Homework Review

What was different about setting up Normal model compared to the Binomial?

Did you run into any difficulties?

How did the output look? Was convergence good?

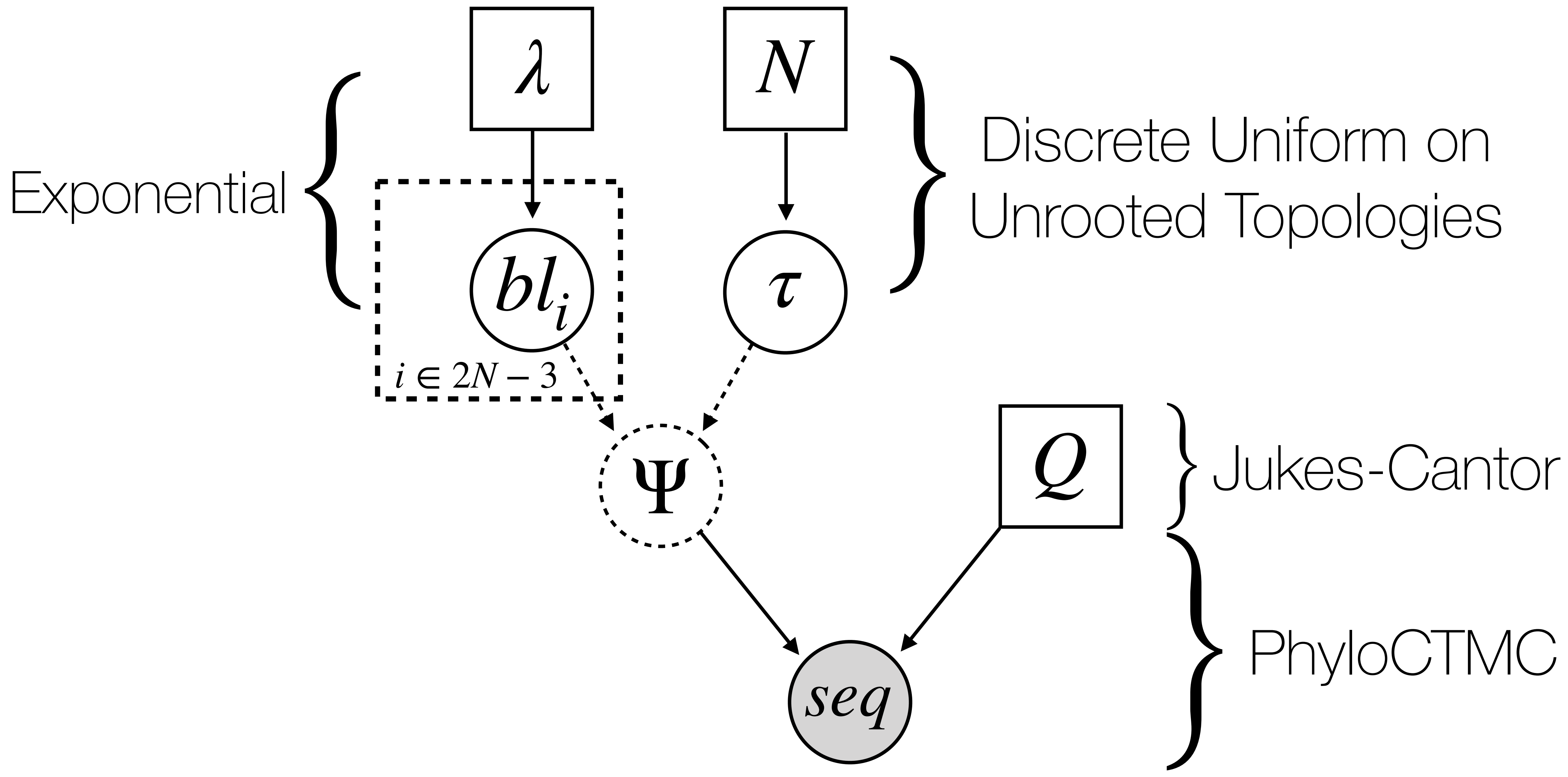
Did you run multiple replicates?

What effect, if any, did changing your prior have?

Phylogenetic Graphical Models

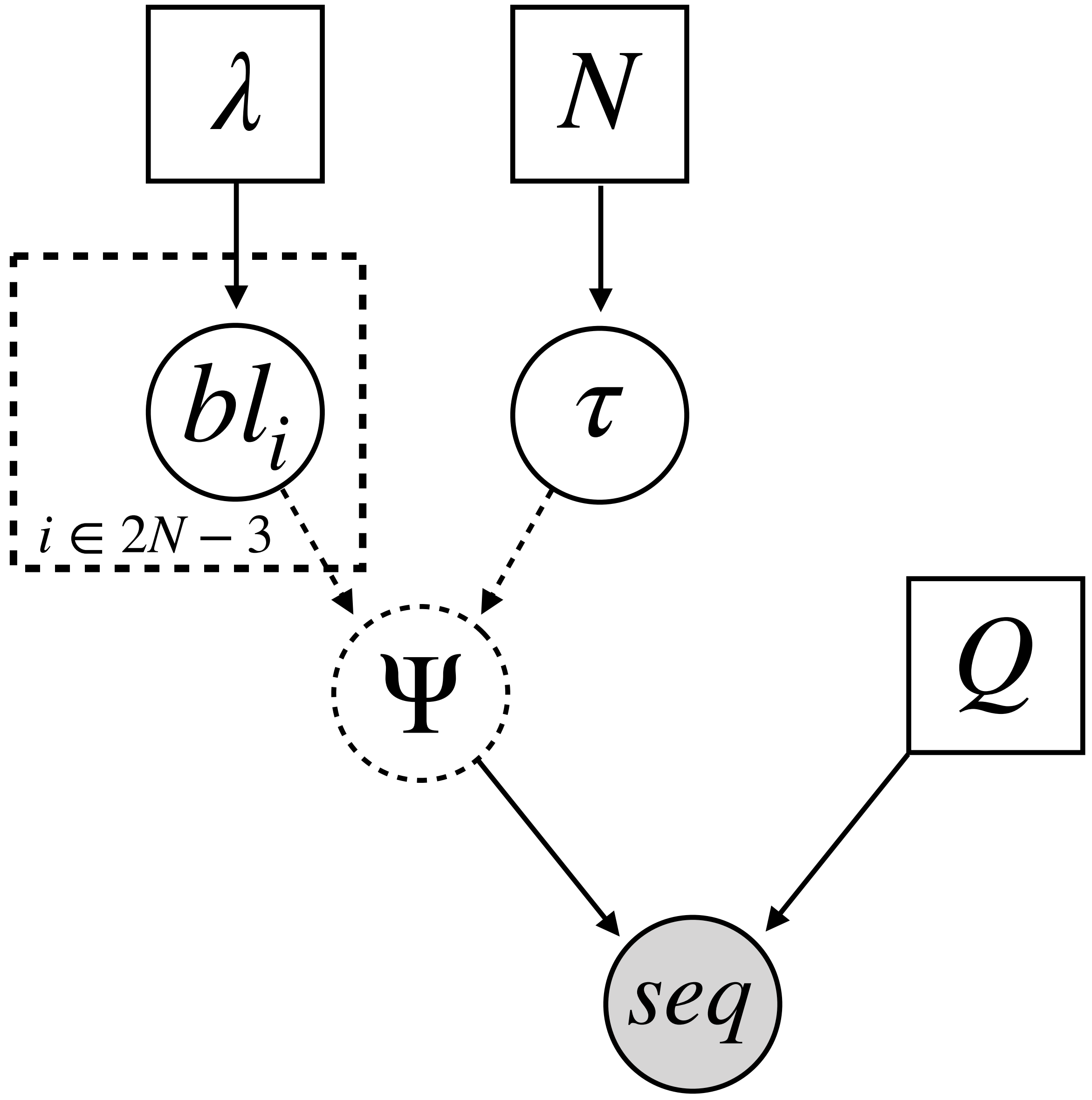
Jukes-Cantor

(Exponential branch-length prior)



Jukes-Cantor

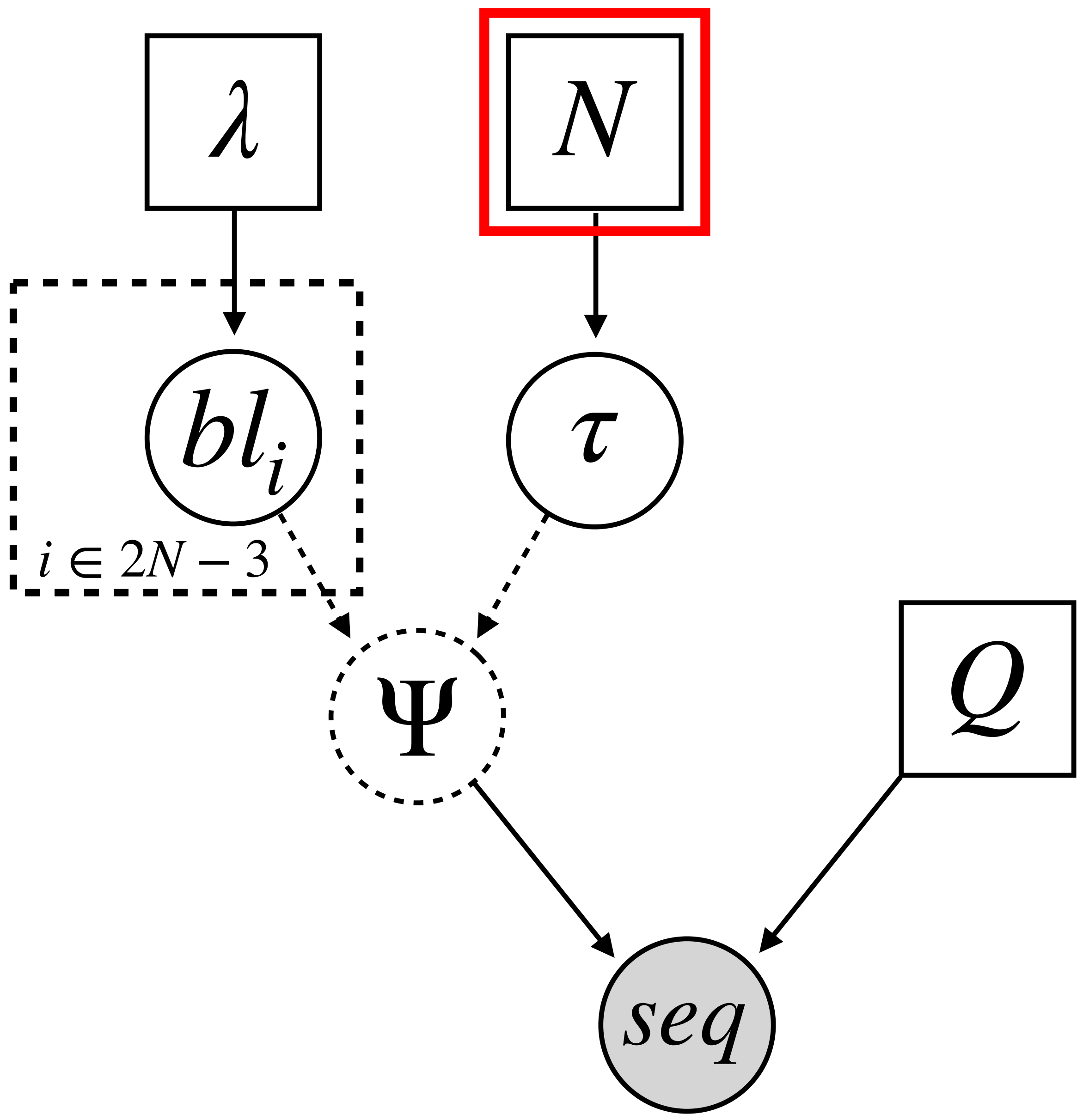
(Exponential branch-length prior)



```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Jukes-Cantor

(Exponential branch-length prior)

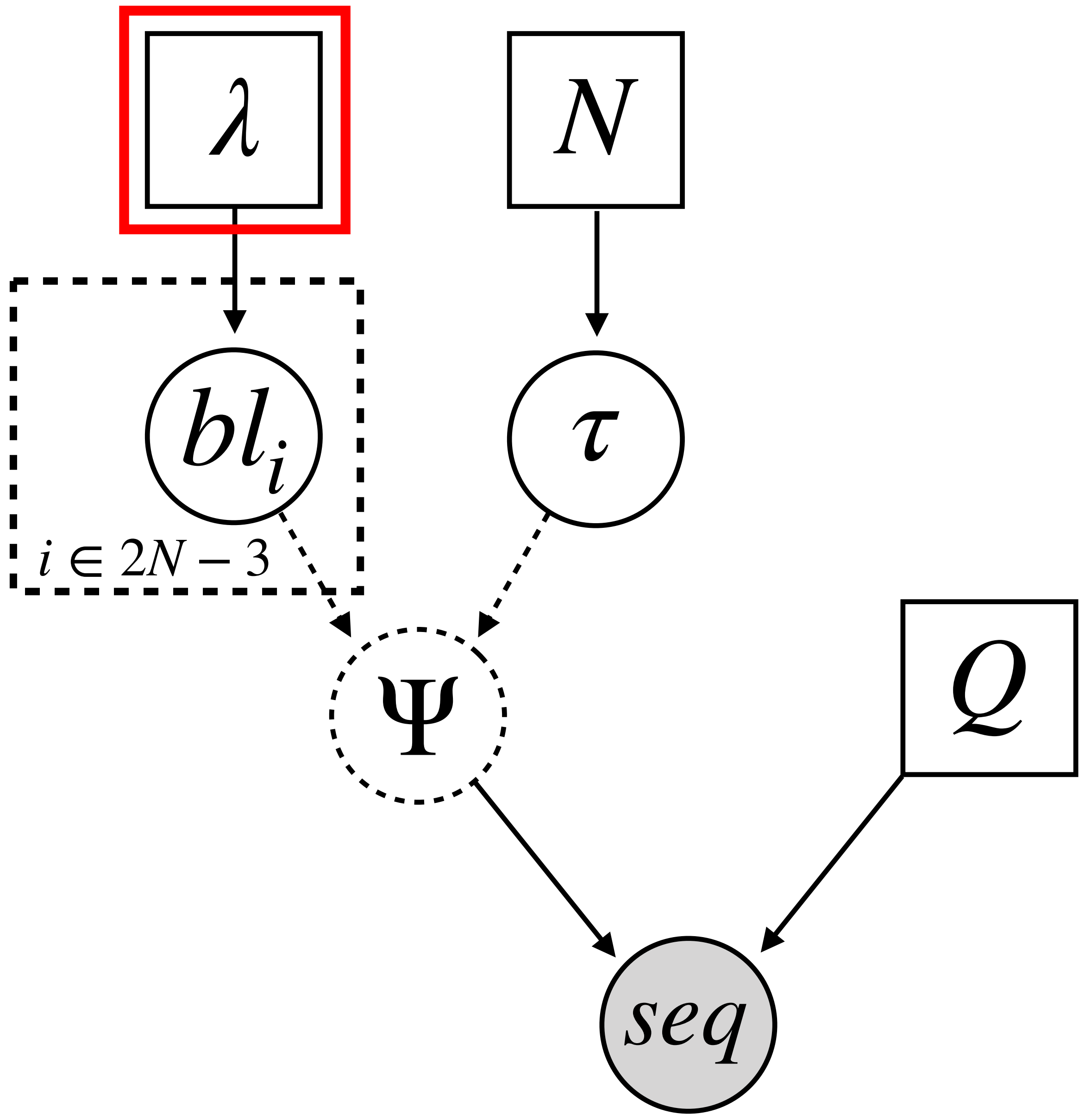


Reading in and summarizing dataset

