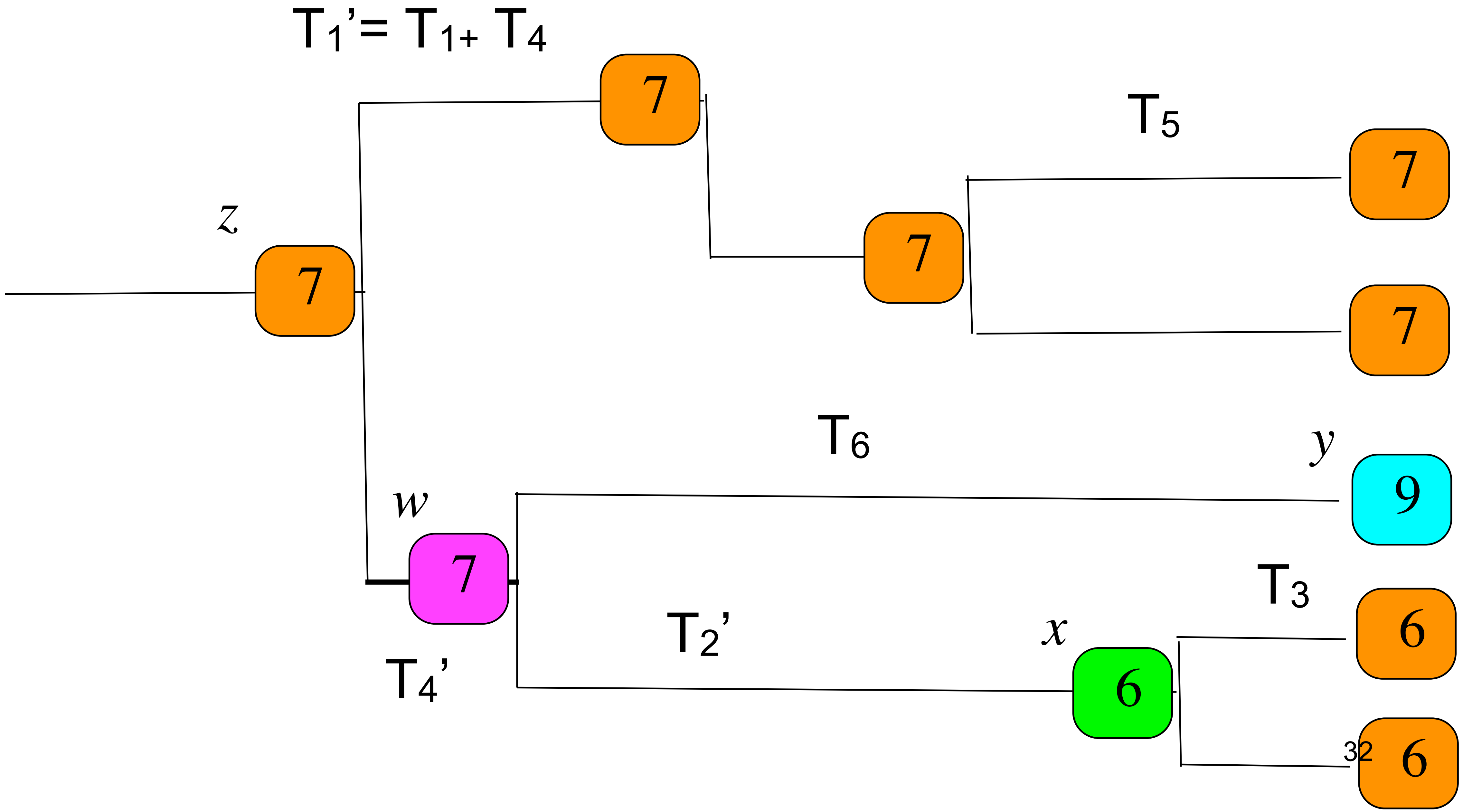
Proposal kernel q, in pictures...



Reattach *x* somewhere between max{*t(x)*,*t(y)*} and *t(z)* [Choose uniformly]

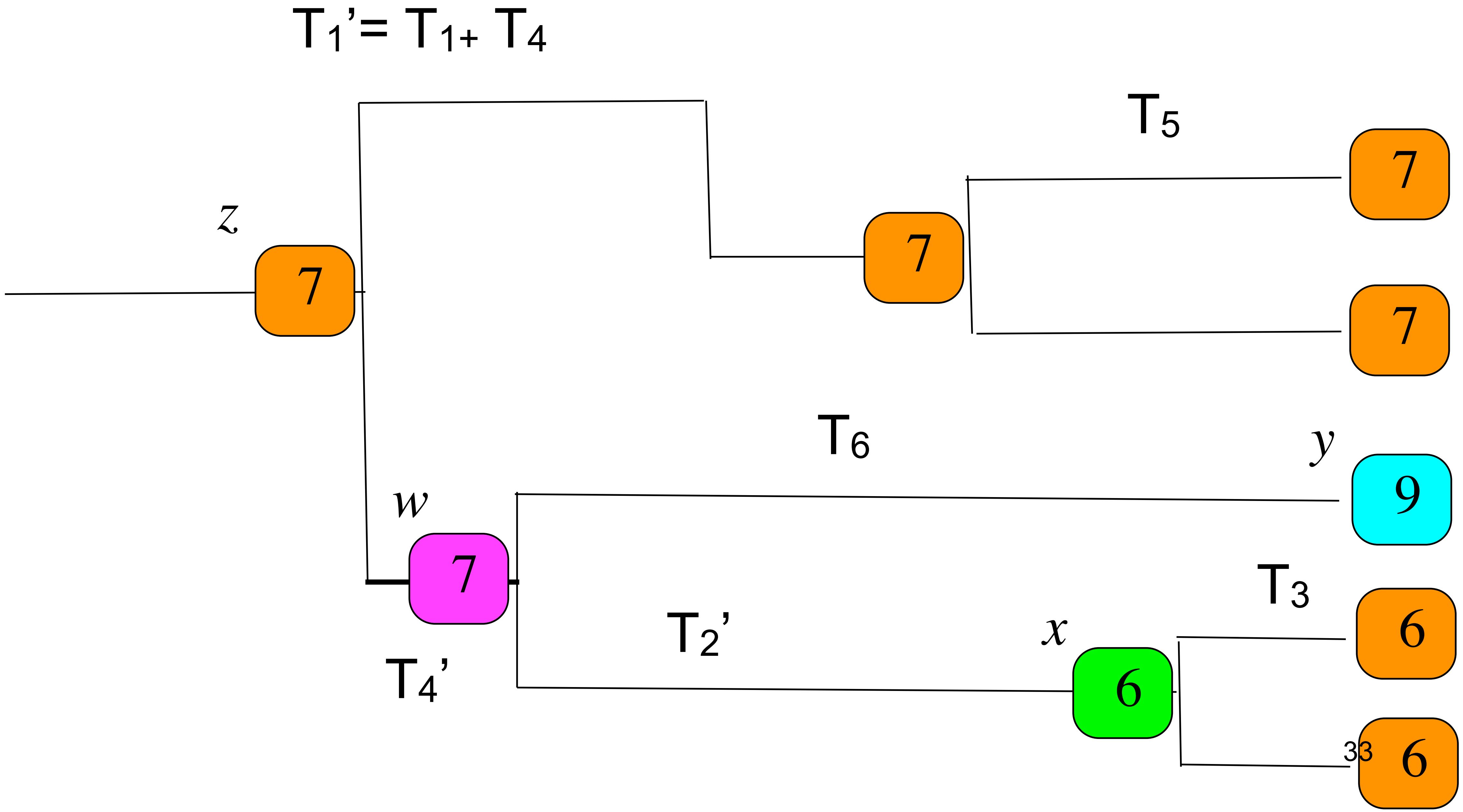
Proposal kernel q, in pictures...

Choose a state for the new node...(remove old node)



