Ideas on sampling branch lengths given two sequences

The requirement is to find a method that is reasonably fast to generate at random (with a probability density that can be computed) branch lengths for a tree topology that are not too different from an actual posterior distribution for the purposes of implementing an importance sampling method for Bayesian phylogenetics. The approach outlined here requires a method to generate a random distance between two "sequences", where "sequences" is quoted because a sequence will be generalized to a site-wise independent probability distribution.

Jukes-Cantor model.— Data: X= number of sites that differ between 2 sequences; n= total number of sites

$$\begin{aligned} X|p &\sim Binomial(n,p) \\ p &\sim Beta(\alpha_0,\beta_0) \\ p|X &\sim Beta(\alpha=\alpha_0+X,\beta=\beta_0+n-X) \end{aligned}$$

Idea:

1. Draw beta distribution
2. Rescale to $(0,3/4)$ instead of $(0,1)$
3. Plug into $\hat{t}_{MLE}$ formula:

$$\hat{t}_{MLE} = -\frac{3}{4}\log\left(1-\frac{4}{3}\frac{x}{n}\right)$$

We introduce the parameter η to control the variance of the JC density for T:

$$\begin{aligned} \frac{x}{n} &= \frac{3\alpha}{a(\alpha+\beta)} \\ \alpha+\beta &= \eta n \end{aligned}$$

Note that the presence of η is multiplying the beta variance by $1/\eta$.

So, the parameters for beta are

$$\begin{aligned} \alpha &= \frac{4}{3}\eta x \\ \beta &= \eta n - \frac{4}{3}\eta x \end{aligned}$$

Procedure:

1. Simulate $W \sim Beta(\alpha,\beta)$
2. Rescale $0.75W = x/n$
3. Compute $T = -0.75\log(1-W)$

We could also simulate $1-W \sim Beta(\beta,\alpha)$.

Code: `branch-length.r`

```
> Q = randomQ(n=4,rescale=TRUE)
> x = simulateSequenceSummary(nsites=500,Q=Q,s=0.15)
> w = simulateBranchLength.jc(nsim=100,x=x,eta=0.8)
```

Tamura-Nei model.— add assumptions

$$
\begin{aligned}
numer1 &= 2 * p.a * p.g * p.r \\
denom1 &= numer1 - p.r^2 * prop.ag - p.a * p.g * prop.tv \\
c1 &= numer1/denom1 \\
numer2 &= 2 * p.c * p.t * p.y \\
denom2 &= numer2 - p.y^2 * prop.ct - p.c * p.t * prop.tv \\
c2 &= numer2/denom2 \\
c3 &= (2 * p.a^2 * p.g^2)/(p.r * denom1) + (2 * p.c^2 * p.t^2)/(p.y * denom2) \\
&\quad + (p.r^2 * (p.c^2 + p.t^2) + p.y^2 * (p.a^2 + p.g^2))/(2 * p.r^2 * p.y^2 - p.r * p.y * prop.tv) \\
mu &= -2 * ((p.a * p.g/p.r) * log(1 - p.r * prop.ag/(2 * p.a * p.g) - prop.tv/(2 * p.r)) \\
&\quad + (p.c * p.t/p.y) * log(1 - p.y * prop.ct/(2 * p.c * p.t) - prop.tv/(2 * p.y)) \\
&\quad + (p.r * p.y - p.a * p.g * p.y/p.r - p.c * p.t * p.r/p.y) * log(1 - prop.tv/(2 * p.r * p.y))) \\
v &= (1/eta) * ((c1^2 * prop.ag + c2^2 * prop.ct + c3^2 * prop.tv) \\
&\quad - (c1 * prop.ag + c2 * prop.ct + c3 * prop.tv)^2)/n
\end{aligned}
$$

Note that the presence of η is multiplying the beta variance by $1/\eta$.