```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Jukes-Cantor

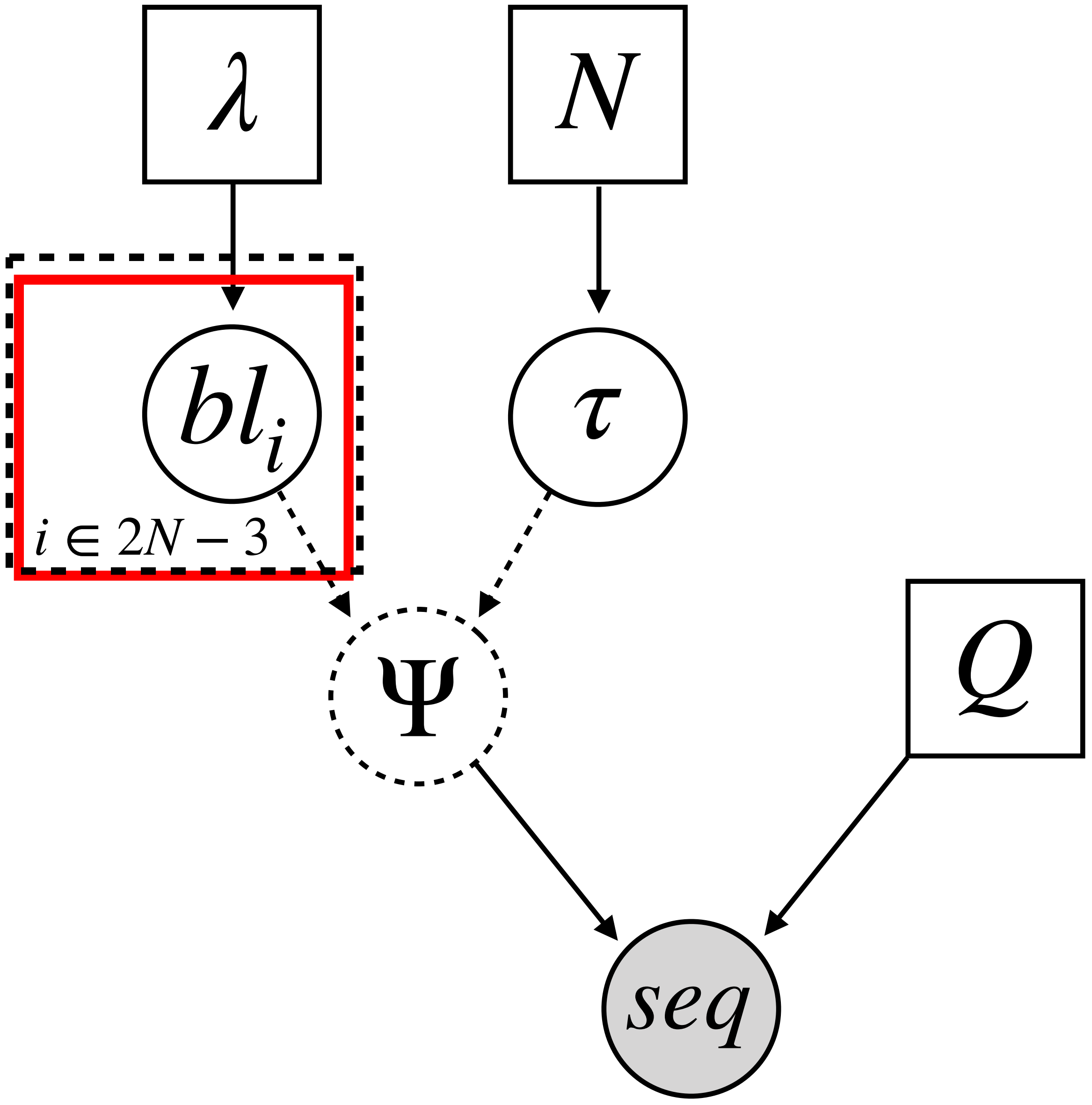
(Exponential branch-length prior)



```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Jukes-Cantor

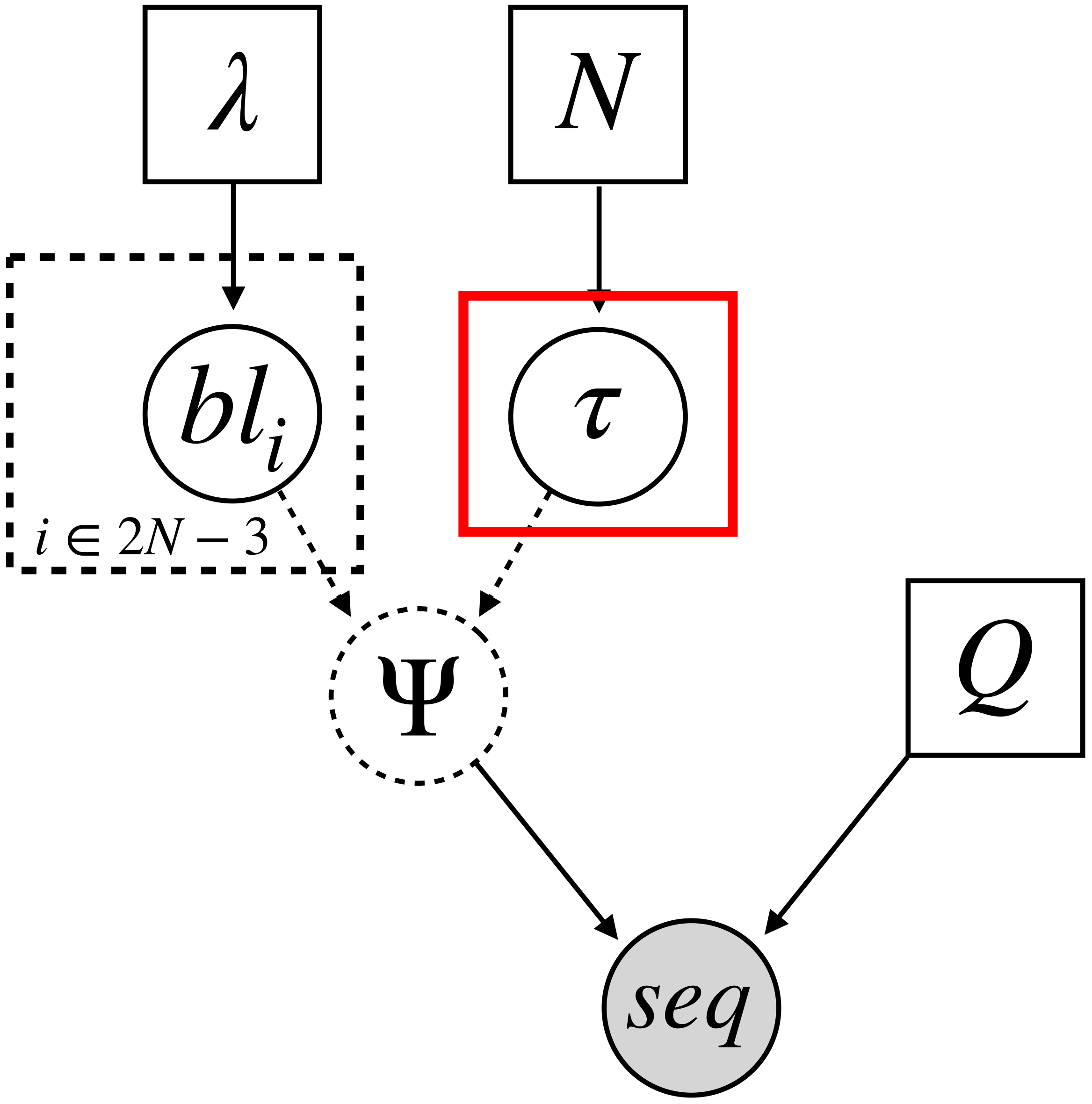
(Exponential branch-length prior)