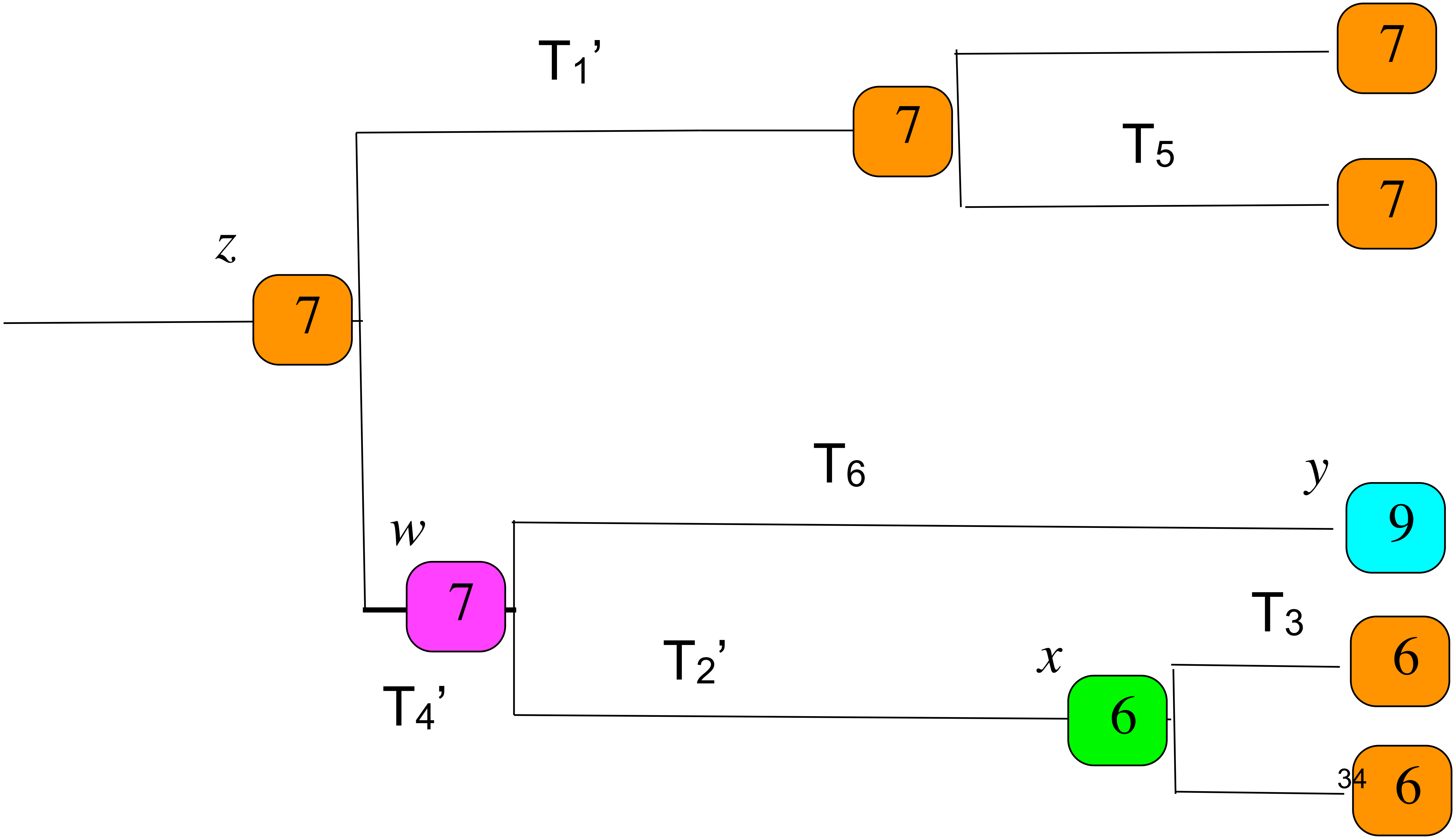
Proposal kernel q, in pictures...

Choose a state for the new node...(remove old node)



Proposal kernel q, in pictures...

Choose a state for the new node...(remove old node)



- Choice of node to attach above (*y*) is not uniform:

$$\mathrm{P}(\mathrm{y} \mid x) \propto \frac{1}{1+|\alpha(x)-\alpha(y)|}$$

where *α(x)* is the state of node x (etc.)

- Choice of state for the new node (*w*):

state ~ Discretized Normal with

mean= [*α(x)*+*α(y)*]/2,

std. dev.= (|*α(x)*-*α(y)*|+1)/4

- These choices are arbitrary **and are likely not the first things they tried!** Coming up with a good proposal kernel takes time. (Bad kernels mix poorly - i.e., the chain stays in the same state for a long time.)

- In order to know that the process will converge to f(θ|D) must have:
  - **Irreducible**: all trees can be reached by a sequence of such changes starting from any other tree
  - **Aperiodic**: There is no ‘period’ (trees can be reached at any iteration i, for i>j (some j)

- These are true for their choice of proposal kernel.

Actual implementation

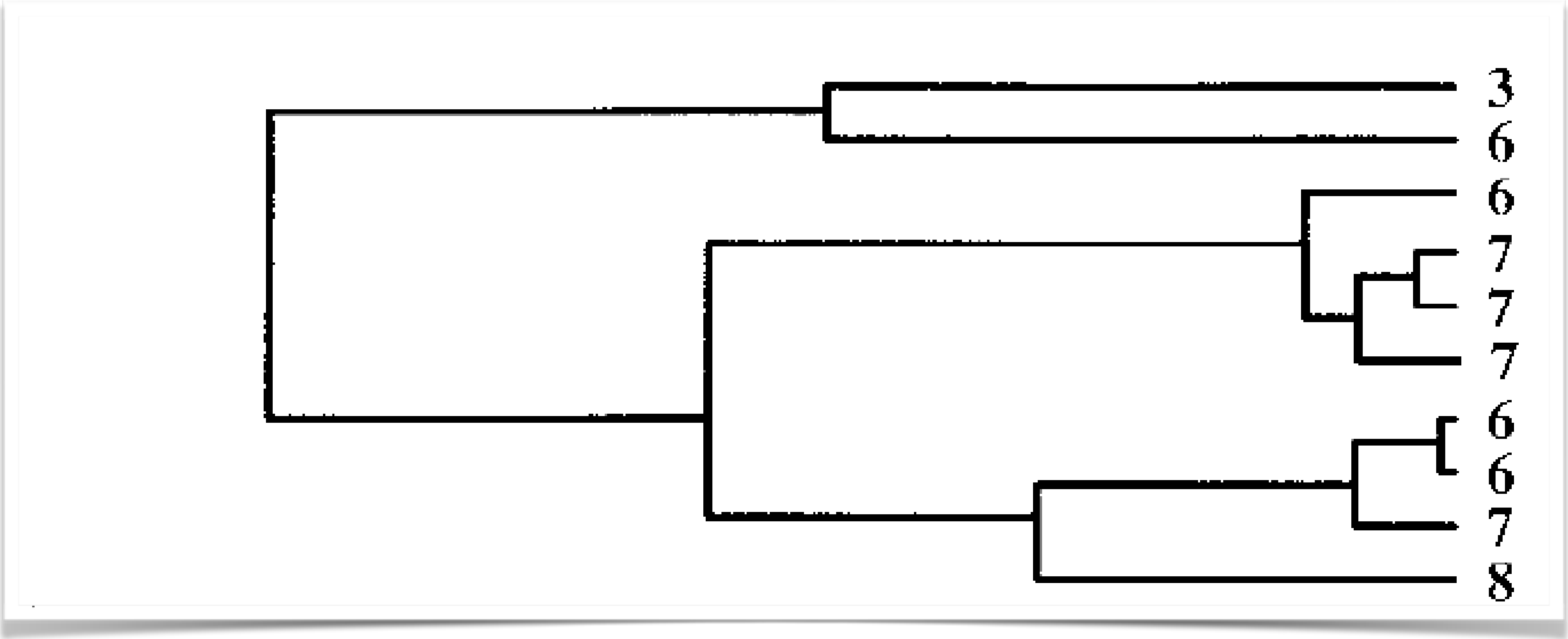
Alternate two proposal steps:

1. Tree topology and state changes: as previously described.

2. Change θ (the mutation parameter.)

Analysis of simulated data (to benchmark performance)

Example (fig 1)



Tree simulated from their model ("coalescent" + step-wise mutation).

Note: Ancestry of the 6's (reconstruction will be challenging!)

θ=5

Height =1.25 coalescent units (prior median =1.54);

Length = 4.82 (prior median =5.21)

Sample iterations:

Not much resemblance to true tree! Why not?