Procedure:

1. Simulate $W \sim Gamma(\mu^2/v, \mu/v)$

Code: `branch-length.r`

```
> Q = randomQ(n=4,rescale=TRUE)
> x = simulateSequenceSummary(nsites=500,Q=Q,s=0.15)
> w = simulateBranchLength.tn(nsim,x,eta=0.9)
```

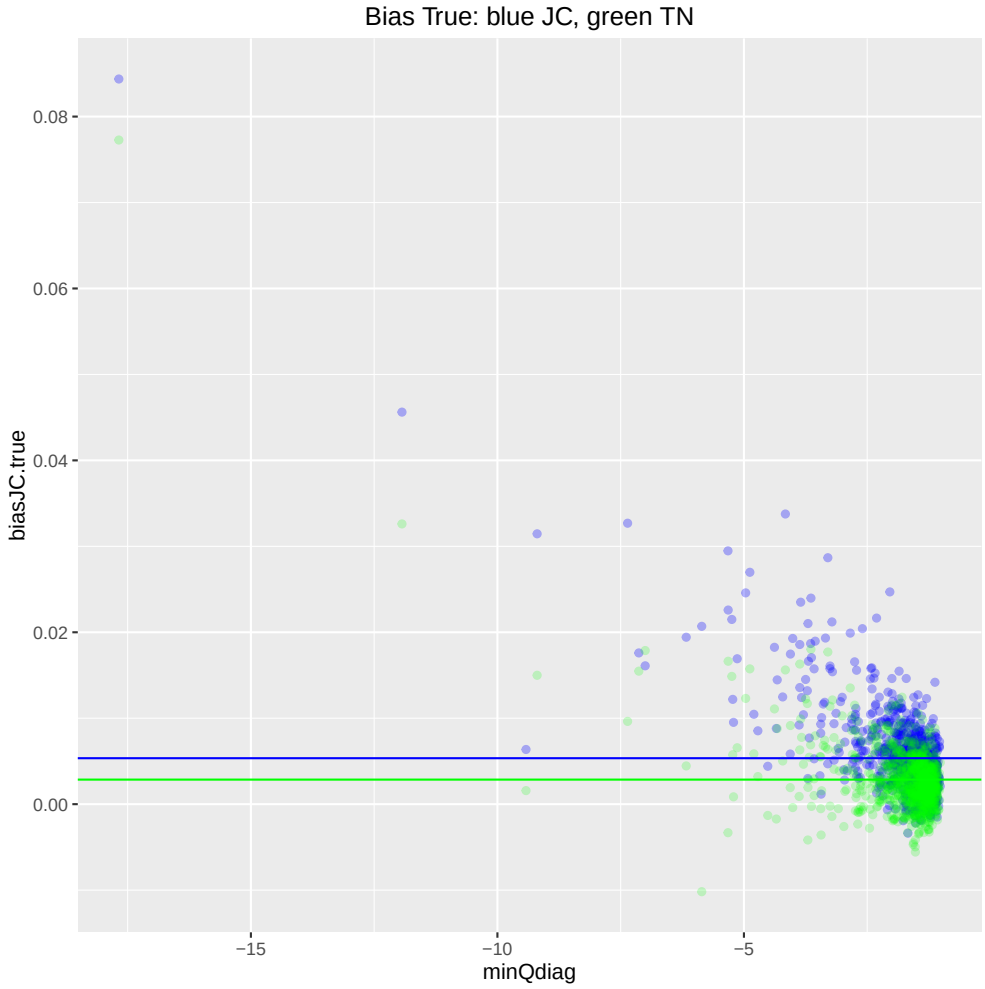