```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Jukes-Cantor

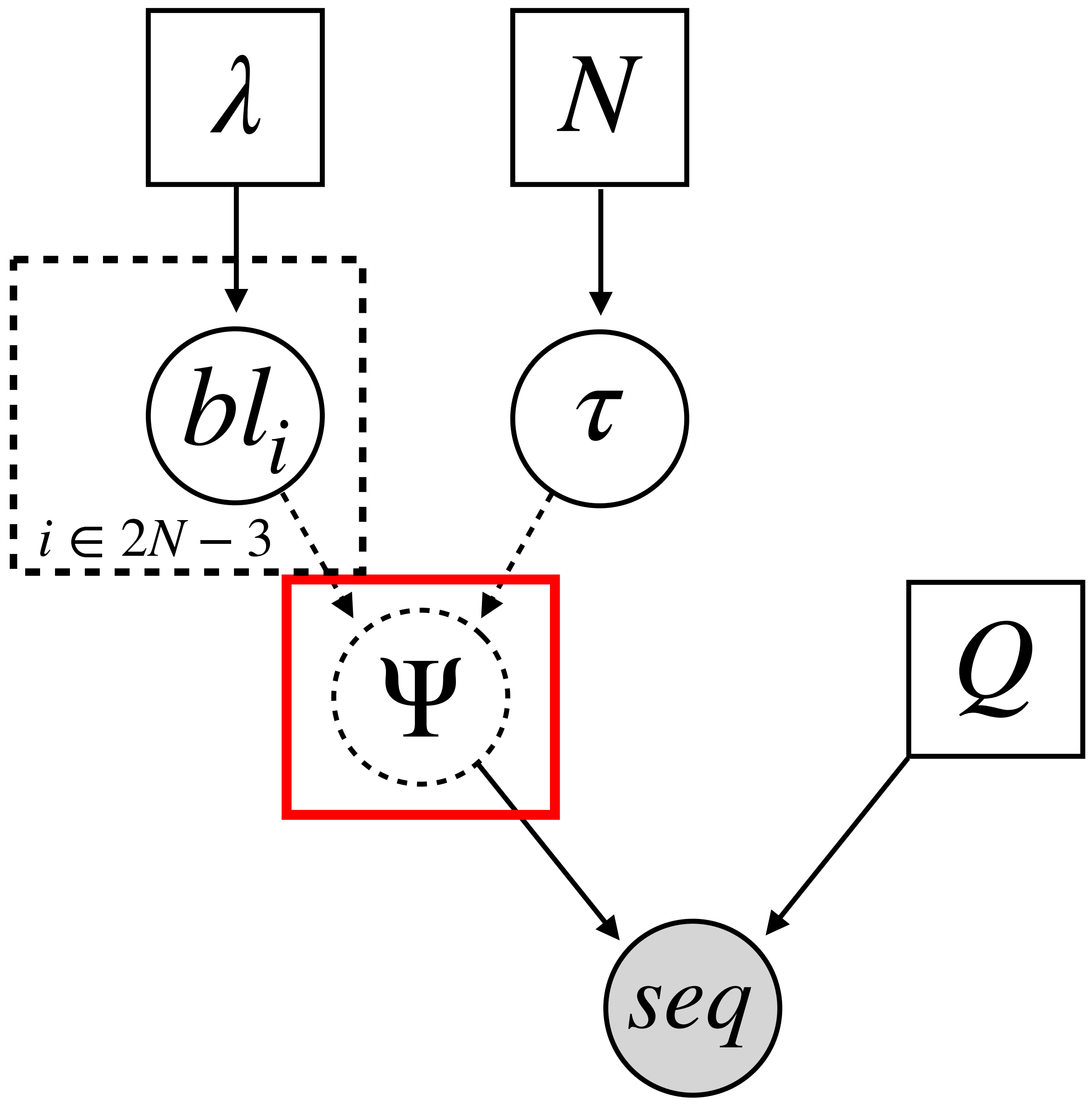
(Exponential branch-length prior)



```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Jukes-Cantor

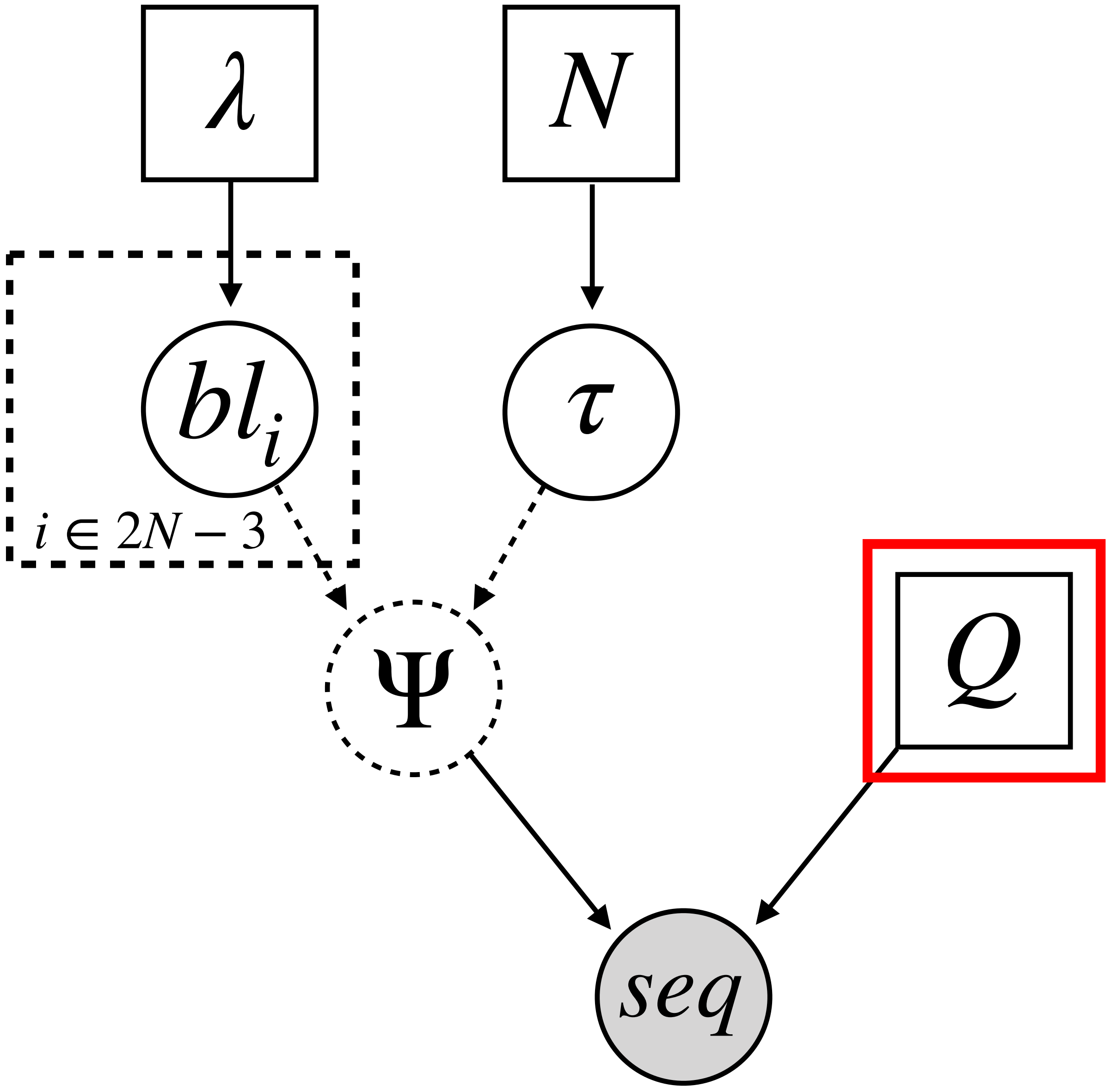
(Exponential branch-length prior)



```
data = readDiscreteCharacterData(“myData.nex”)
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type=“DNA”)

seq.clamp(data)
```

Jukes-Cantor

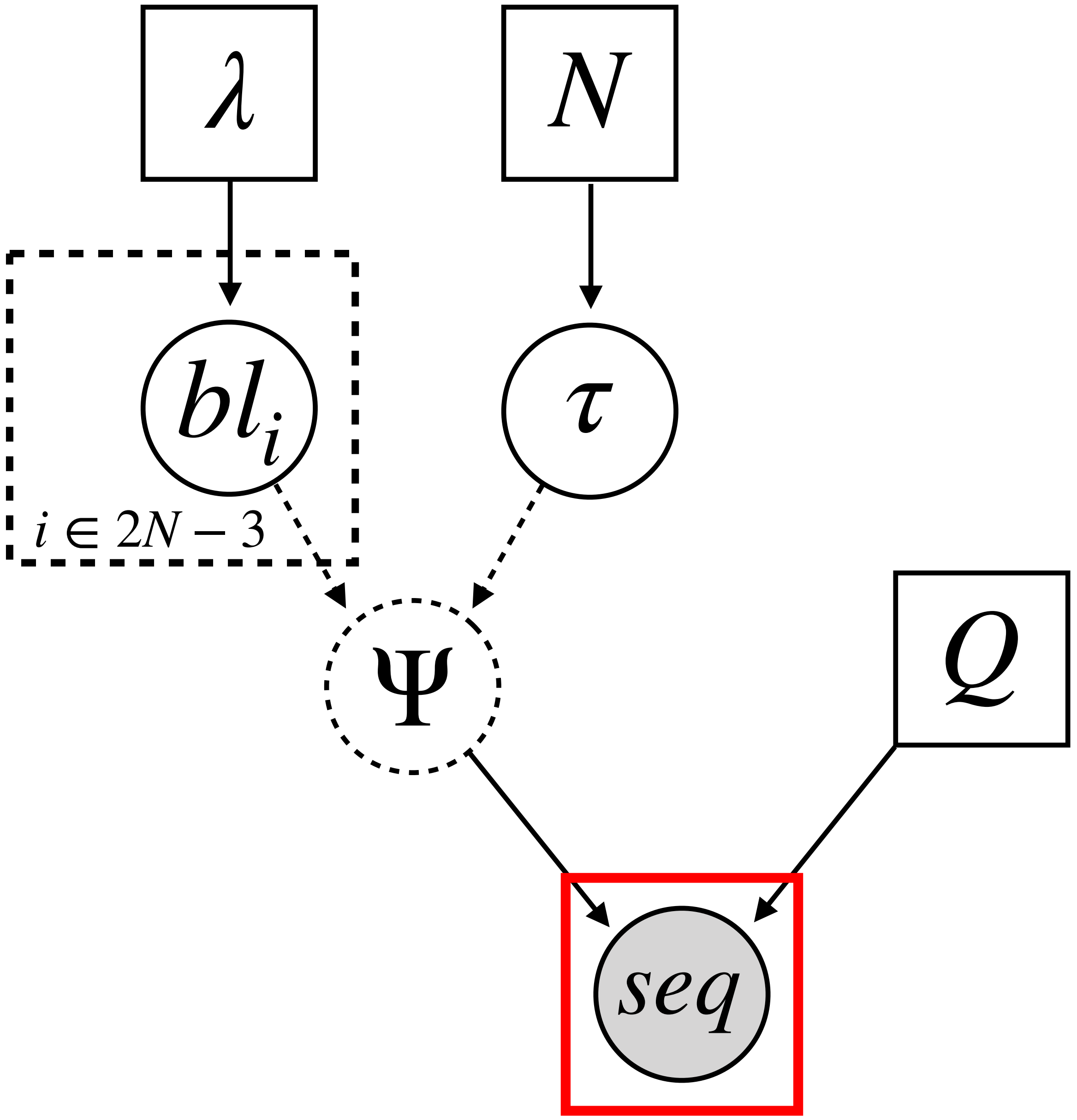
(Exponential branch-length prior)