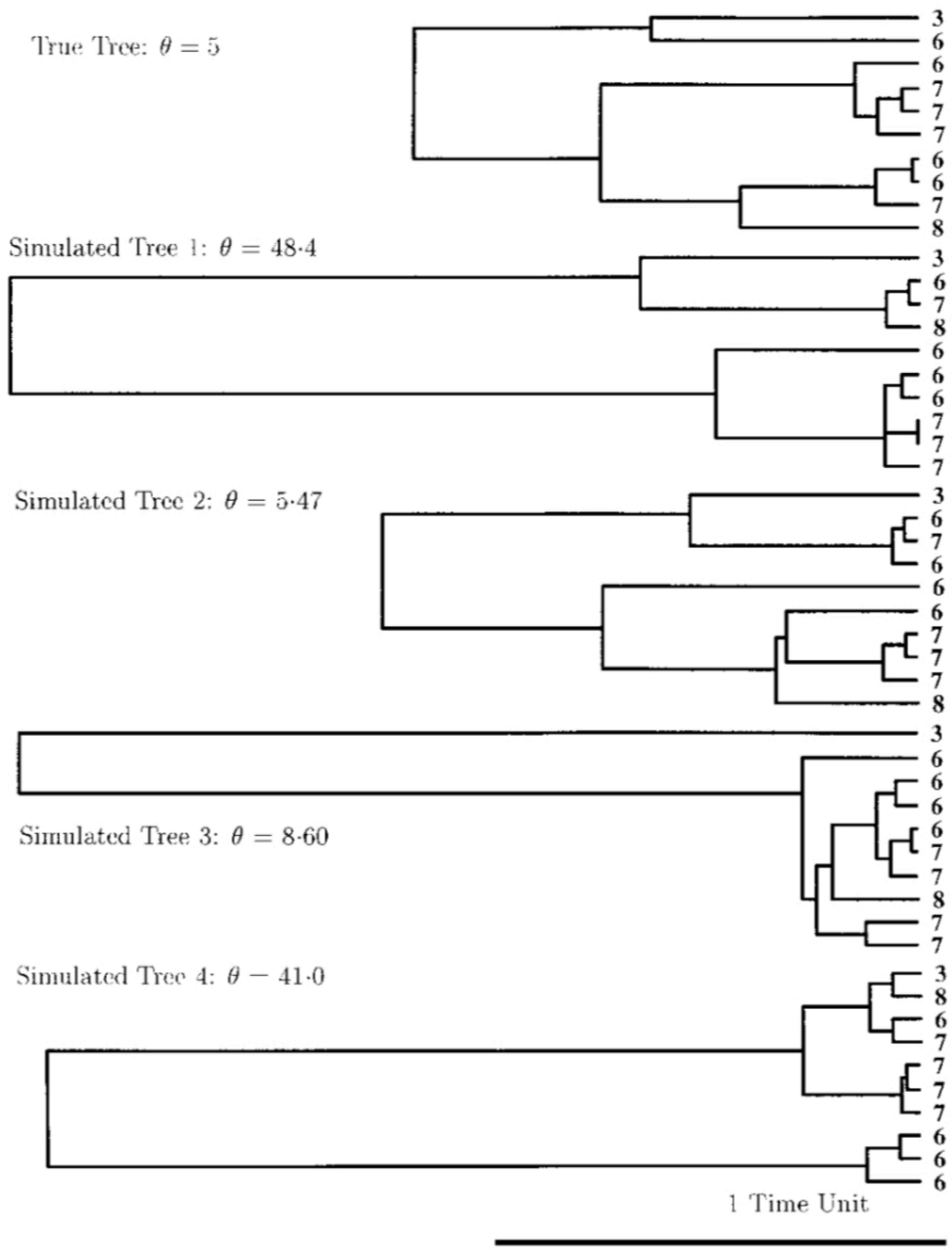

True Tree: $\theta = 5$
3 6 6 7 7 7 6 6 7 8
Simulated Tree 1: $\theta = 48{\cdot}4$
3 6 7 8 6 6 6 7 7 7
Simulated Tree 2: $\theta = 5{\cdot}47$
3 6 7 6 6 6 7 7 7 8
Simulated Tree 3: $\theta = 8{\cdot}60$
3 6 6 6 6 7 7 8 7 7
Simulated Tree 4: $\theta - 41{\cdot}0$
3 8 6 7 7 7 7 6 6 6
1 Time Unit

Figure 1.—The top tree ("true") is simulated from the coalescent-with-ladder-mutation model with θ = 5. The other four trees are simulated from the postdata distribution given the allelic data of the true tree. These trees are samples numbered 2000, 4000, 6000, and 8000 from the MCMC run corresponding to row 1 of Table 1.

Results:

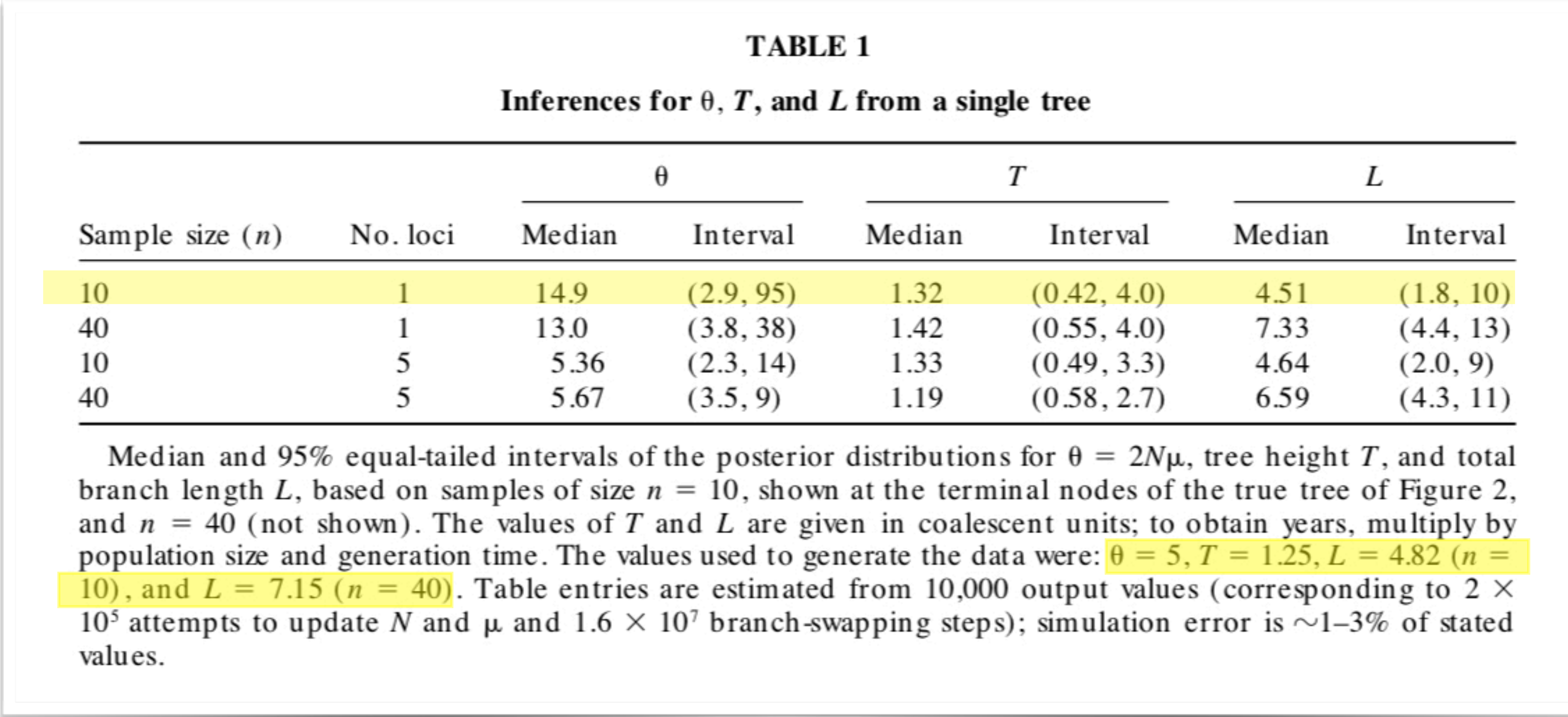


TABLE 1

Inferences for θ, *T*, and *L* from a single tree

		θ		*T*		*L*	
Sample size (*n*)	No. loci	Median	Interval	Median	Interval	Median	Interval
10	1	14.9	(2.9, 95)	1.32	(0.42, 4.0)	4.51	(1.8, 10)
40	1	13.0	(3.8, 38)	1.42	(0.55, 4.0)	7.33	(4.4, 13)
10	5	5.36	(2.3, 14)	1.33	(0.49, 3.3)	4.64	(2.0, 9)
40	5	5.67	(3.5, 9)	1.19	(0.58, 2.7)	6.59	(4.3, 11)

Median and 95% equal-tailed intervals of the posterior distributions for $\theta = 2N\mu$, tree height T, and total branch length L, based on samples of size $n = 10$, shown at the terminal nodes of the true tree of Figure 2, and $n = 40$ (not shown). The values of T and L are given in coalescent units; to obtain years, multiply by population size and generation time. The values used to generate the data were: $\theta = 5$, $T = 1.25$, $L = 4.82$ ($n = 10$), and $L = 7.15$ ($n = 40$). Table entries are estimated from 10,000 output values (corresponding to 2×10^5 attempts to update N and μ and 1.6×10^7 branch-swapping steps); simulation error is ~1–3% of stated values.

Row 1: Analysis of single tree of size 10

All posteriors have large variance

Results:

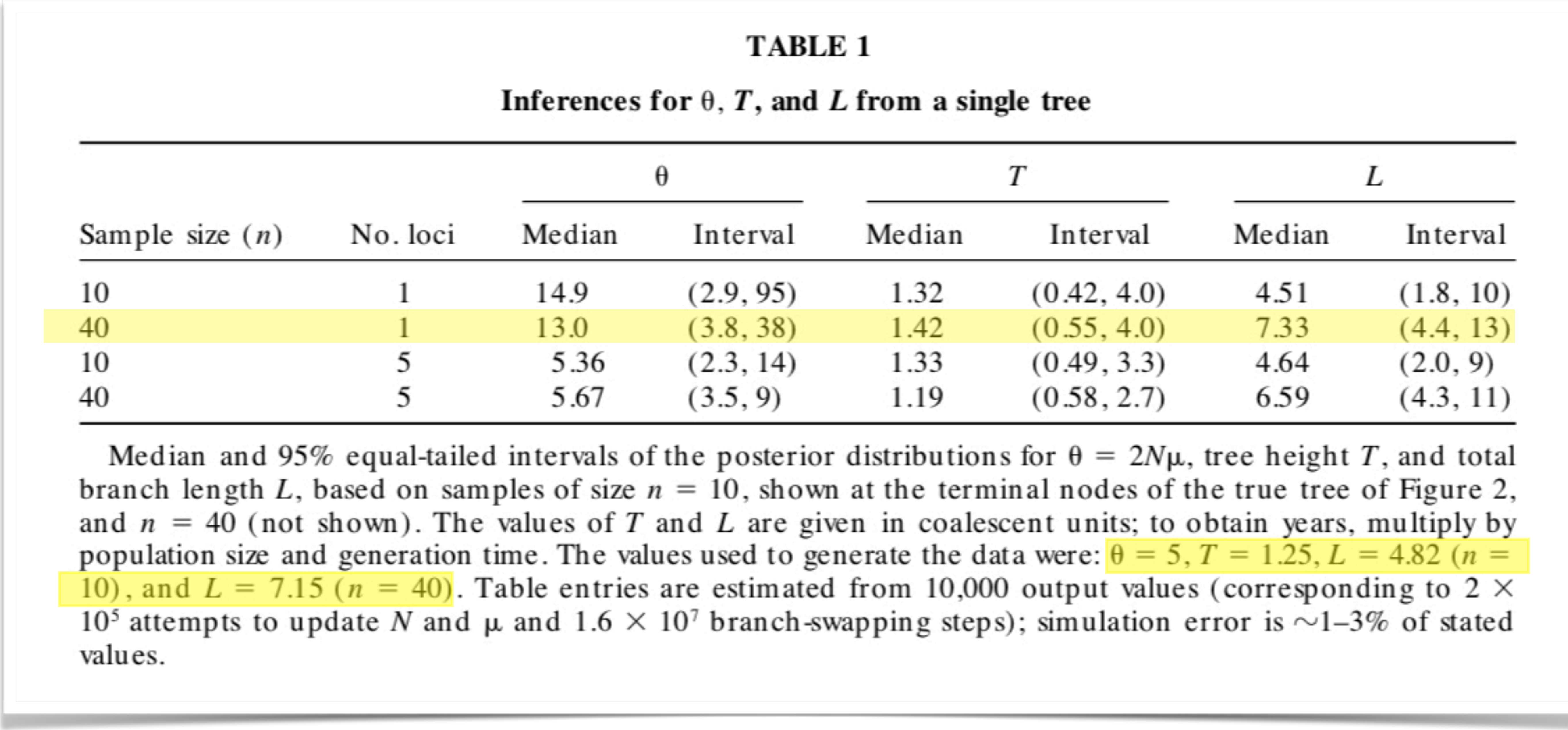


TABLE 1

Inferences for θ, *T*, and *L* from a single tree

		θ		*T*		*L*	
Sample size (n)	No. loci	Median	Interval	Median	Interval	Median	Interval
10	1	14.9	(2.9, 95)	1.32	(0.42, 4.0)	4.51	(1.8, 10)
40	1	13.0	(3.8, 38)	1.42	(0.55, 4.0)	7.33	(4.4, 13)
10	5	5.36	(2.3, 14)	1.33	(0.49, 3.3)	4.64	(2.0, 9)
40	5	5.67	(3.5, 9)	1.19	(0.58, 2.7)	6.59	(4.3, 11)

Median and 95% equal-tailed intervals of the posterior distributions for $\theta = 2N\mu$, tree height T, and total branch length L, based on samples of size $n = 10$, shown at the terminal nodes of the true tree of Figure 2, and $n = 40$ (not shown). The values of T and L are given in coalescent units; to obtain years, multiply by population size and generation time. The values used to generate the data were: $\theta = 5$, $T = 1.25$, $L = 4.82$ ($n = 10$), and $L = 7.15$ ($n = 40$). Table entries are estimated from 10,000 output values (corresponding to 2×10^5 attempts to update N and μ and 1.6×10^7 branch-swapping steps); simulation error is ~1–3% of stated values.

Row 2: Tree of 40 indivs.

Variance for θ reduced, other variances largely unchanged (Tree height T was actually the same)

5 completely linked micro-satellite loci:

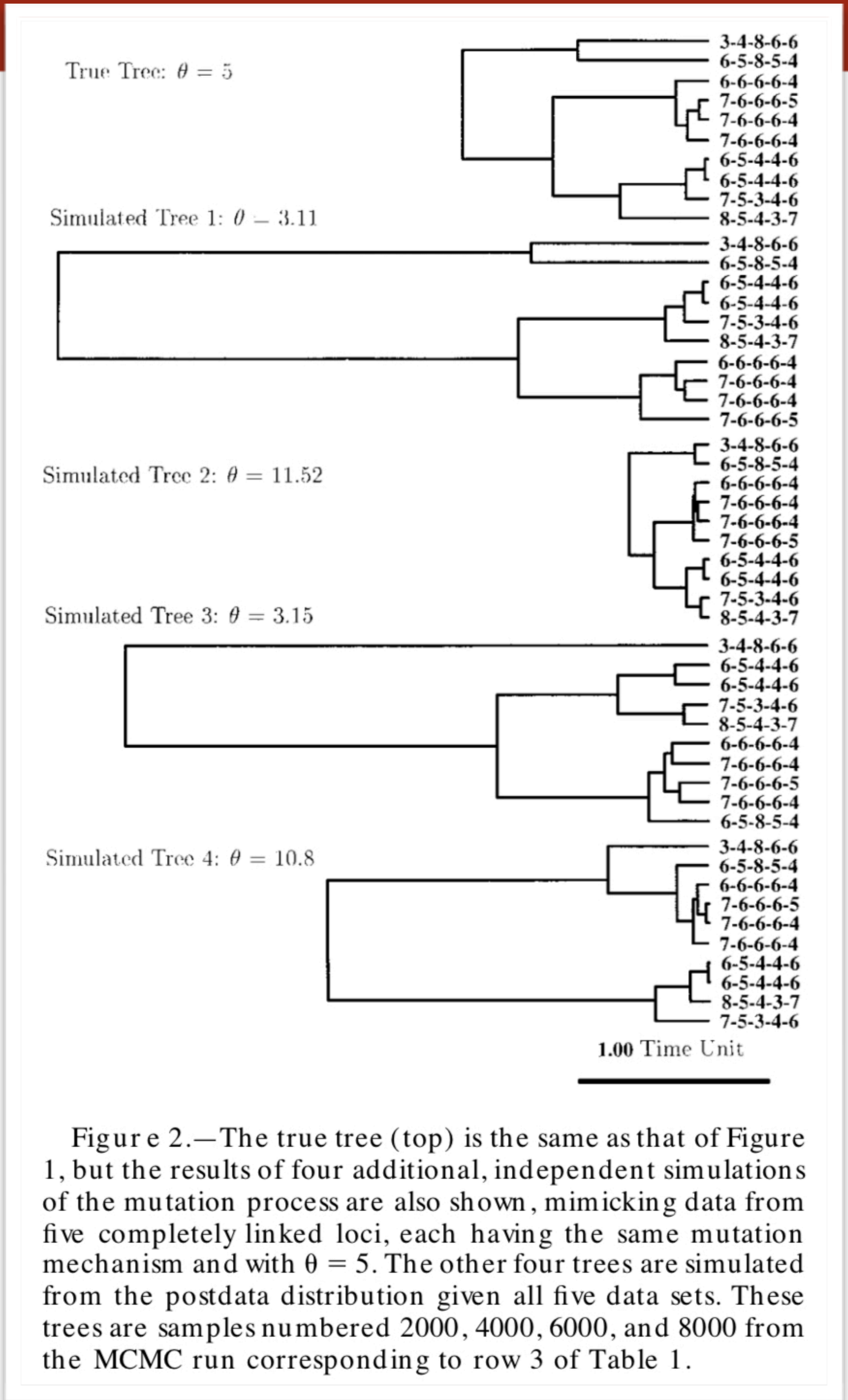


Figure 2.—The true tree (top) is the same as that of Figure 1, but the results of four additional, independent simulations of the mutation process are also shown, mimicking data from five completely linked loci, each having the same mutation mechanism and with $\theta = 5$. The other four trees are simulated from the postdata distribution given all five data sets. These trees are samples numbered 2000, 4000, 6000, and 8000 from the MCMC run corresponding to row 3 of Table 1.

Results:

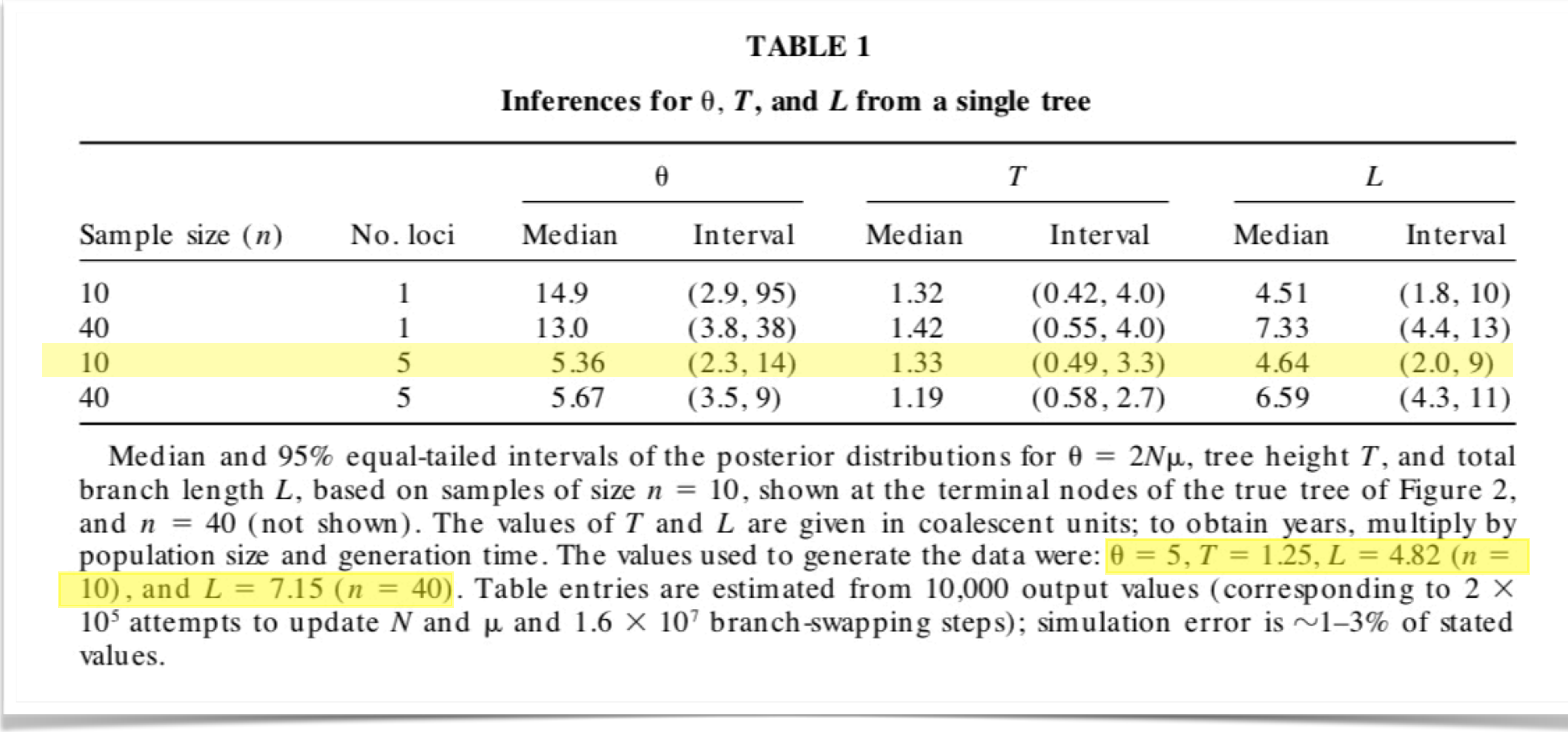


TABLE 1

Inferences for θ, T, and L from a single tree

		θ		T		L	
Sample size (n)	No. loci	Median	Interval	Median	Interval	Median	Interval
10	1	14.9	(2.9, 95)	1.32	(0.42, 4.0)	4.51	(1.8, 10)
40	1	13.0	(3.8, 38)	1.42	(0.55, 4.0)	7.33	(4.4, 13)
10	5	5.36	(2.3, 14)	1.33	(0.49, 3.3)	4.64	(2.0, 9)
40	5	5.67	(3.5, 9)	1.19	(0.58, 2.7)	6.59	(4.3, 11)

Median and 95% equal-tailed intervals of the posterior distributions for $\theta = 2N\mu$, tree height T, and total branch length L, based on samples of size $n = 10$, shown at the terminal nodes of the true tree of Figure 2, and $n = 40$ (not shown). The values of T and L are given in coalescent units; to obtain years, multiply by population size and generation time. The values used to generate the data were: $\theta = 5$, $T = 1.25$, $L = 4.82$ ($n = 10$), and $L = 7.15$ ($n = 40$). Table entries are estimated from 10,000 output values (corresponding to 2×10^5 attempts to update N and μ and 1.6×10^7 branch-swapping steps); simulation error is ~1–3% of stated values.

Row 3: 5 linked loci, sample size=10

Results:

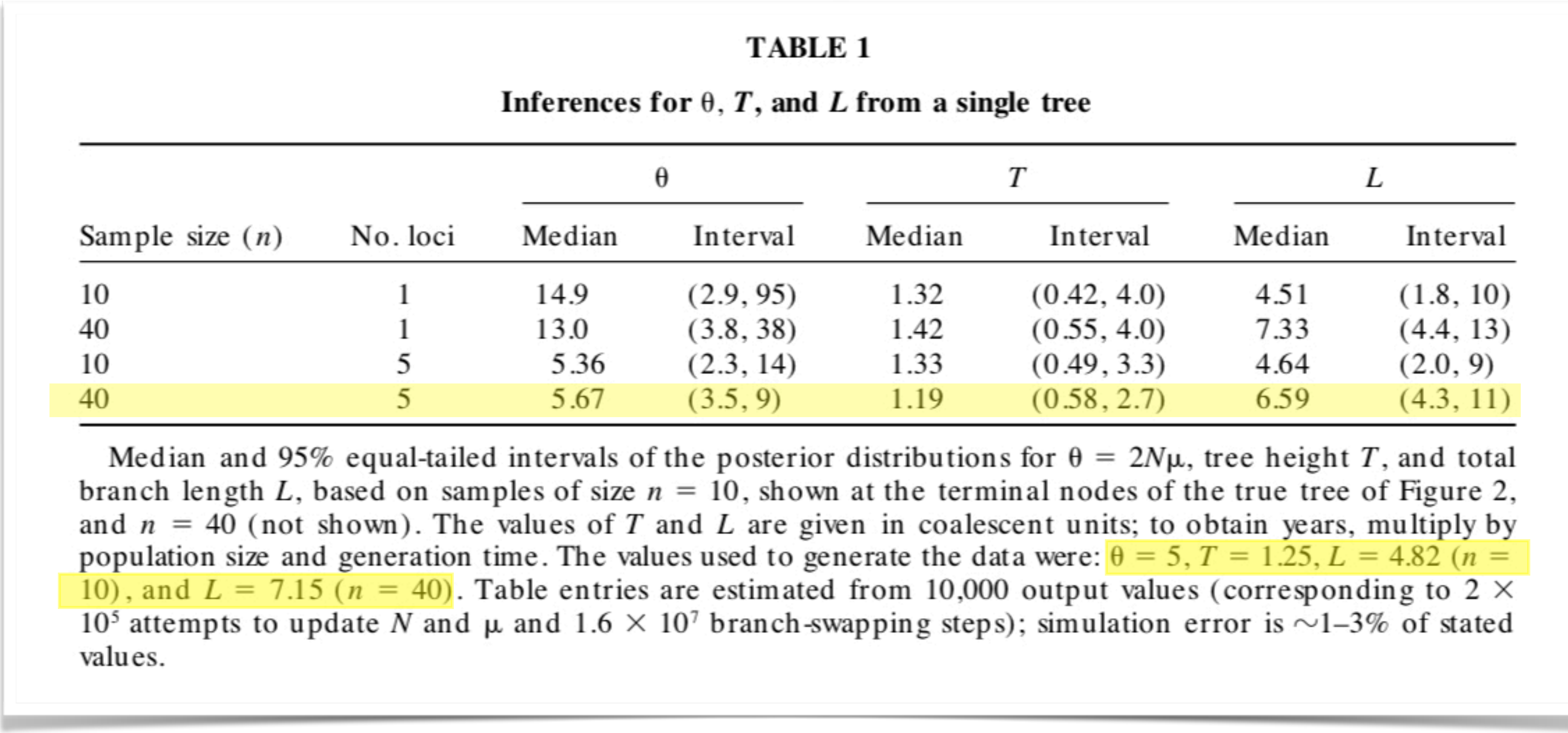


TABLE 1

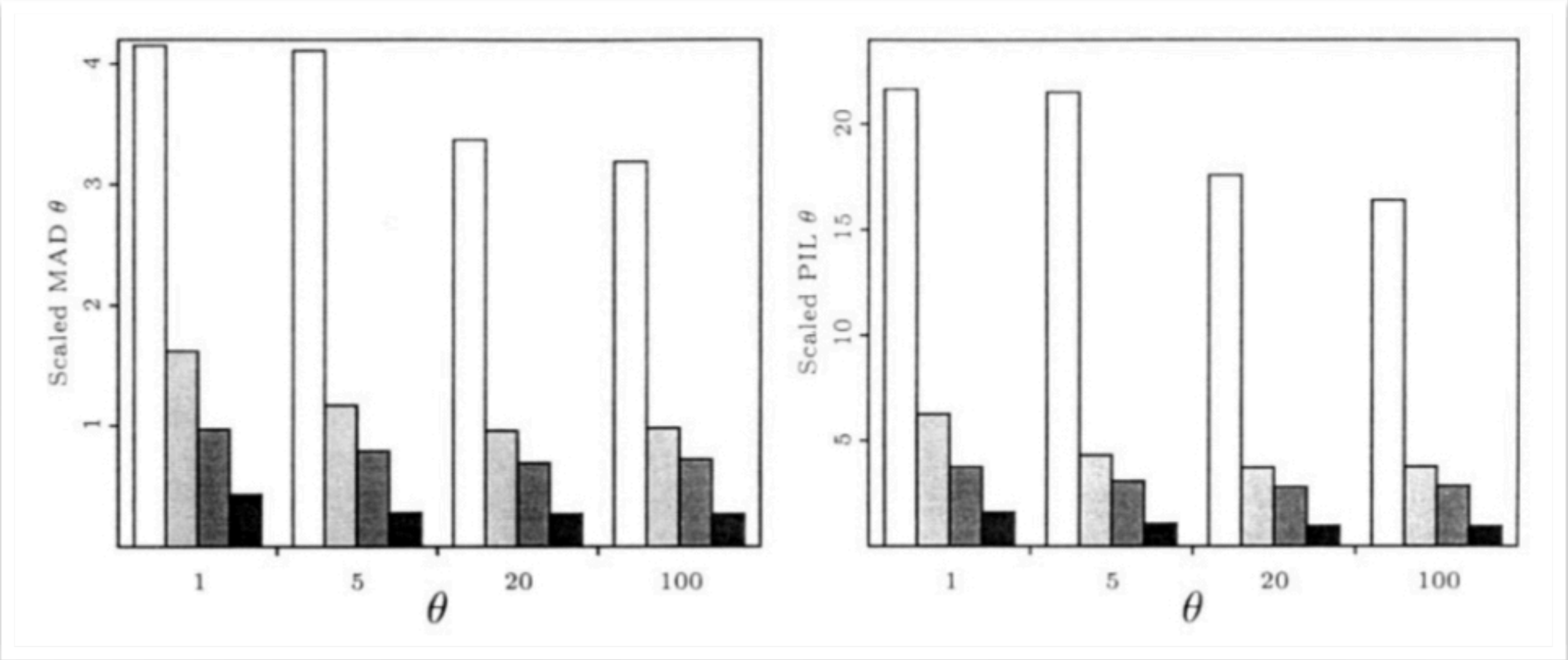
Inferences for θ, *T*, and *L* from a single tree

		θ		*T*		*L*	
Sample size (*n*)	No. loci	Median	Interval	Median	Interval	Median	Interval
10	1	14.9	(2.9, 95)	1.32	(0.42, 4.0)	4.51	(1.8, 10)
40	1	13.0	(3.8, 38)	1.42	(0.55, 4.0)	7.33	(4.4, 13)
10	5	5.36	(2.3, 14)	1.33	(0.49, 3.3)	4.64	(2.0, 9)
40	5	5.67	(3.5, 9)	1.19	(0.58, 2.7)	6.59	(4.3, 11)

Median and 95% equal-tailed intervals of the posterior distributions for $\theta = 2N\mu$, tree height T, and total branch length L, based on samples of size $n = 10$, shown at the terminal nodes of the true tree of Figure 2, and $n = 40$ (not shown). The values of T and L are given in coalescent units; to obtain years, multiply by population size and generation time. The values used to generate the data were: $\theta = 5$, $T = 1.25$, $L = 4.82$ ($n = 10$), and $L = 7.15$ ($n = 40$). Table entries are estimated from 10,000 output values (corresponding to 2×10^5 attempts to update N and μ and 1.6×10^7 branch-swapping steps); simulation error is ~1–3% of stated values.