1 Comparison JC vs TN

1.1 Bias

Code: `JCvsTN.r`

```
> s=0.15
> nrep=1000
> nsites=1000
> eta=0.8
```

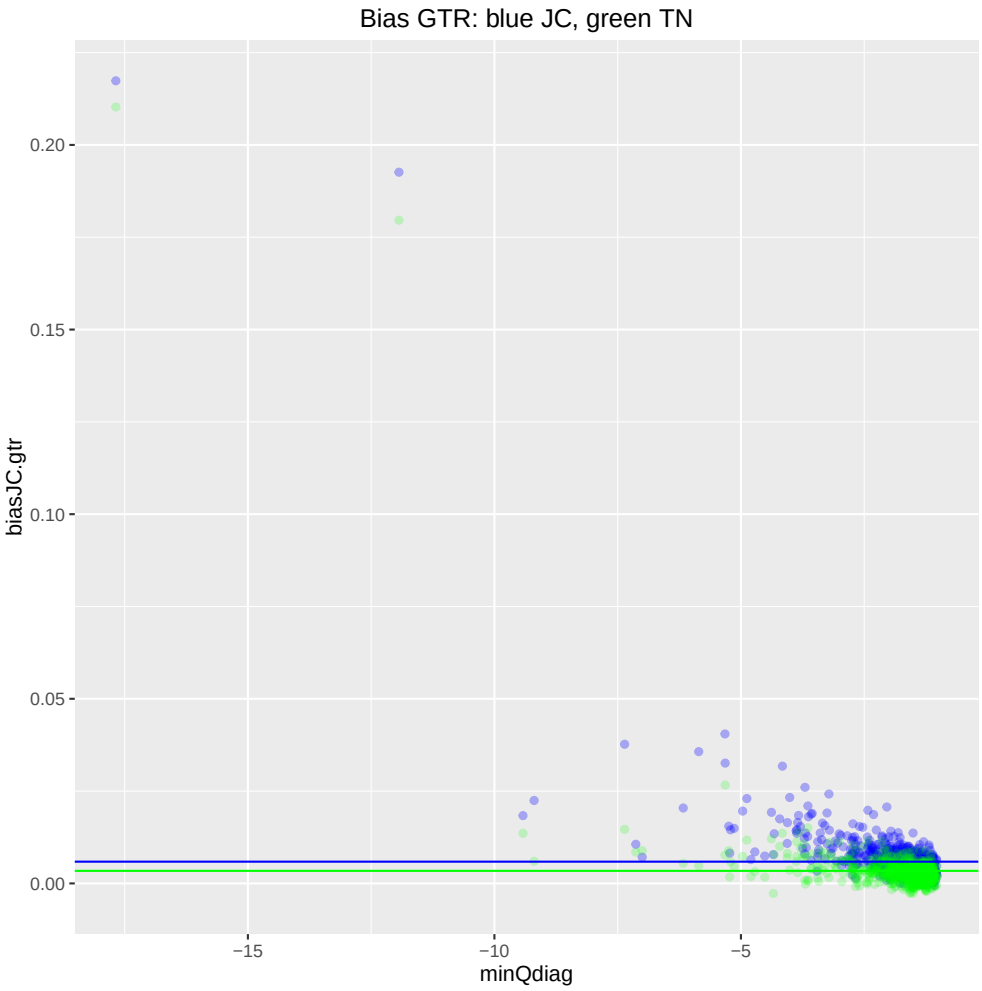
- bias=true mode - JC(TN) mode
- Most times bias positive (~ 0.926)
- Most times (~ 0.99), bias less than 0.02 (spread density ~ 0.1).