```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Jukes-Cantor

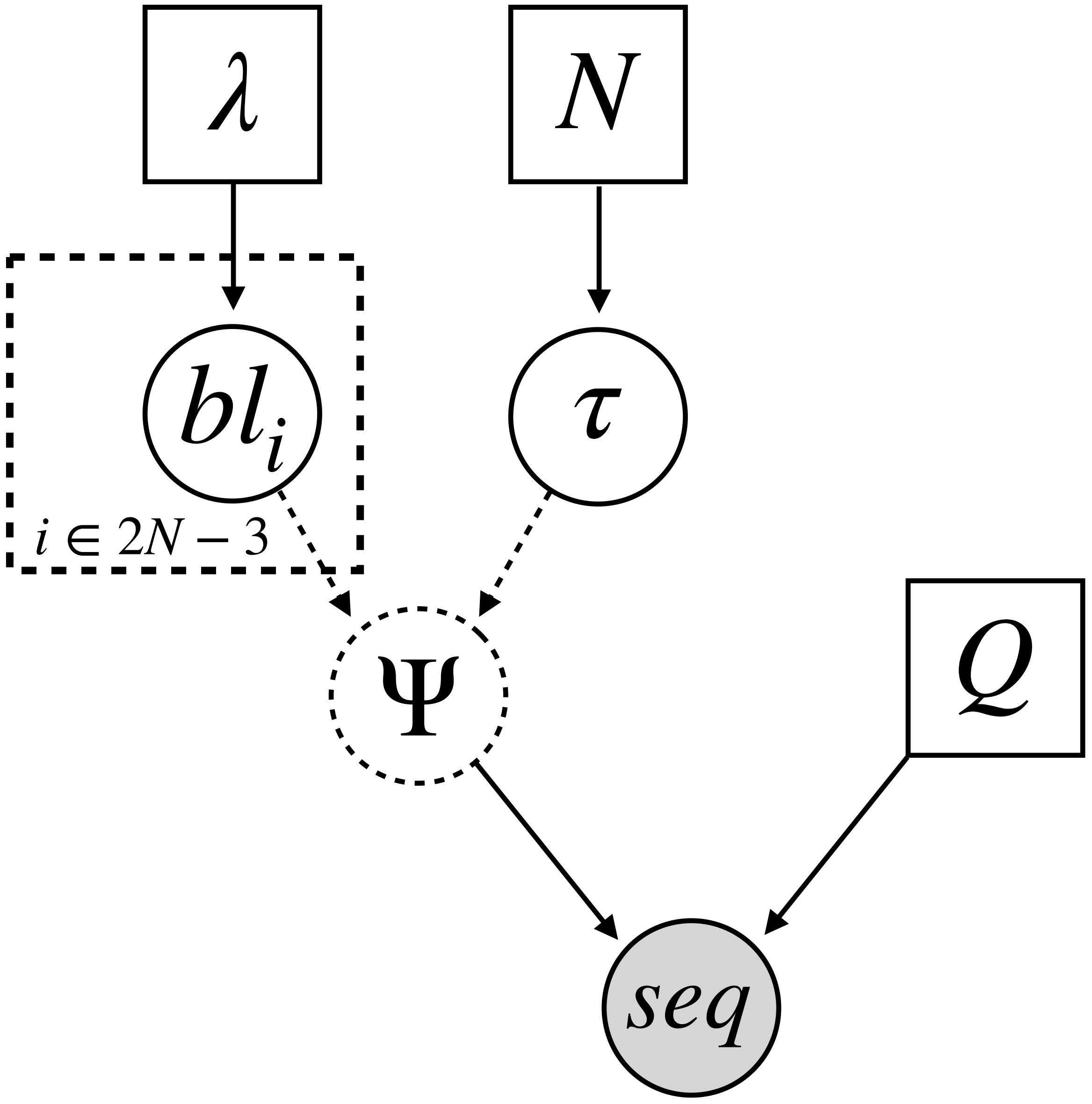
(Exponential branch-length prior)

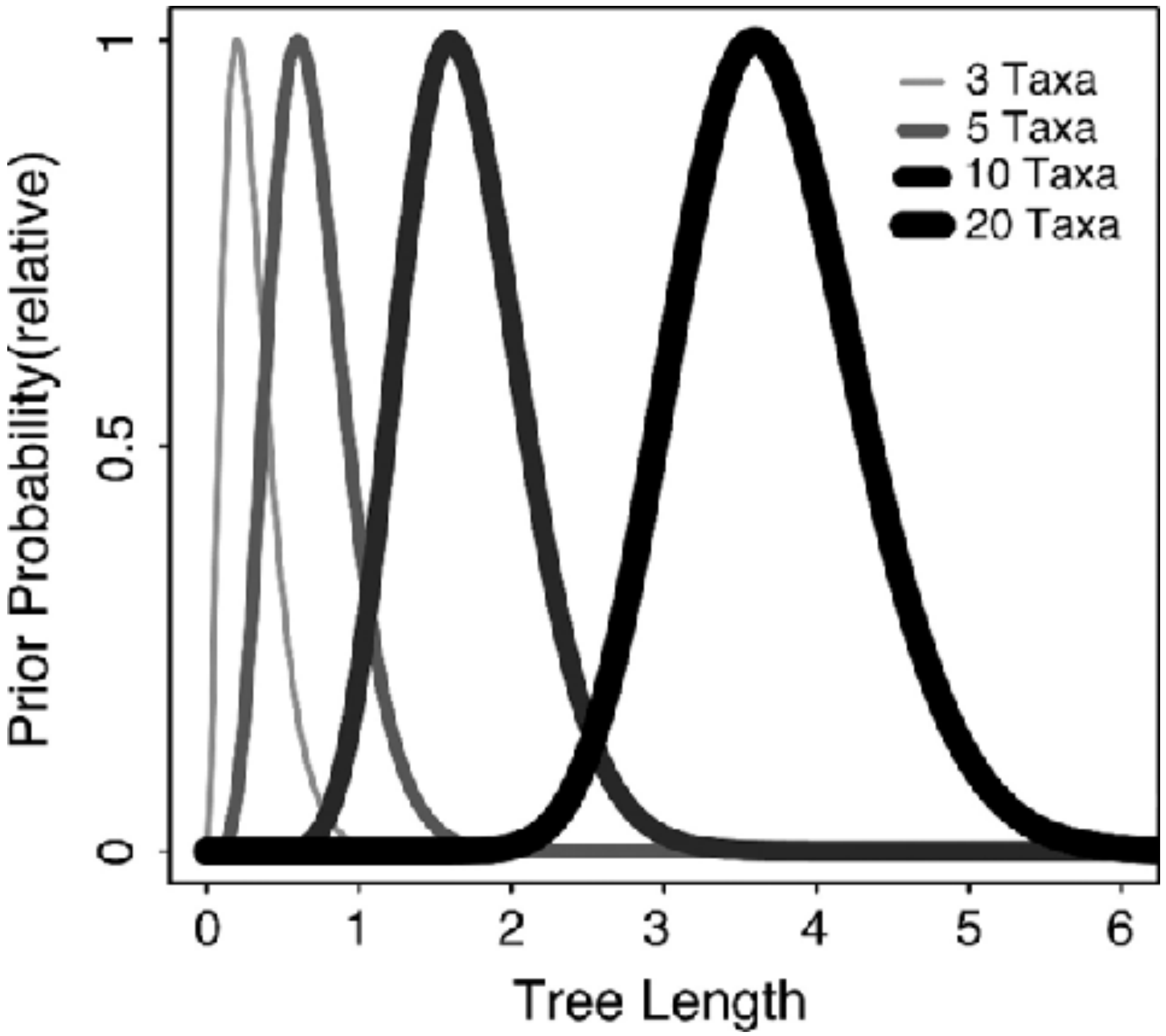


```
data = readDiscreteCharacterData(“myData.nex”)
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

lambda <- 10

for (i in 1:n_branches){
      bl[i] ~ dnExponential(lambda)
}

topology ~ dnUniformTopology(taxa)

psi := treeAssembly(topology, bl)

Q <- fnJC(4)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type=“DNA”)

seq.clamp(data)
```

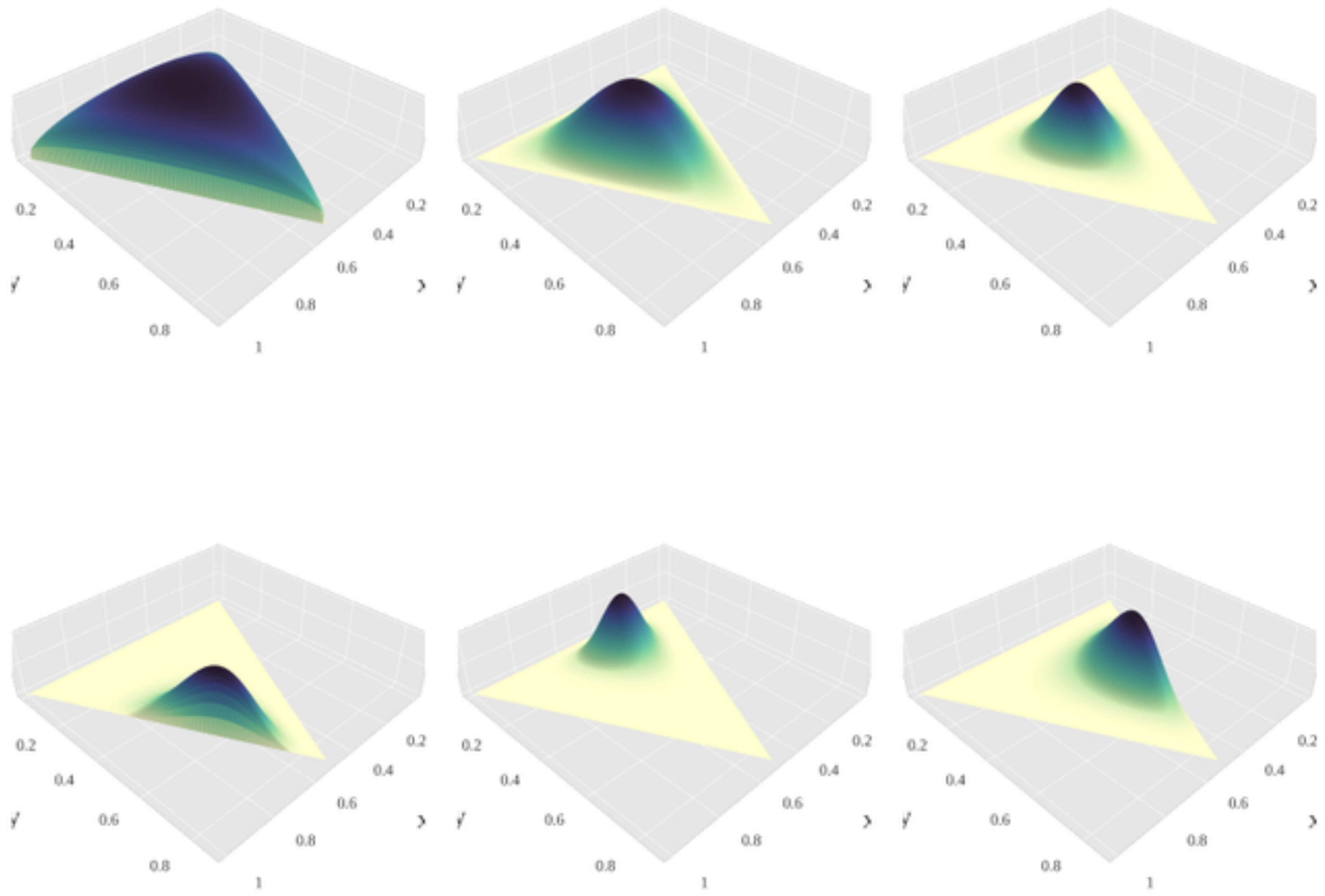
Jukes-Cantor

(Exponential branch-length prior)