Row 4: 5 linked loci, sample size=40

Figure 3: results averaged over 140 data sets for θ:

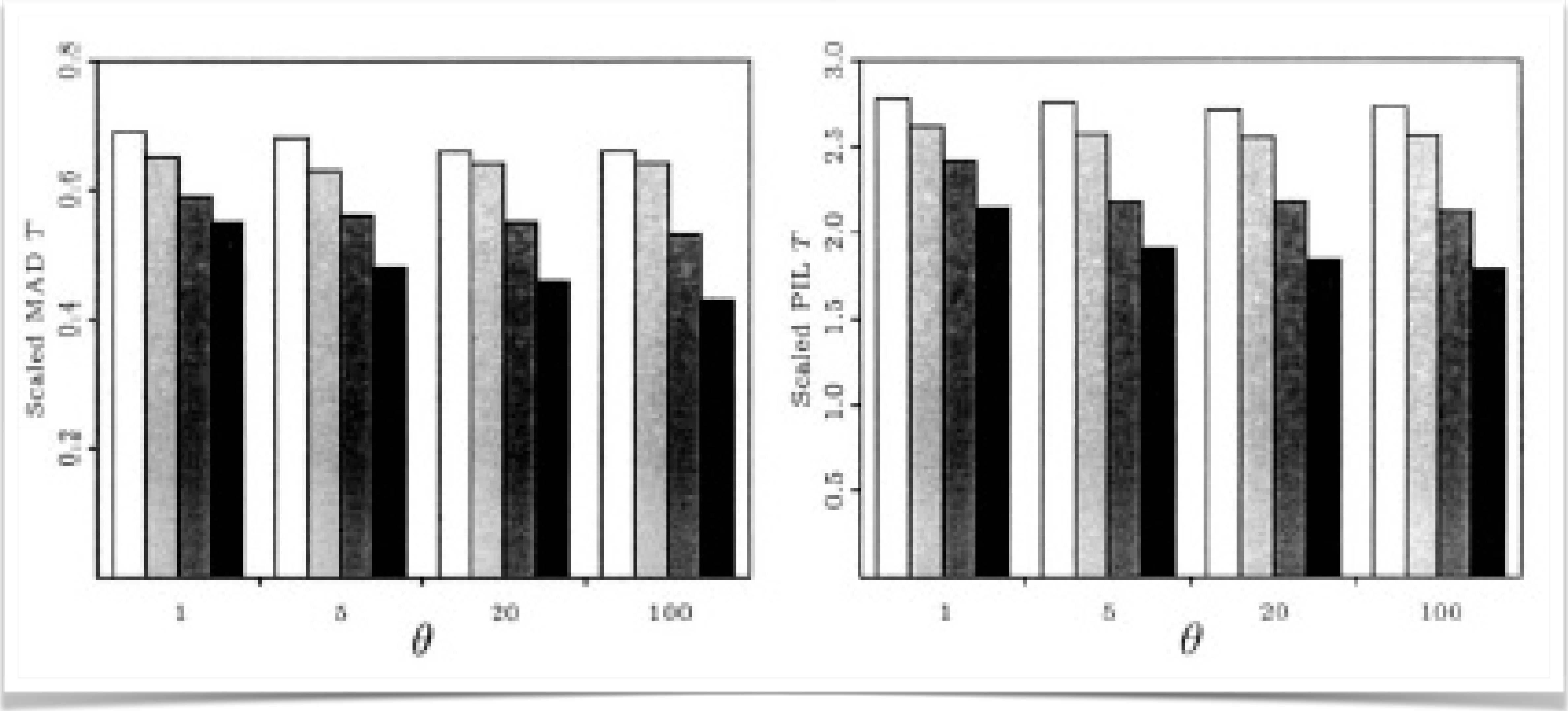


Mean absolute difference [MAD] and 95% probability interval length [PIL] for θ (scaled by true θ)

Bar 1 = single locus, n=10; Bar 2 = single locus, n=40

Bar 3 = 5 loci, n=10; Bar 4 = 5 loci, n=40.

Averaged over 140 data sets: for *T*



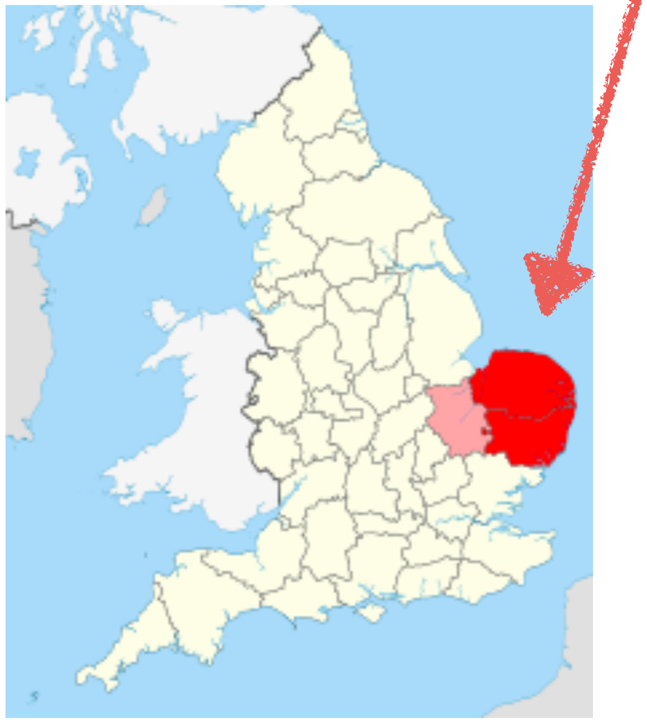
Mean absolute difference [MAD] and 95% probability interval length [PIL] for T (scaled by true T)

Bar 1 = single locus, n=10; Bar 2 = single locus, n=40

Bar 3 = 5 loci, n=10; Bar 4 = 5 loci, n=40.

Actual application: Y-chromosome Adam

- Data of Cooper *et al.* (1996)
- 212 individuals (E. Anglia, Sardinia, Nigeria)
- 5 micro-satellite loci
- Two analyses:
  1. Roughly 20 indivs from each population [NSE]
  2. All 174 E. Anglians (only) [EA]



Nigeria
Atlantic Ocean
Copyright © Ontheworldmap.com

Prior distributions

- $\theta=2N\mu$, where μ is the actual mutation rate and N is the (effective) population size. (confounding)
- Heyer *et al.* (1997) observed 3 mutations in 1491 meioses --> μ~0.02. Used a gamma prior with mode 3/1492 and mean 4/1492
- Two priors for N, both centered at ~5000:

  “low -variance” - Gamma (concentrated on [1000,10000])

  “high variance” - Lognormal (some support for values >20000)
- Coalescent prior for *T*.

MCMC details

- 40 branch swaps between each attempt to update *N* and *μ*
- **Sample once every 4100 iterations to avoid correlation(!)**
- Discard first 2000 such samples (‘burn-in’), retain next 10000 such samples
- Do two analyses for each scenario (starting from a different, randomly generated tree) [Stationarity test]