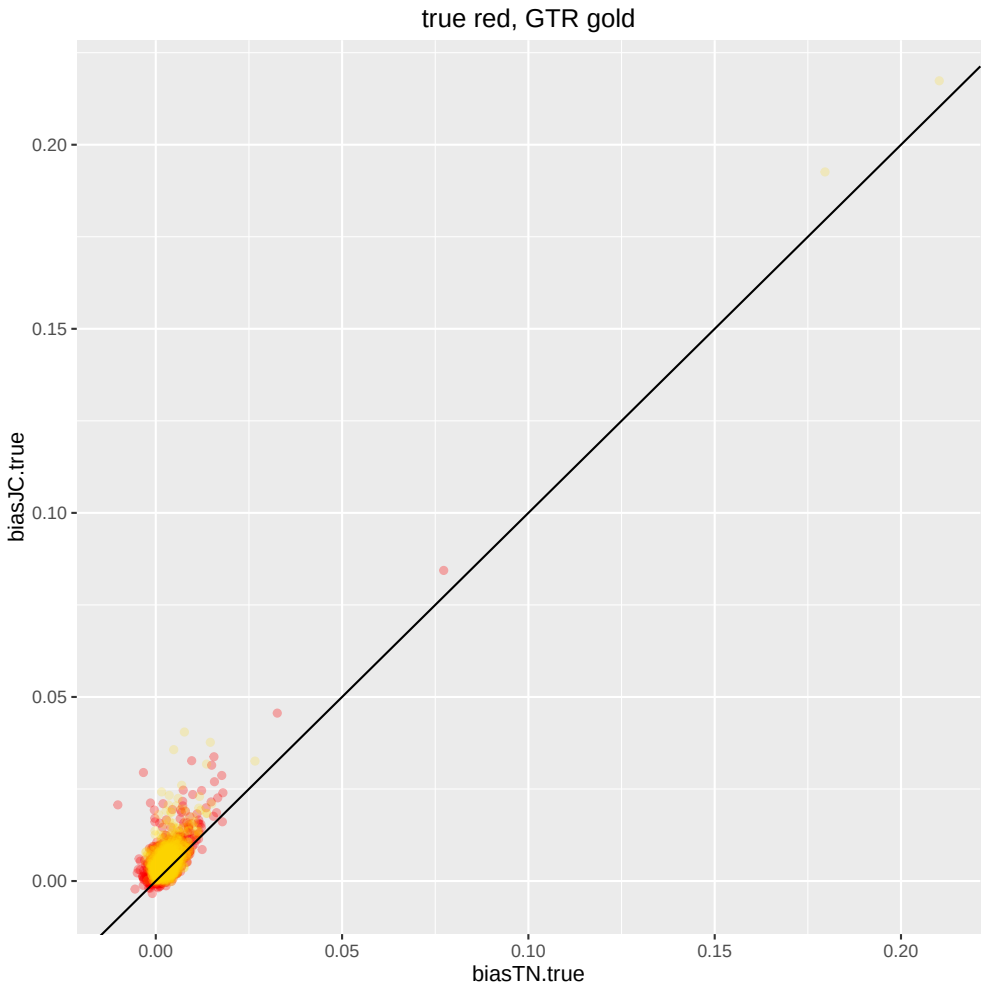


- JC bias is systematically bigger than TN bias (see figures).
- When one of them is surprisingly big, the other is too
- Big biases associated to too negative diagonal values in Q (< -5.0)

Problems with TN

- Errors with rows (or columns) of zero in the count matrix
- Errors if row (or column) only one entry different than zero (or close to vector of zeros): NaN caused by negative argument in the logarithm.
- This is also associated with very small values in the diagonal of Q matrix, but not as small as causing big bias (< -2.2).


Bias GTR: blue JC, green TN
biasJC.gtr
0.20
0.15
0.10
0.05
0.00
−15
−10
−5
minQdiag