Dirichlet Distribution

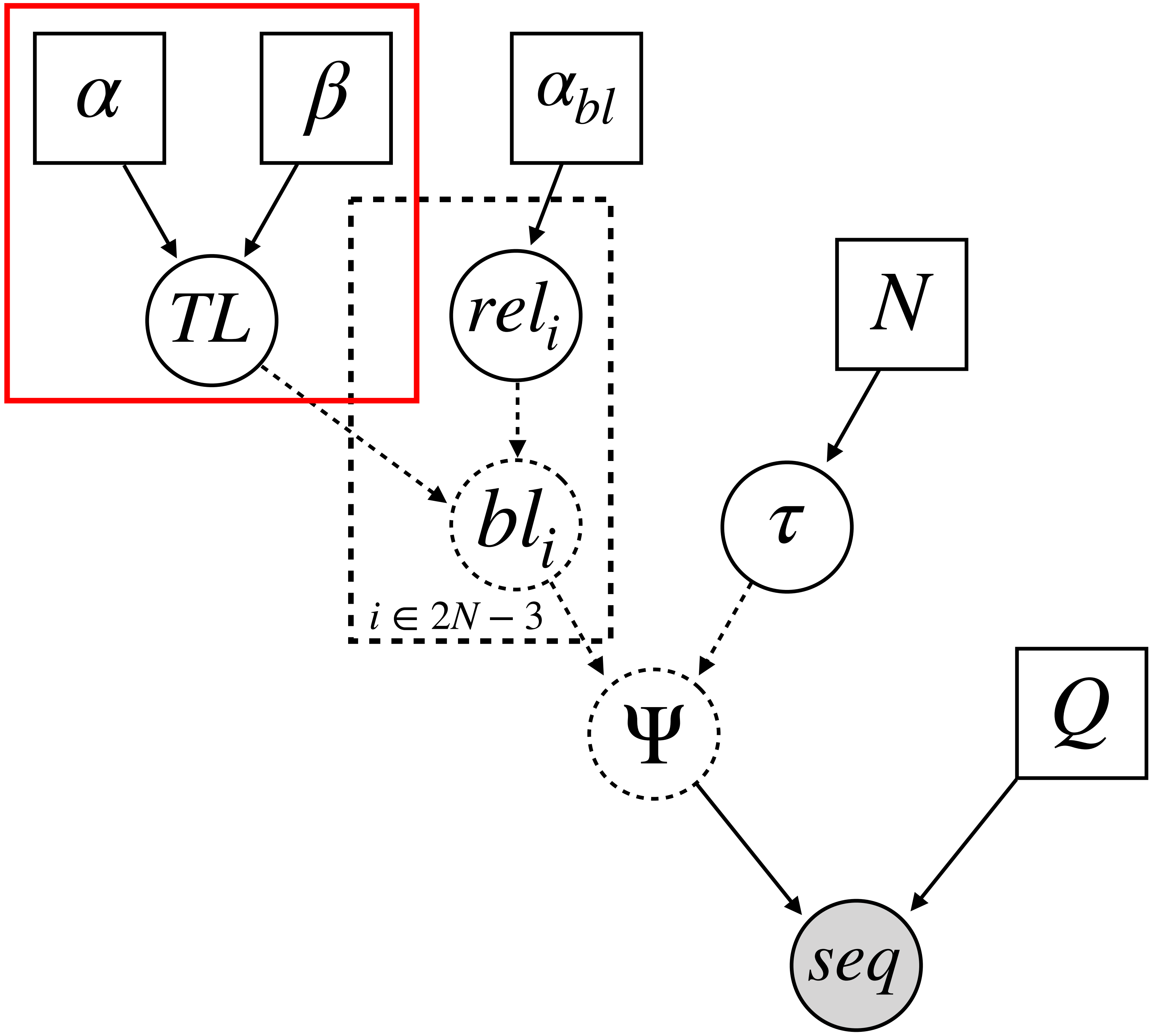


The Dirichlet distribution is a multivariate generalization of the Beta distribution. It describes the joint distribution of a set of parameters, each of which is bounded between 0 and 1, **and whose sum is 1**.

https://en.wikipedia.org/wiki/Dirichlet_distribution

Jukes-Cantor

(Compound Dirichlet tree- and branch-length prior)

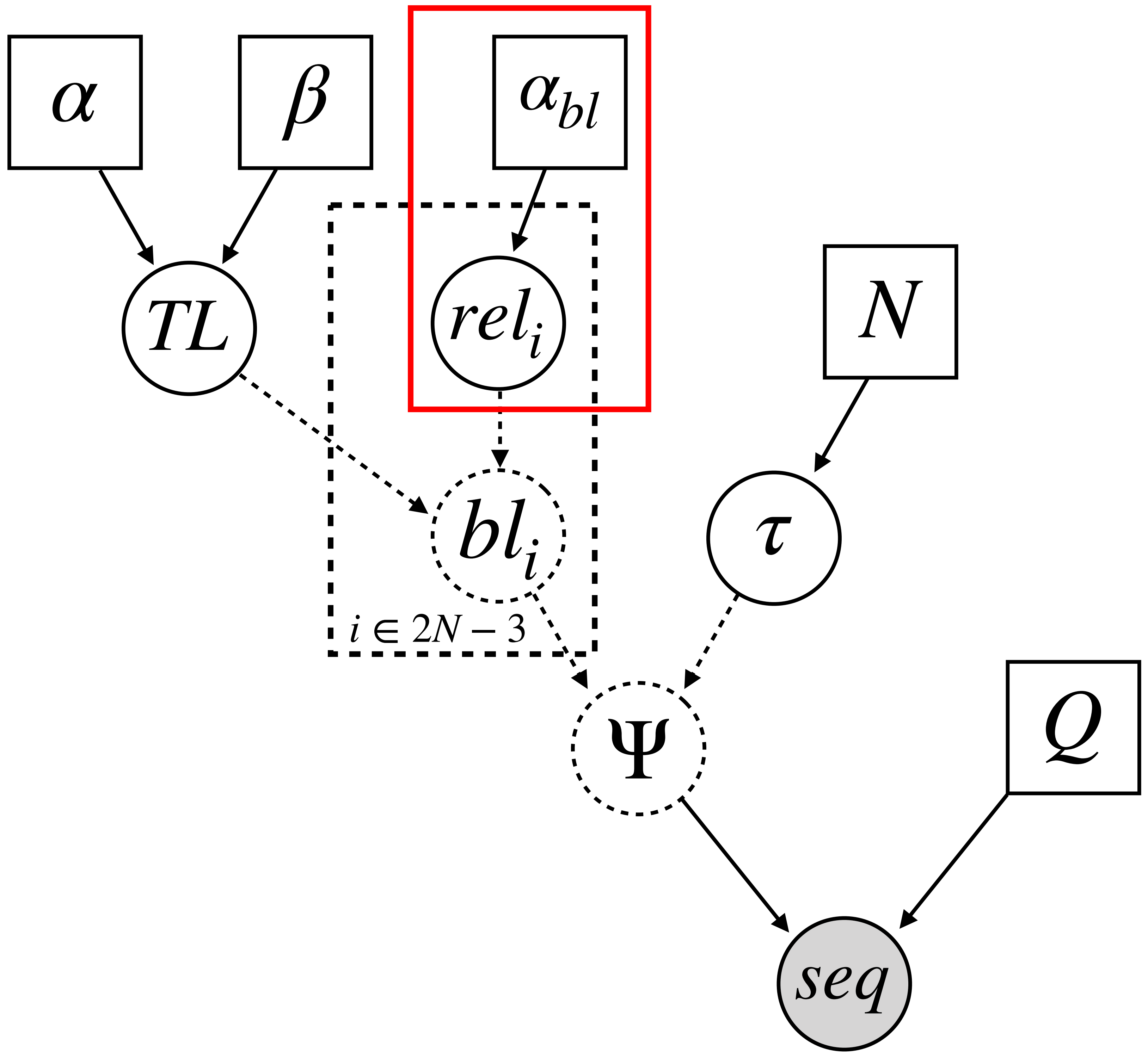


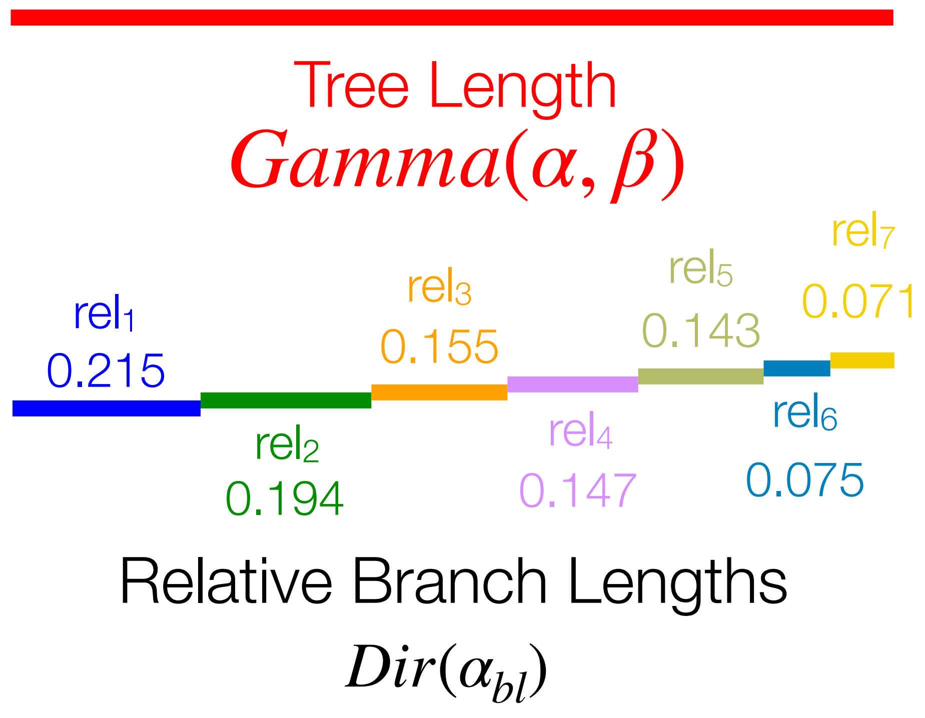
Tree Length

$Gamma(\alpha, \beta)$

Jukes-Cantor

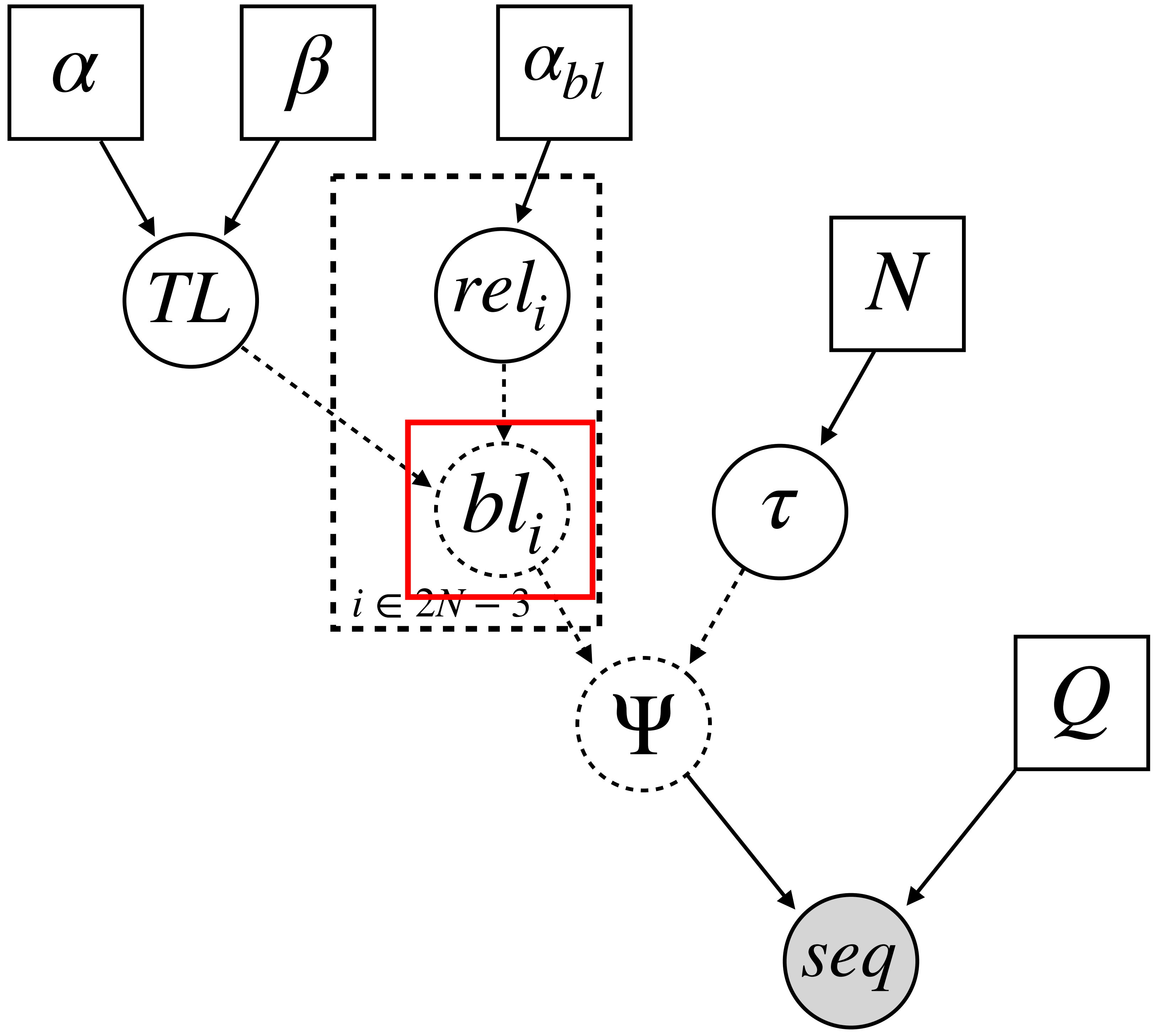
(Compound Dirichlet tree- and branch-length prior)