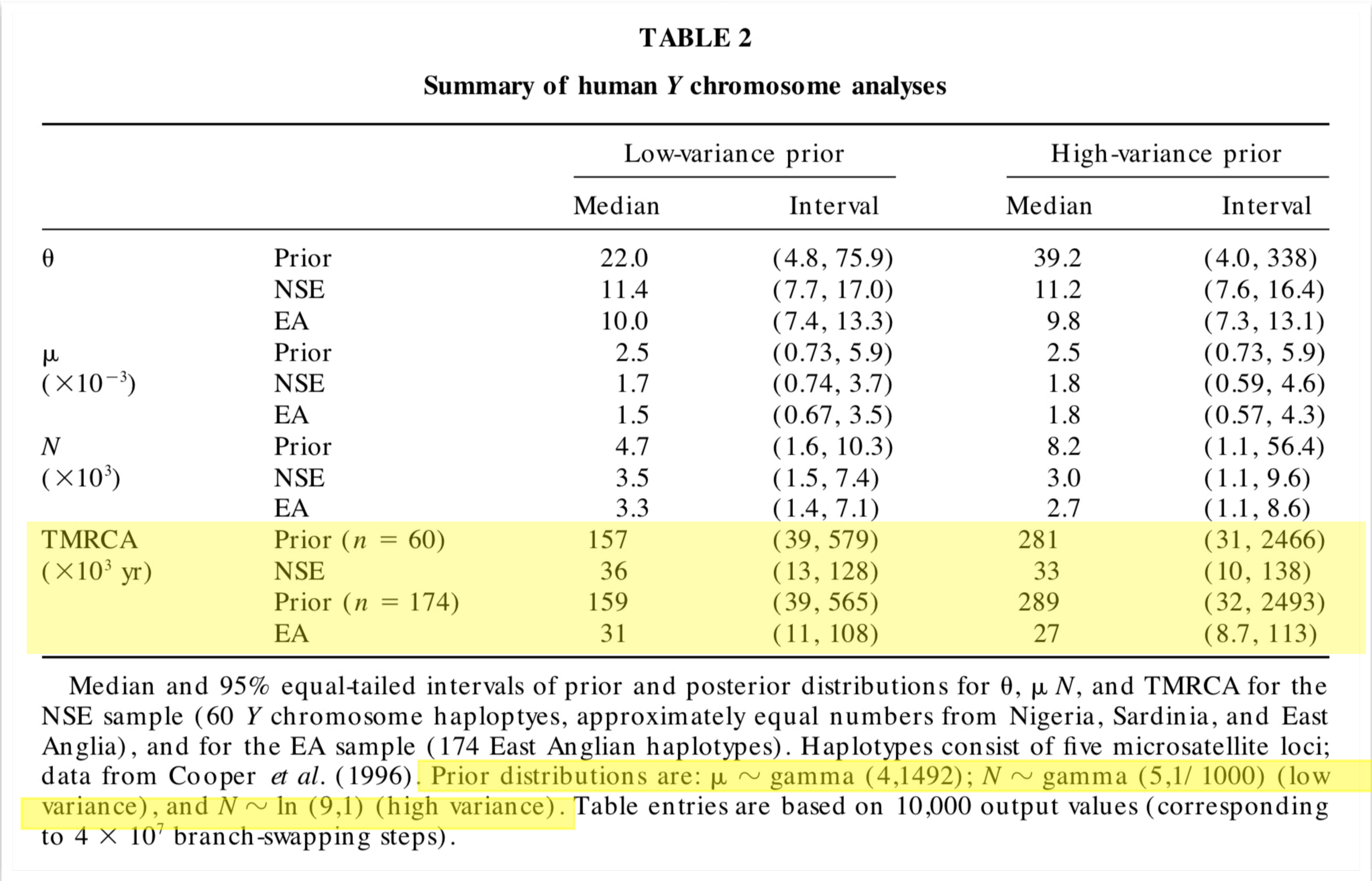


TABLE 2

Summary of human Y chromosome analyses

		Low-variance prior		High-variance prior	
		Median	Interval	Median	Interval
θ	Prior	22.0	(4.8, 75.9)	39.2	(4.0, 338)
	NSE	11.4	(7.7, 17.0)	11.2	(7.6, 16.4)
	EA	10.0	(7.4, 13.3)	9.8	(7.3, 13.1)
μ	Prior	2.5	(0.73, 5.9)	2.5	(0.73, 5.9)
$(\times 10^{-3})$	NSE	1.7	(0.74, 3.7)	1.8	(0.59, 4.6)
	EA	1.5	(0.67, 3.5)	1.8	(0.57, 4.3)
N	Prior	4.7	(1.6, 10.3)	8.2	(1.1, 56.4)
$(\times 10^{3})$	NSE	3.5	(1.5, 7.4)	3.0	(1.1, 9.6)
	EA	3.3	(1.4, 7.1)	2.7	(1.1, 8.6)
TMRCA	Prior (n = 60)	157	(39, 579)	281	(31, 2466)
$(\times 10^{3}$ yr)	NSE	36	(13, 128)	33	(10, 138)
	Prior (n = 174)	159	(39, 565)	289	(32, 2493)
	EA	31	(11, 108)	27	(8.7, 113)

Median and 95% equal-tailed intervals of prior and posterior distributions for θ, μ N, and TMRCA for the NSE sample (60 Y chromosome haploptyes, approximately equal numbers from Nigeria, Sardinia, and East Anglia), and for the EA sample (174 East Anglian haplotypes). Haplotypes consist of five microsatellite loci; data from Cooper *et al.* (1996). Prior distributions are: $\mu \sim$ gamma (4,1492); $N \sim$ gamma (5,1/ 1000) (low variance), and $N \sim$ ln (9,1) (high variance). Table entries are based on 10,000 output values (corresponding to 4×10^{7} branch-swapping steps).

Robustness to choice of prior

In summary

- They built a full probability model to allow exact calculation of P(D|θ) (subject to correctness of the model itself).
- Augmented state-space to make calculation possible. *This is often a useful tool.*
- Multiple (tightly linked) loci are required for useful inference.
- Inference for the two datasets was quite similar despite geographical differences.
- TMRCA estimates substantially lower than earlier studies from mtDNA, but broadly consistent with Tavaré *et al.* (1997 - Y chromosome) - stochastic variation?

Current beliefs (wikipedia)

- Mitochondrial Eve: 150,000 years ago (110-230 kya).
- Y-chromosome Adam: 200,000 to 300,000 years ago, roughly consistent with the emergence of anatomically modern humans.

Gelman stationarity test

- Run two (or more) replicate analyses and compare the answers.
- Compare the variance of the parameter distribution in the two (or more) chains.
- If both chains have reached stationarity, the variance within each replicate analysis will be the same as that across (i.e. after combining) the two replicates.
- So the Gelman test compares "across run" variance to "within run" variance (dividing the former by the latter). Stationarity has been reach if the statistic is close to 1 ("less than 1.1" is a common "rule of thumb"). This ratio is often called. "r-hat"
- R-hat is a diagnostic of convergence, not a proof of convergence.

Examinable assignment 3 - part 1

- Use the IndeptGamma.R code to use MCMC to produce samples from a Gamma(2.3,2.7) random variable
- Show how the performance deteriorates when you run the algorithm to produce samples from a Gamma(0.1,0.01).
  - Use the Gelman plot to illustrate the deterioration in performance
- Find a proposal kernel (i.e. q(x->x'), the way of producing new candidate values) that performs more efficiently.
  - Again use the Gelman plots to show how performance has changed.
  - Discuss how the performance has improved

- Due on April 5th, at 1pm

Code on GitHub in Assignment3 as 'IndeptGamma.R'

Examinable Assignment 3 part 2: Code-breaking (due April 5th, 1pm)

- Gzo uclfg gcpo C qhcs okof te Gollk Qoeetb zo rhf slvem ce h LtqqfLtkio Fcqxol Rlhcgz tvgfcso gzo gollhio tu Gzo Sheiolf. Gzo ahlmced qtg hggoesheg zhs yltvdzg gzo ihl tvg hes zo rhf fgcqq ztqsced gzo sttl taoe yoihvfo Gollk Qoeetbf qoug uttg rhf fgcqq shedqced tvgfcso, hf cu zo zhs utldtggoe zo zhs teo. Zo zhs h ktvedqttmced uhio yvg zcf zhcl rhf yteo rzcgo. Ktv itvqs goqq yk zcf okof gzhg zo rhf aqhfgolos gt gzo zhclqceo, yvg tgzolrcfo zo qttmos qcmo hek tgzol ecio ktveddvk ce h sceeol jhimog rzt zhs yooe faoesced gtt pviz pteok ce h jtceg gzhg obcfgf utl gzhg avlatfo hes utl et tgzol. Gzolo rhf h dclq yofcso zcp. Zol zhcl rhf h qtxoqk fzhso tu shlm los hes fzo zhs h scfgheg fpcqo te zol qcaf hes txol zol fztvqsolf fzo zhs h yqvo pcem gzhg hqptfg phso gzo LtqqfLtkio qttm qcmo jvfg hetgzol hvgtptycqo. Cg scseg wvcgo. Etgzced ihe. Gzo hggoesheg rhf gzo vfvhq zhqugtvdz izhlhigol ce h rzcgo ithg rcgz gzo ehpo tu gzo lofghvlheg fgcgizos hiltff gzo ulteg tu cg ce los. Zo rhf doggced uos va. "Qttm, pcfgol," zo fhcs rcgz he osdo gt zcf xtcio, "rtvqs ktv pces h rztqo qtg avqqced ktvl qod cegt gzo ihl ft C ihe mces tu fzvg gzo sttl Tl fztvqs C taoe cg hqq gzo rhk ft ktv ihe uhqq tvg" Gzo dclq dhxo zcp h qttm rzciz tvdzg gt zhxo fgvim hg qohfg utvl ceizof tvg tu zcf yhim. Cg scseg ytgzol zcp oetvdz gt dcxo zcp gzo fzhmof. Hg Gzo Sheiolf gzok dog gzo ftlg tu aotaqo gzhg scfcqqvfcte ktv hytvg rzhg h qtg tu dtquced pteok ihe st utl gzo aolftehqcgk.

[CodedMessage_Short.txt in Assignment3 repo on Github]

Assignment 3 - Part 2

- Use one of the methods you have met in the course to try to break the code.
- I may not be easy to break the code completely without resorting to manual ‘tweaks’ at the end, but you should be able to get to the point at which you can work out what the text is saying.
- Write it up as an Rmarkdown file and include a description of the methods you are using (for both parts of the assignment) and a discussion of your results. Upload it to your version of the Assignment 3 repo on Github please.
- Due April 5th, 1pm.