true red, GTR gold
biasJC.true
biasTN.true
0.00
0.05
0.10
0.15
0.20

1.2 Coverage

Code: `JCvsTN.r`

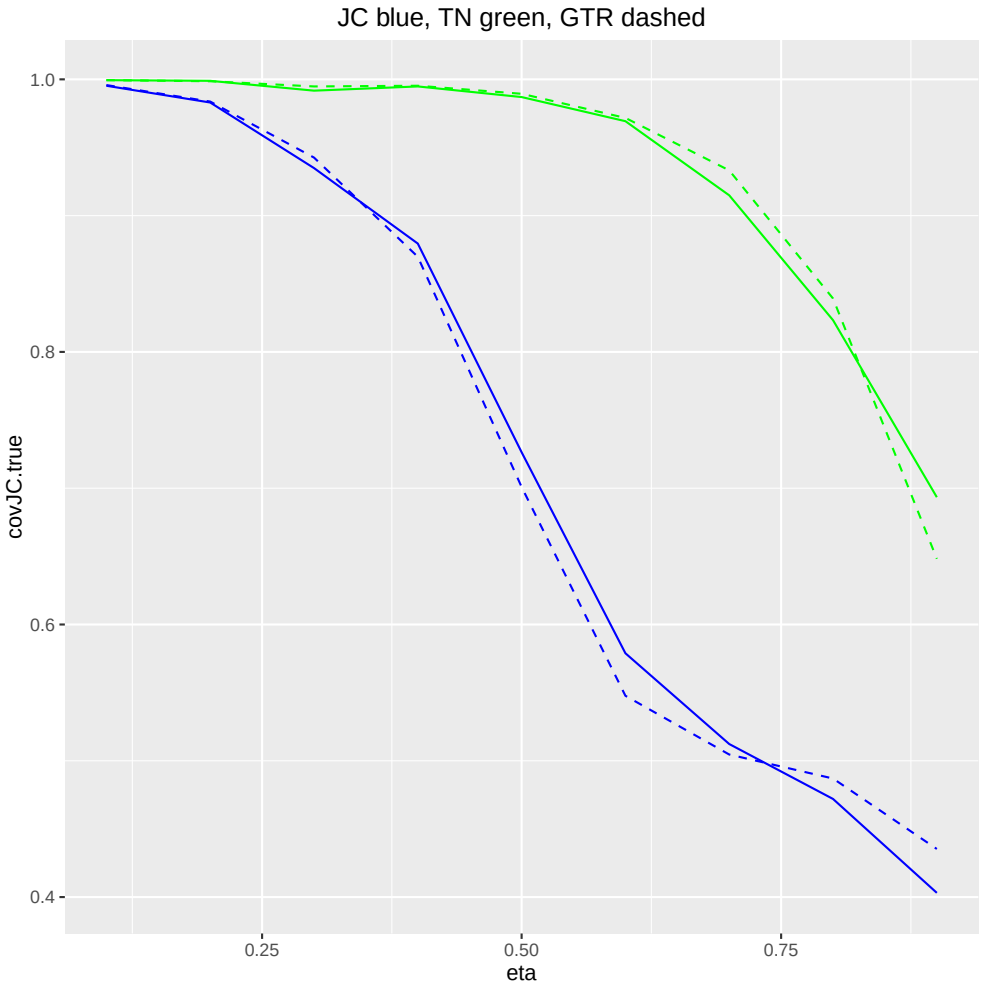


Figure 1: Percentage of coverage with the cutoff method