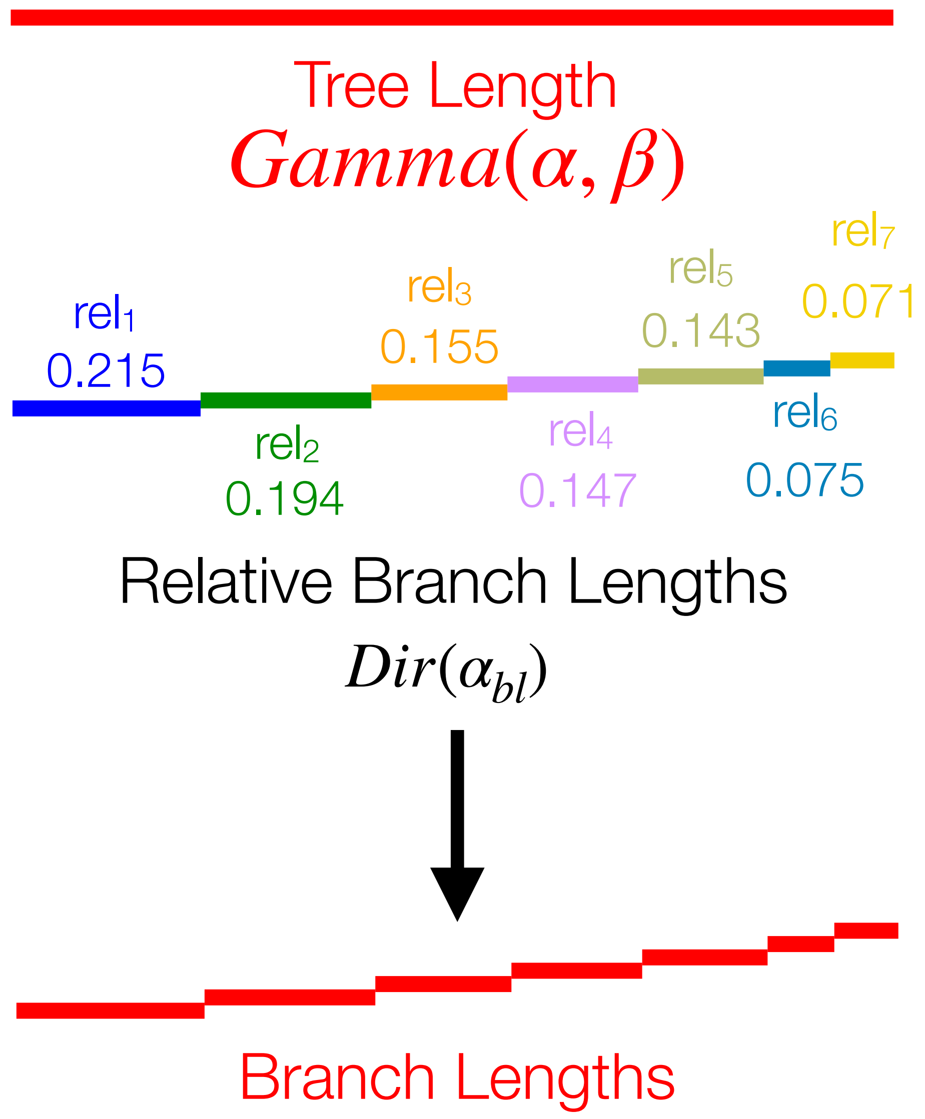




Jukes-Cantor

(Compound Dirichlet tree- and branch-length prior)





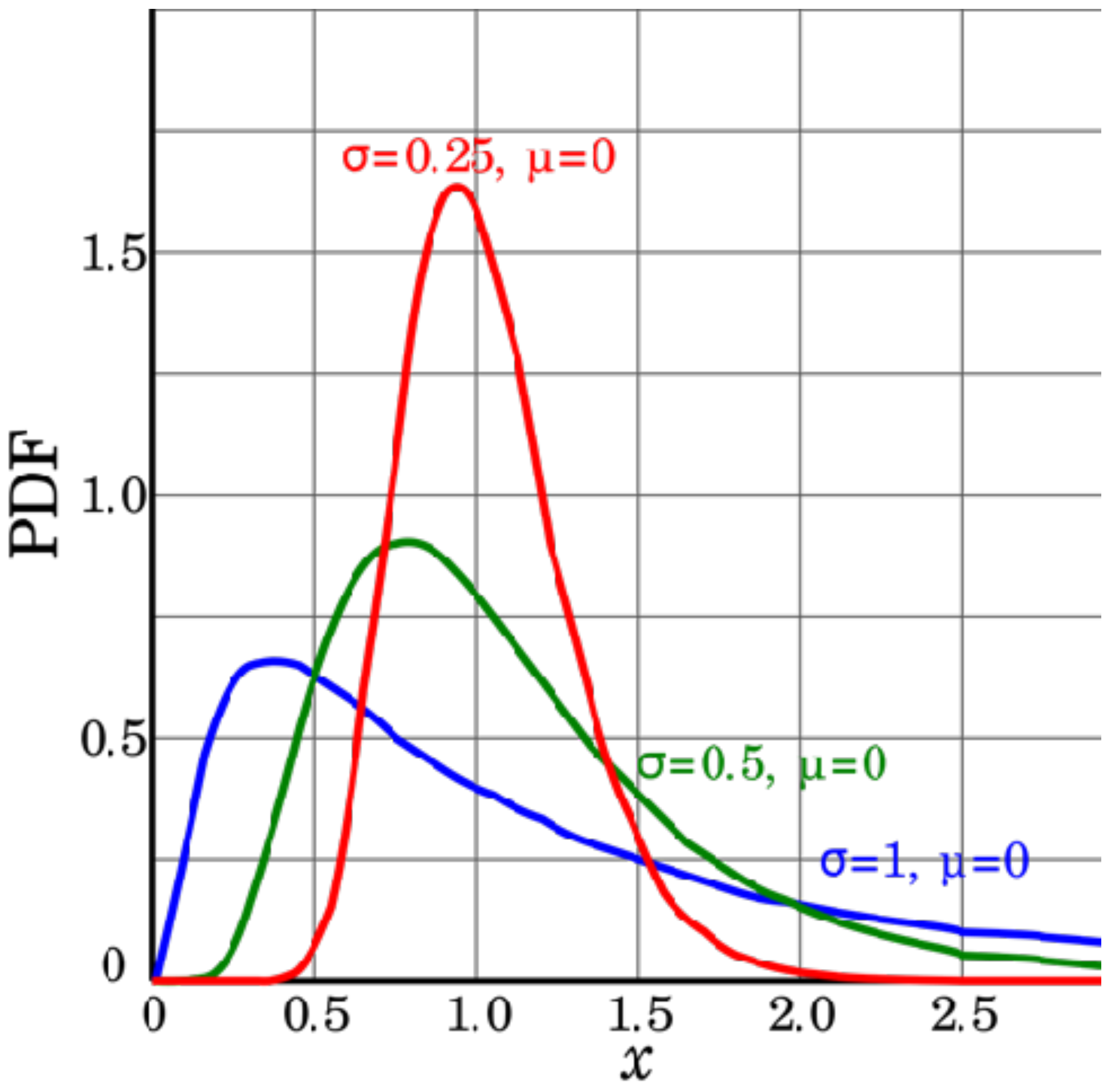
Jukes-Cantor

(Compound Dirichlet tree- and branch-length prior)

```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3

alpha <- 2
beta <- 4

TL ~ dnGamma(alpha,beta)

alpha_bl <- 1.0

rel_branch_lengths ~ dnDirichlet( rep(alpha_bl,n_branches) )

br_lens := rel_branch_lengths * TL

topology ~ dnUniformTopology(taxa)

Q <- fnJC(4)

psi := treeAssembly(topology, br_lens)

seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")

seq.clamp(data)
```

Compound Dirichlet

Log-Normal Distribution

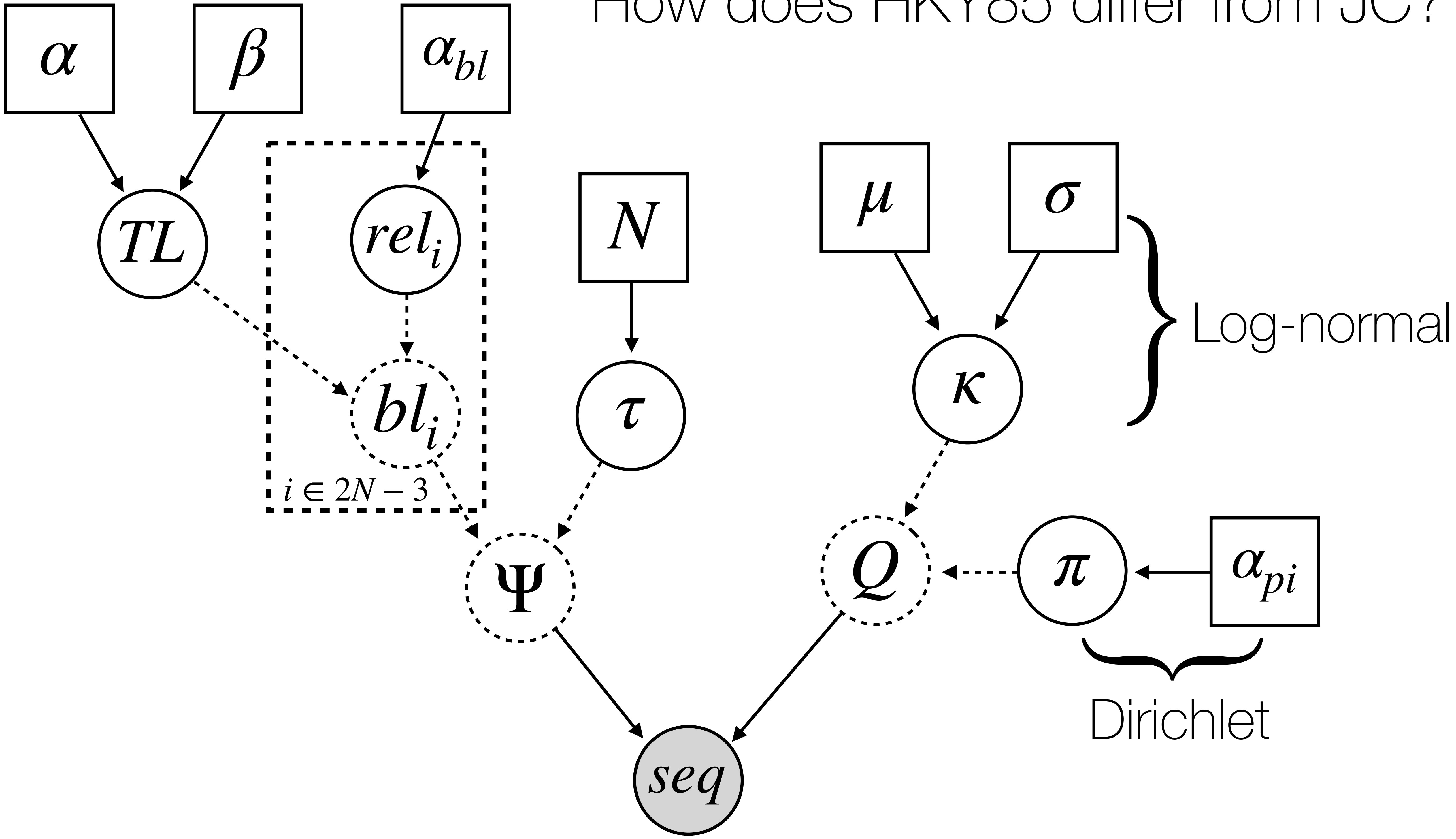


In probability theory, a log-normal (or lognormal) distribution is a continuous probability distribution of a random variable whose logarithm is normally distributed.