Suggestion

- Need something to maximize, so let's maximize the likelihood.
- Suppose we have a sequence of letters $l_1\ l_2\ l_3\ l_4\ l_5$. Then [Recall $P(A|B)=P(A\cap B)/P(B)$]:
  - $P(l_1\ l_2\ l_3\ l_4\ l_5)=P(l_1)\ P(l_2|l_1)\ P(l_3|l_2\ l_1)\ P(l_4|l_3\ l_2\ l_1)\ P(l_5|l_4\ l_3\ l_2\ l_1)$
- Suppose the sequence of letters is Markovian. Then:
  - $P(l_1\ l_2\ l_3\ l_4\ l_5)=P(l_1)\ P(l_2|l_1)\ P(l_3|l_2)\ P(l_4|l_3)\ P(l_5|l_4)$
  - $P(l_1\ l_2\ l_3\ l_4\ l_5)=P(l_1)\ [P(l_1l_2)/P(l_1)]\ [P(l_2l_3)/P(l_2)]\ [P(l_3l_4)/P(l_3)]\ [P(l_4l_5)/P(l_4)]$
- Let $f_{\alpha\beta}$ denote the frequency with which the letter pair $\alpha\beta$ is observed in text (from the file on Blackboard). Let f_α denote the frequency with which the letter α is observed in text. Then $f_\alpha = \sum_\beta f_{\alpha\beta}$.
- Use $f_{\alpha\beta}$ as an estimate of $P(\alpha\beta)$ and f_α as an estimate of $P(\alpha)$.
- Now treat it as a maximization (of likelihood) or MCMC problem.

Another (Simpler!) MH-MCMC Example - Bivariate normals

- Assume we have some data from a bivariate normal distribution, for which we wish to estimate the means.
- For convenience, we will assume the variance/covariance structure is known.

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N\left[\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \sigma_1^2 & \rho\,\sigma_1\sigma_2 \\ \rho\sigma_2\sigma_1 & \sigma_2^2 \end{pmatrix}\right]$$

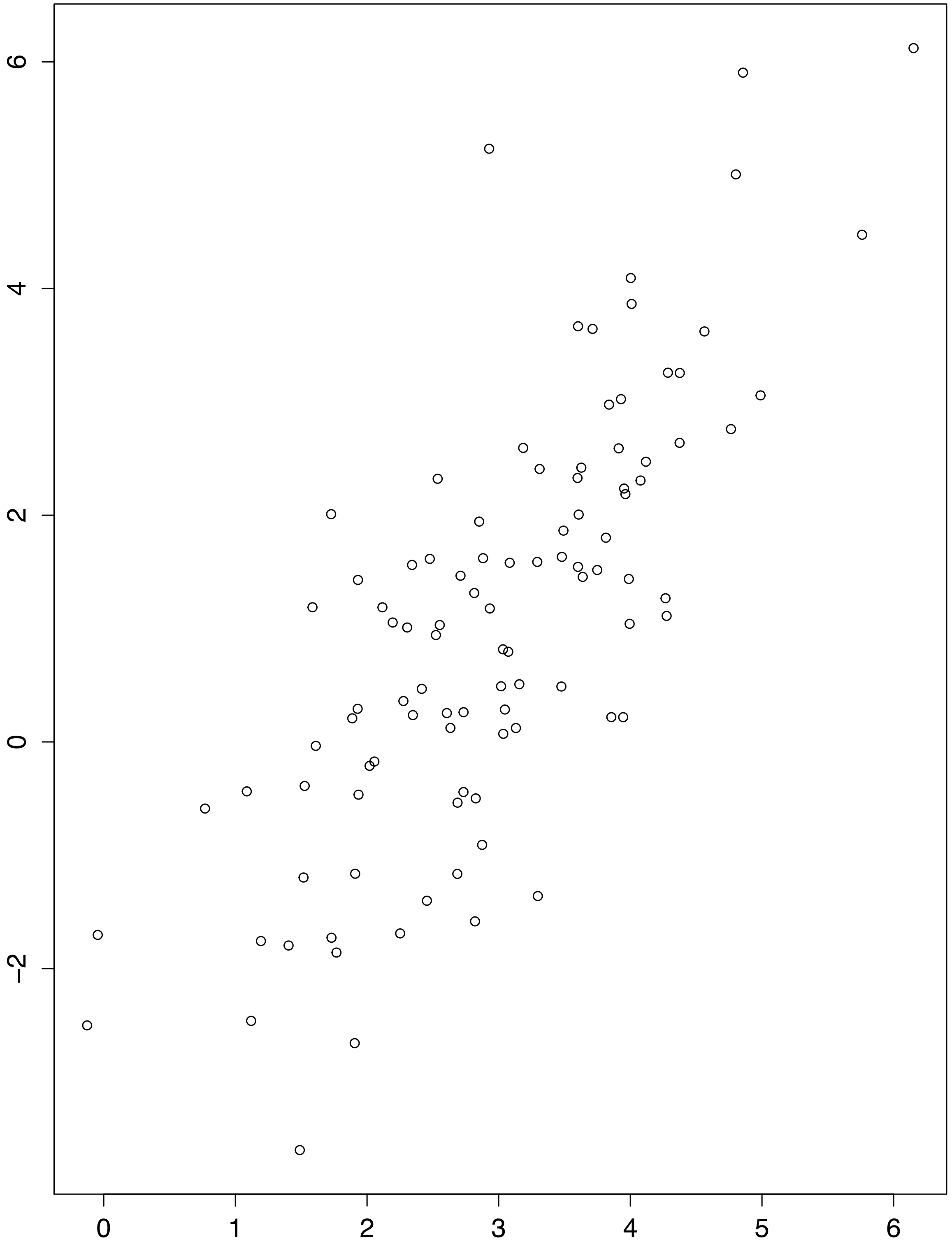
Properties:

$$X_1 \sim N(\mu_1, \sigma_1^2)$$

$$X_2 \sim N(\mu_2, \sigma_2^2)$$

$$Correlation(X_1, X_2) = \rho. \text{ [Recall } \rho_{X_1X_2} = \frac{Cov(X_1X_2)}{\sigma_X\sigma_Y}\text{]}$$

The data (correlation=0.75)



MVNormals.Rmd on Github

```
# Generate some test data
mu.vector <- c(3, 1)    # the vector of means for the multi-variate normal
variance.matrix <- cbind(c(1, 0), c(0, 4))   # the variance-covariance matrix for the multi-variate normal
# variance.matrix <- cbind(c(1, 1.5), c(1.5, 4))
# variance.matrix <- cbind(c(1, 1.98), c(1.98, 4))
# Now generate one hundred samples from that distribution:
our.data<-rmvnorm(n=100,mean=mu.vector,sigma=variance.matrix)

# how many iterations do we want in our MH-MCMC process?
total.iterations<-10000

do.MHMCMC<-function(number.of.iterations){
  # start our MH-MCMC process off from somewhere
  current.mu<-c(3,1)
  current.mu<-runif(2,0,4)

  # define a vector to store the output of the MH-MCMC process….
  posterior.mu<-mat.or.vec(number.of.iterations,2)
```

In-class exercise

- Try running the algorithm for each of the three test datasets:

```
mu.vector <- c(3, 1)    # the vector of means for the multi-variate normal
variance.matrix <- cbind(c(1, 0), c(0, 4))   # the variance-covariance matrix for the multi-variate normal
# variance.matrix <- cbind(c(1, 1.5), c(1.5, 4))
# variance.matrix <- cbind(c(1, 1.98), c(1.98, 4))
```

- Explore behavior (use Gelman diagnostics, acfs, posterior densities) for each case.
- Fix it, for cases in which it performs badly.

Lab time

MCMC in R

- “Metrop” library.
- See Week8—MCMCPackage repo on Github.
  - Works with the “log un-normalized posterior
    - Posterior is $f(\theta|D)= f(D|\theta)\pi(\theta)/f(D)$
    - Un-normalized posterior is $f(\theta|D) \propto f(D|\theta)\pi(\theta)$
    - log(un-norm^lzd^ posterior is $\ln(f(D|\theta)\pi(\theta))$

MCMC in R - “metrop” package

- MCMC1.Rmd on Github (in the MCMCpackage repo).
- Uses R’s built-in “metrop” package
- The example conducts a Bayesian analysis of the data ‘logit’ from R’s “mcmc” package.
- I have put the mcmc package doc file (“mcmc.pdf”) there as well

Summary of R MCMC library

- ‘metrop’ is a convenient way to implement simple MH-MCMC in R.
- Control the proposal step size (via scale), aiming to obtain an acceptance rate of around 20%
- You can thin the process out, using ‘nbatch' and ‘nspace’ to remove autocorrelation. (See Chapter 1 of “The Handbook of MCMC”, by Gelman et al., for a long discussion of whether you should sub-sample your data like this, or worry about ‘burn-in’.)

Exercises - non-examinable

1. Adapt the example from the MCMC1.Rmd document to run several MCMC chains for this problem and check convergence using the ‘coda’ package.

2. Use the metrop function to conduct an analysis of the dataset (QTLdata.txt) in that Repo
   - Genotype data for a set of 100 Single nucleotide polymorphisms, for 50 (inbred) individuals.
   - 1 row per individual; quantitative phenotype in last column.
   - The quantitative trait that depends upon one of those SNPs.
   - Find that SNP and estimate its ‘effect size’ (how much it changes the mean phenotype value).

3. **Use the metrop() function to implement the multivariate normal example from earlier today.**

END