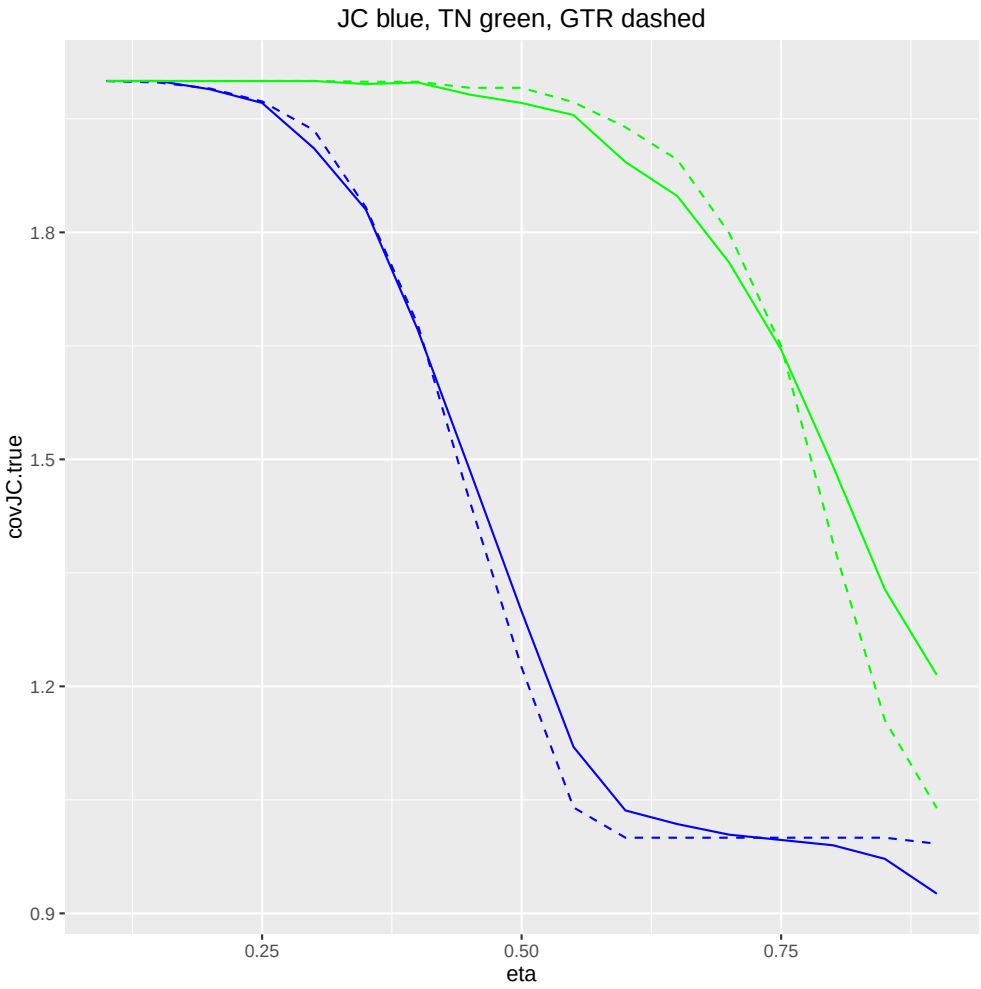


Figure 2: Percentage of coverage with the 95% CI

1.2.1 Densities

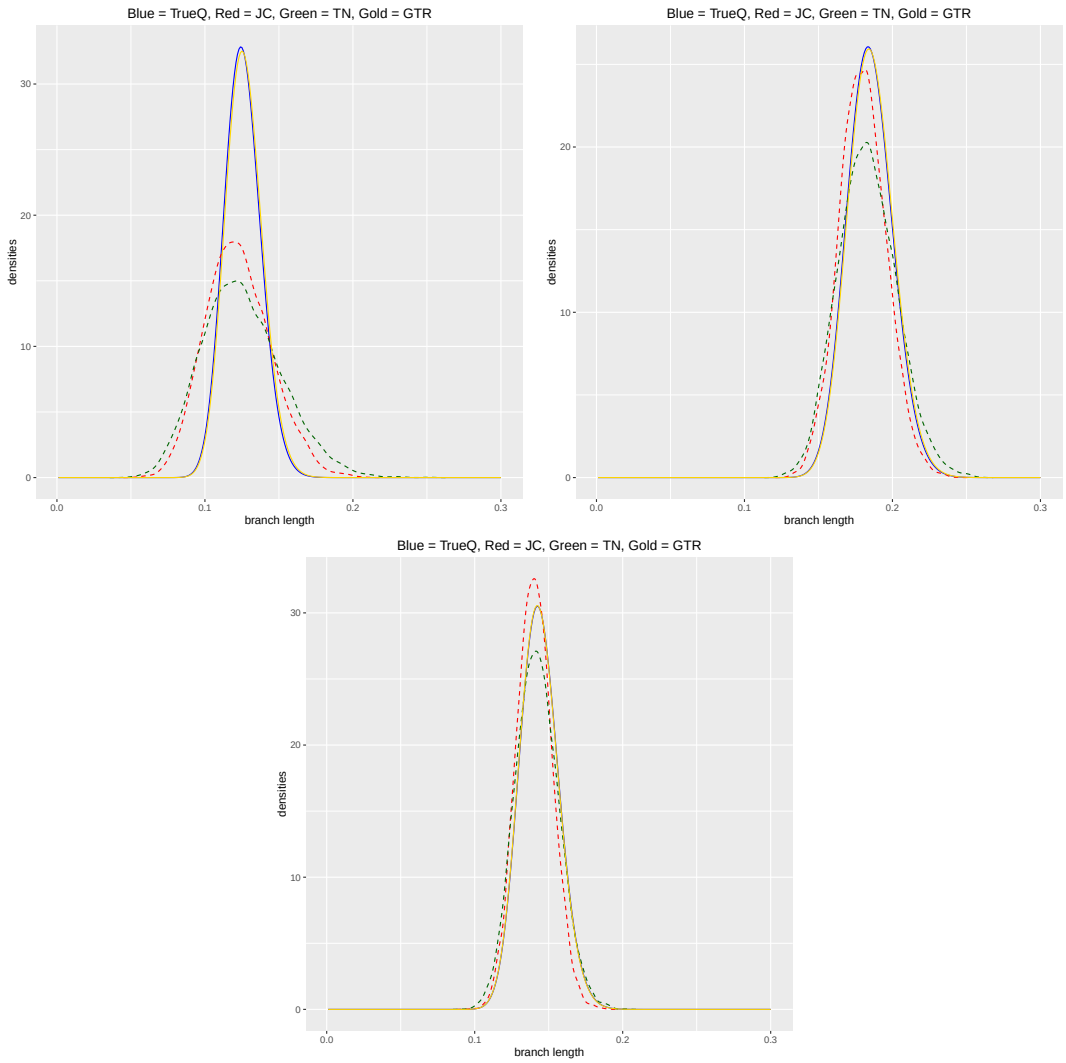


Figure 3: $\eta = 0.2, 0.6, 0.8$