https://en.wikipedia.org/wiki/Log-normal_distribution

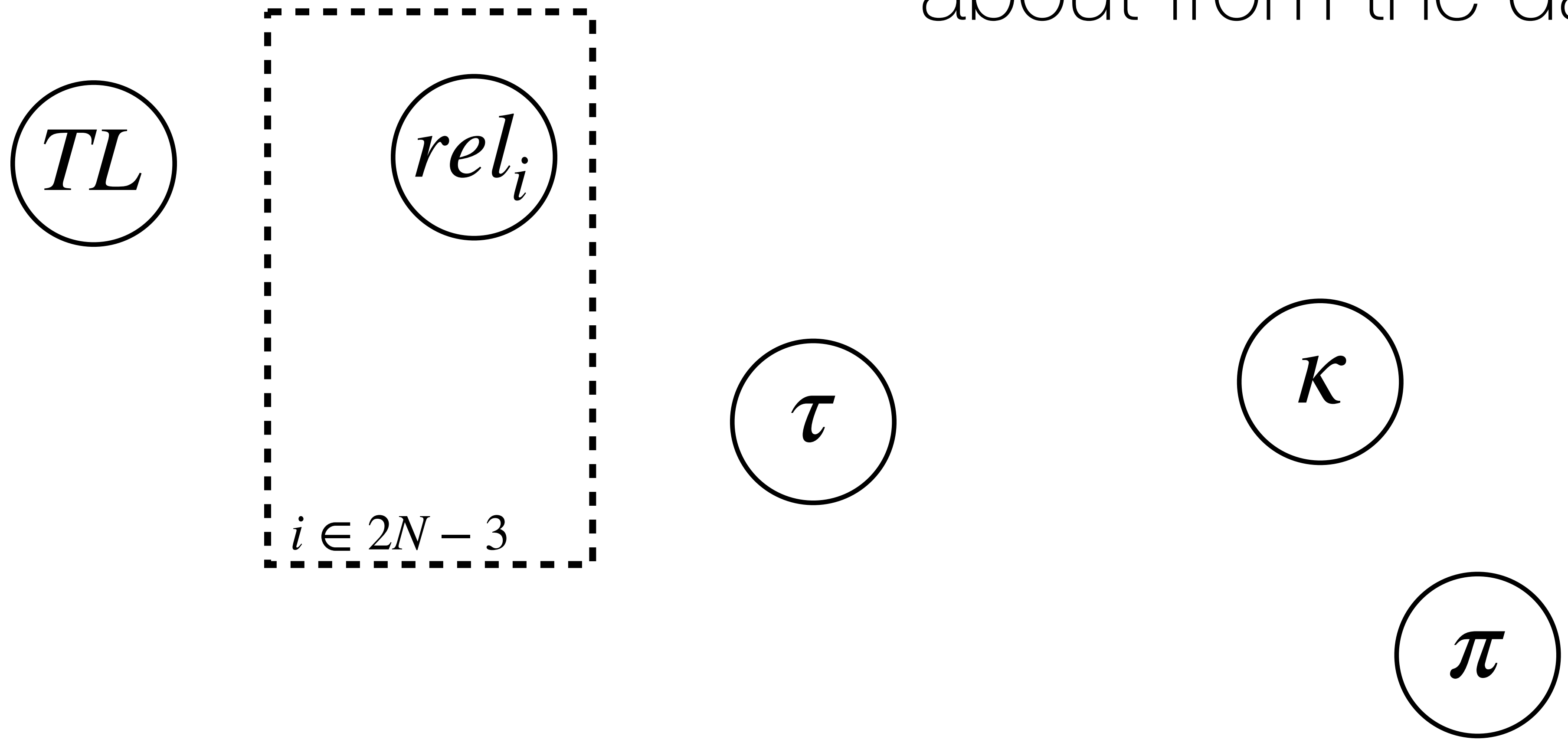
HKY85

How does HKY85 differ from JC?

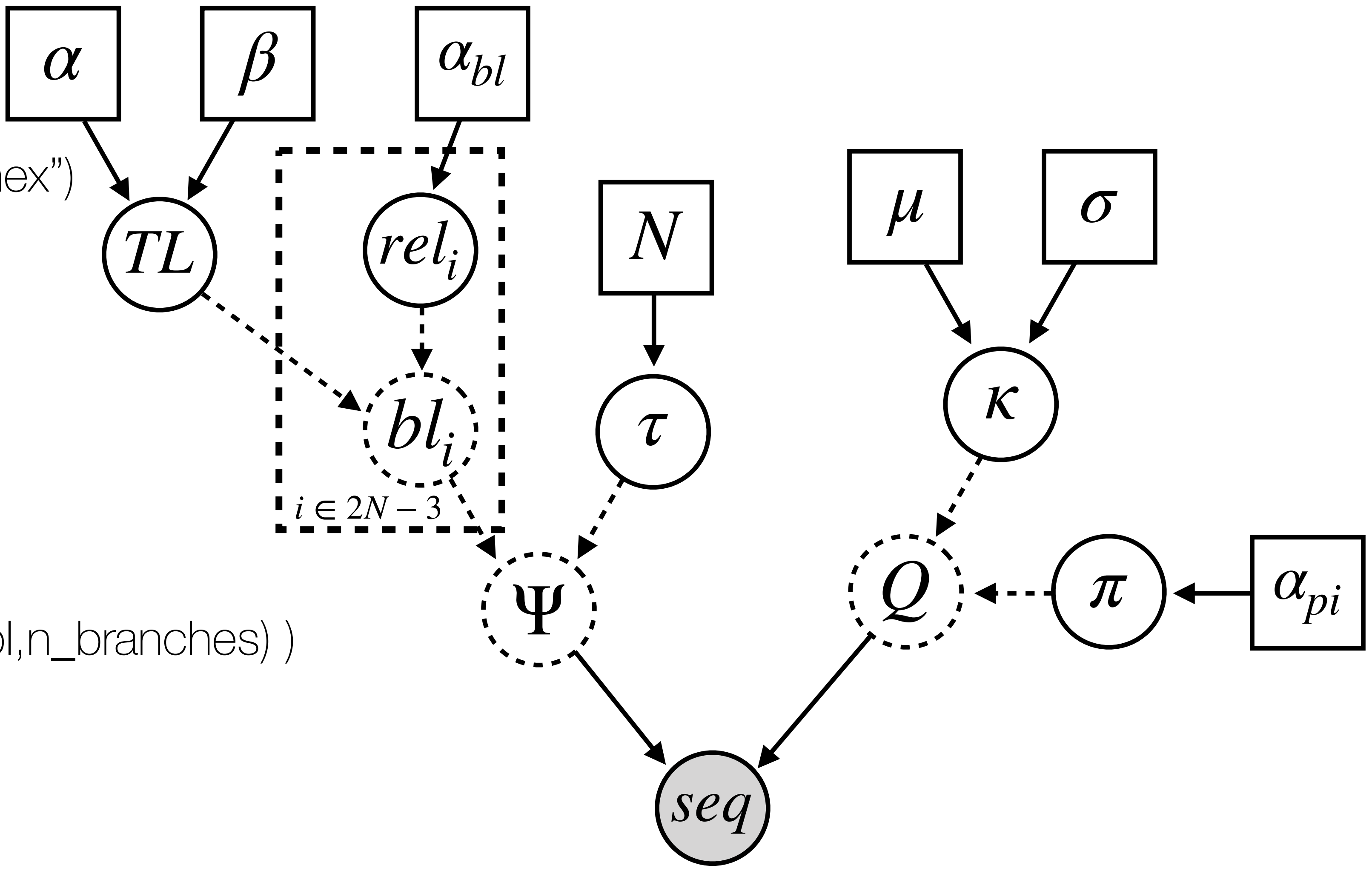


HKY85

Remember, these are the parameters whose values we're inferring (learning about from the data).



```
data = readDiscreteCharacterData("myData.nex")
taxa <- data.taxa()
n_taxa <- data.ntaxa()
n_branches <- 2 * n_taxa - 3
alpha <- 2
beta <- 4
TL ~ dnGamma(alpha,beta)
alpha_bl <- 1.0
rel_branch_lengths ~ dnDirichlet( rep(alpha_bl,n_branches) )
br_lens := rel_branch_lengths * TL
topology ~ dnUniformTopology(taxa)
psi := treeAssembly(topology, br_lens)
```



```
mu <- 0
sigma <- 1
kappa ~ dnLognormal(mu,sigma)

alpha_pi <- 1.0
pi ~ dnDirichlet( rep(alpha_pi,4) )

Q <- fnHKY(kappa,pi)
```

HKY85

```
seq ~ dnPhyloCTMC(tree=psi,Q=Q,type="DNA")
seq.clamp(data)
```

Things to Remember

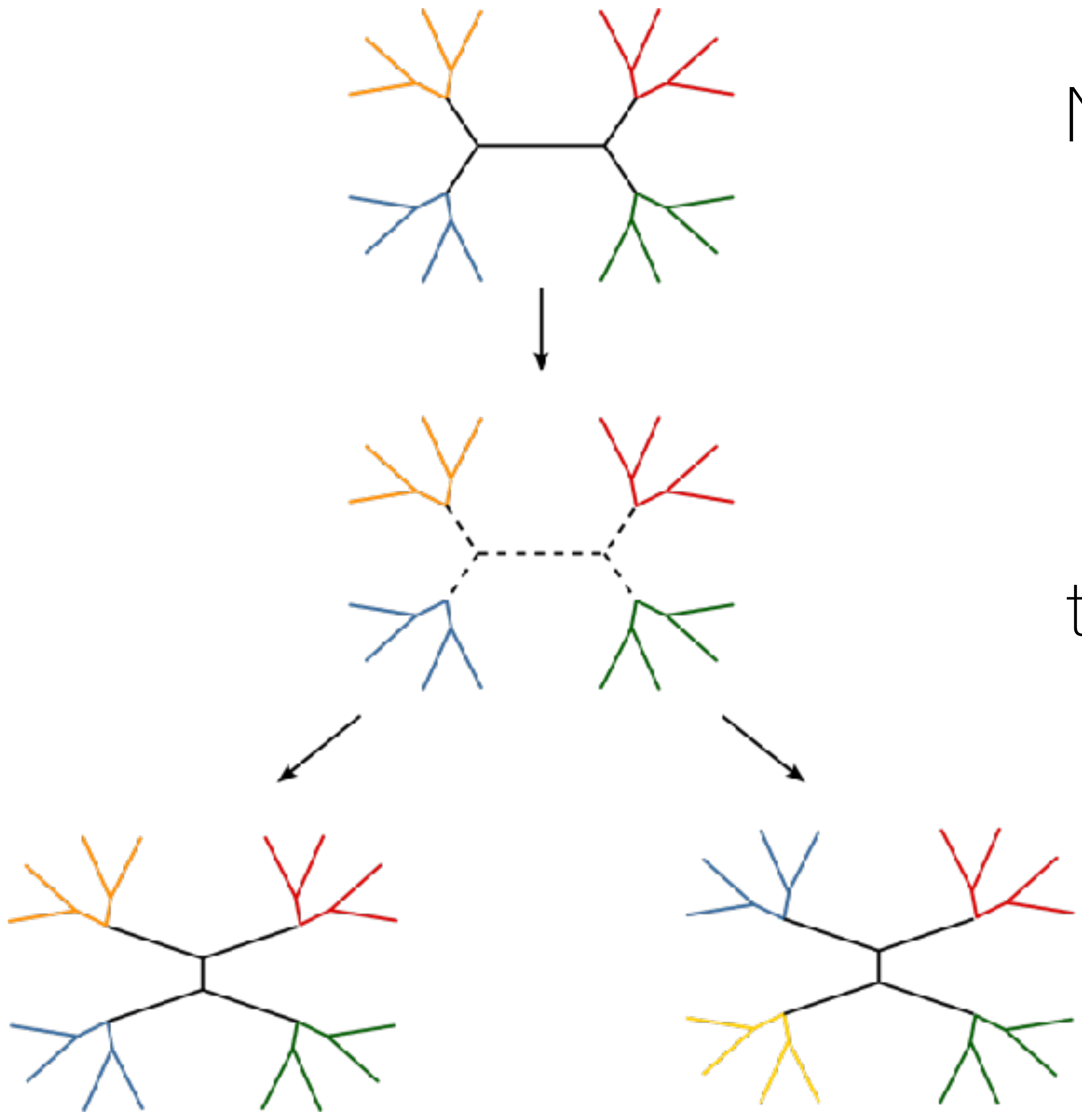
As we add more stochastic nodes, remember to assign moves to all of them!

Vectors modeled with a Dirichlet need to have a joint proposal mechanism, since they need to sum to 1. Here are a couple:

mvBetaSimplex
mvDirichletSimplex

You'll often want to run analyses on the same dataset with multiple models. Start by saving your simplest model into a text file (.rev). Then, copy and paste the code into a file for the next model and adjust as necessary. Keep going as you work your way to more complex models.

Topology Moves

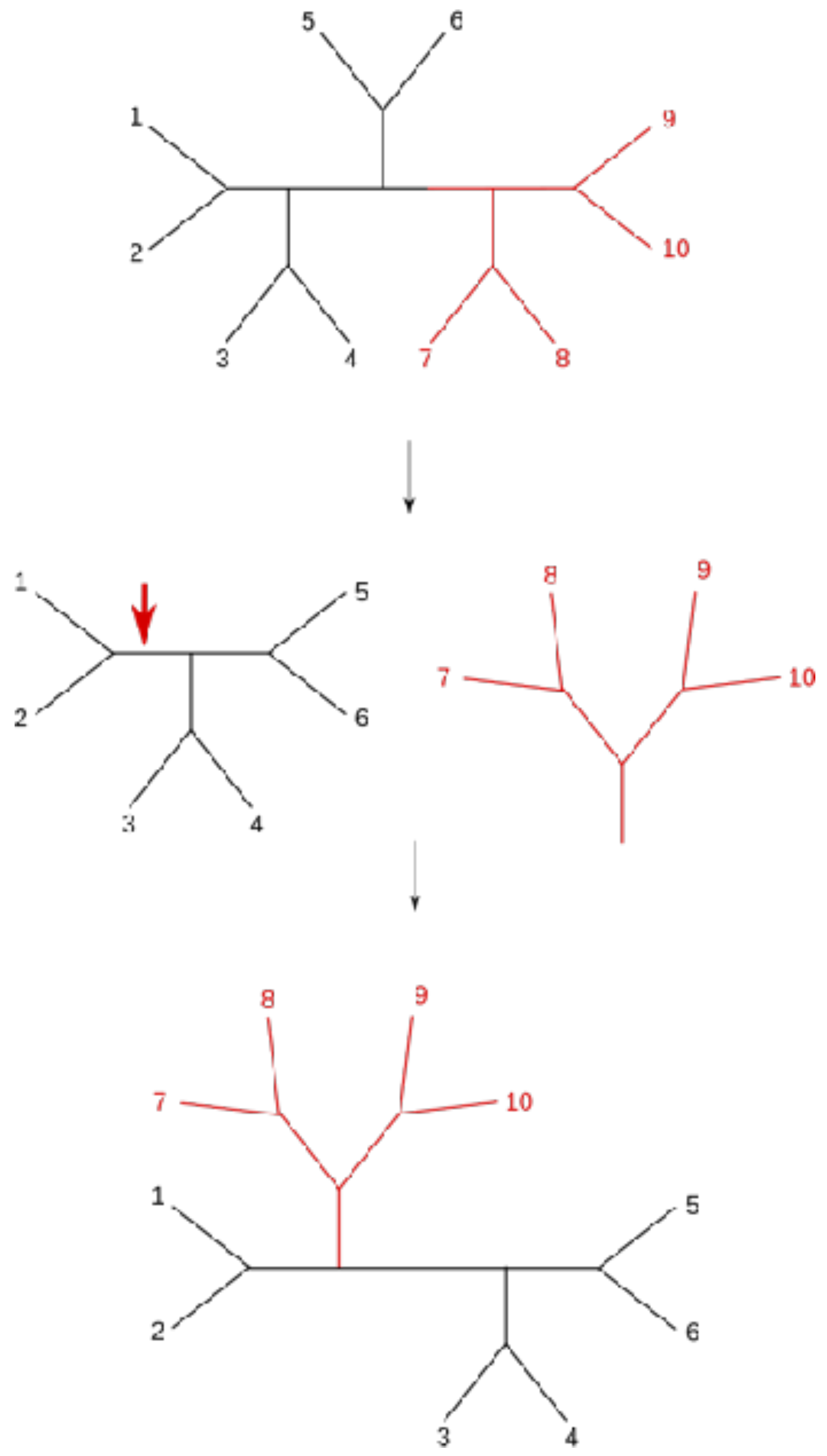


Nearest-Neighbor Interchange (NNI)

The most minor topological move.

https://en.wikipedia.org/wiki/Tree_rearrangement

Topology Moves



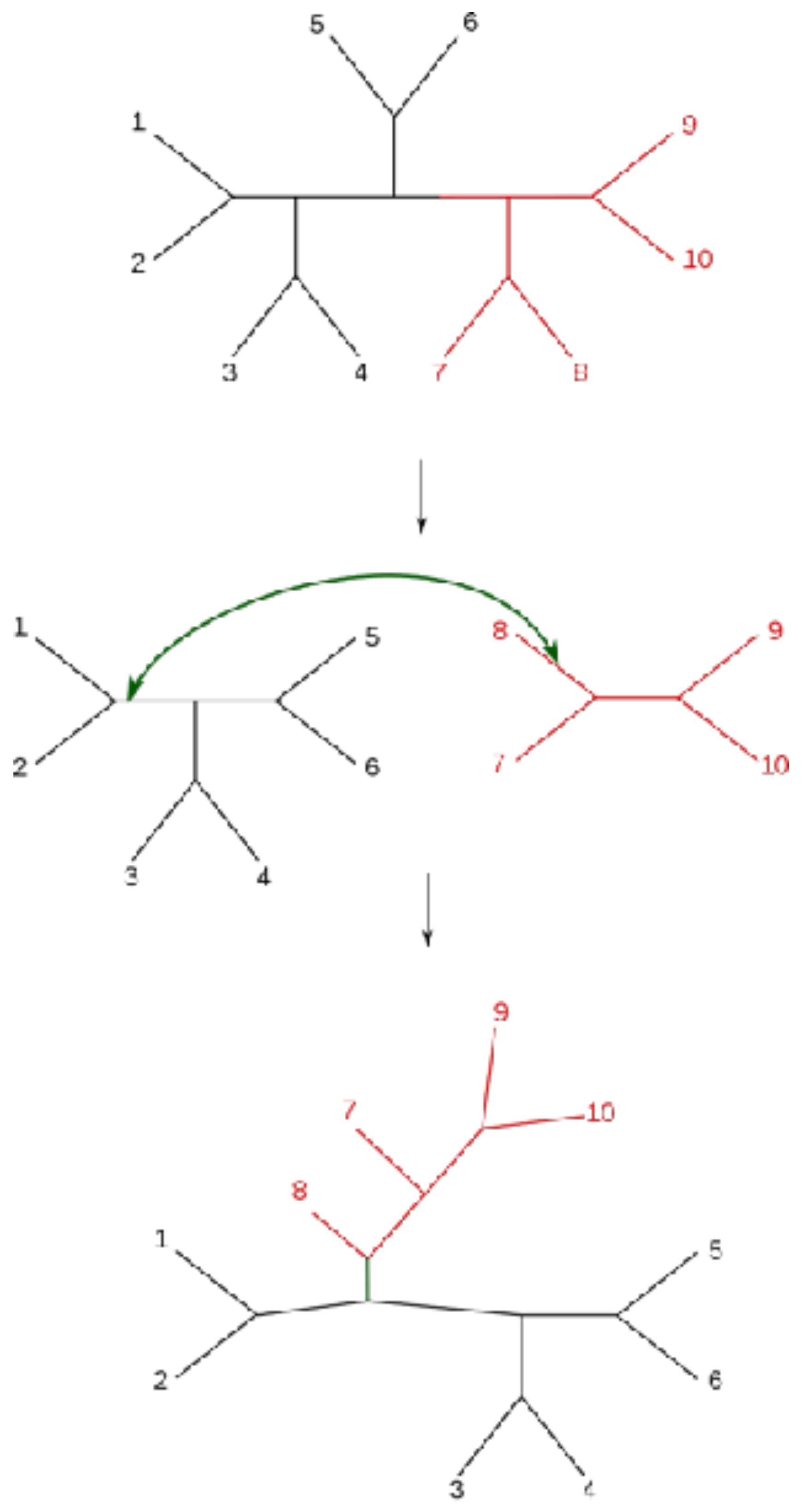
Subtree Prune and Regraft (SPR)

Intermediate topological move.

Often a good balance between exploring new tree space without disrupting things too much.

https://en.wikipedia.org/wiki/Tree_rearrangement

Topology Moves



Tree Bisection and Reconnection (TBR)

Disruptive topological move.

https://en.wikipedia.org/wiki/Tree_rearrangement

Summarizing Phylogenetic MCMC Output

For scalar (numerical) parameters, we often report:

Mean

Median

95% Highest Posterior Density Interval

95% Credibility Interval (often similar to HPD)

Summarizing Phylogenetic MCMC Output

For trees, we often report:

Maximum A Posteriori (MAP) Tree

("Best" Tree)

Majority-Rule Consensus Tree

(Formed from all bipartitions with posterior > 0.5)

Greedy Consensus Tree

(Formed by ranking bipartitions and adding them until tree is fully resolved)

Maximum Clade Credibility Tree

(Sampled tree whose product of bipartition probabilities is greatest)

95% Credible Set

(Set of most probable trees whose total probability is >= 95%)

RevBayes CTMC Tutorial

https://revbayes.github.io/tutorials/ctmc/

Look at Table 1 in the tutorial above.

Using the model descriptions in the table, use the same principles we've been practicing to set up each of these models in RevBayes.

Fill out Table 2 for all the models that don't include +Gamma or +I. (we'll get to those shortly)

Keep your .rev files and inferred clade probabilities. Before next week, once you've worked through all the models, submit your .rev files and inferred probabilities to the homework folder on GitHub.