Simple neighbor-joining example with nucelotide and "geography" data

Andy Magee

Combining data types in neighbor joining

Here we investigate a heuristic way to combine geographic and nucleotide data.

The heuristic

To combine data, we will separately estimate distance matrices for nucleotide data, d_{nuc}, and geographic data, d_{geo}. Then we will combine them by adjusting for both the relative rate of change in both datasets and the numbers of characters in each. This will yield the equation $d_{tot} = (n_{sites}/(n_{sites}+1)) \times d_{nuc} + (1/(n_{sites}+1)) \times r_{rel} \times d_{geo}$, where there are n_{sites} sites in the nucleotide aligment and r_{rel} is a relative rate multiplier that accounts for the different rates of change in the two datasets. Specifically, we seek to make the distances equivalent between the two data sources. If we had the clock rates for both datasets, r_{nuc} and r_{geo}, then we would simply have $r_{rel} = r_{nuc}/r_{geo}$. In a run of TreeTime, we would have estimates of these at best, so we can also consider estimating the relative rate. Since the time is the same for all branches, distance is proportional to rate, and so we can estimate this by, say, dividing the sum of all distances in the nucleotide dataset by the sum of all distances in the geographic dataset.

Simulating trees

We start by simulating 100 trees from an epidemic, each with 100 tips. These are serially sampled trees.

```
library(TreeSim)
```

```
## Loading required package: ape
```

```
## Loading required package: geiger
```

```
library(phangorn)

set.seed(42)

# Simulating parameters (based on an HIV dataset)
ntaxa <- 100
Re <- 2.2
mu <- 0.14
phi <- 0.044
r <- 1.0
lambda <- Re * (mu + phi * r)

# make them into what sim.bdsky.stt wants
l <- lambda
d <- mu + phi * r
s <- (phi * r)/d

# Simulate trees
trees <- lapply(1:100,function(i){
```

```
  sim.bdsky.stt(ntaxa,c(l,l),c(d,d),c(0,1),c(s,s))[[1]]
})
```

The data

We now purposefully simulate sequence data that is insufficient to resolve the tree. to do this, we simulate 1000 sites, and choosing a tree length T in expected substitutions per site such that $T \times n_{sites} < n_{branches}$. We will simulate geographic data with approximately 10 changes across the tree, and for now assume there are only 4 states so we can use standard inference machinary. We will simulate both geography and DNA under GTR models, but analyze them under simpler models.

```
# Information for simulated data
n.edges <- length(trees[[1]]$edge.length)
n.sites <- 1000

nuc.nsubs.persite <- (0.8 * n.edges)/n.sites
geo.nsubs.persite <- 10

clock.nuc <- numeric(length(trees))
clock.geo <- numeric(length(trees))

er.nuc <- c(1,6,2,3,8,1)
bf.nuc <- c(0.2,0.3,0.3,0.2)

er.geo <- c(6,5,4,3,2,1)
bf.geo <- c(0.1,0.2,0.3,0.4)

# Simulate data
seqs.nuc <- vector("list",length(trees))
seqs.geo <- vector("list",length(trees))

for (i in 1:length(trees)) {
  phy <- trees[[i]]
  clock.nuc[i] <- nuc.nsubs.persite/sum(phy$edge.length)
  phy$edge.length <- phy$edge.length * clock.nuc[i]
  seqs.nuc[[i]] <- simSeq(phy,l=n.sites,Q=er.nuc,bf=bf.nuc)

  phy <- trees[[i]]
  clock.geo[i] <- geo.nsubs.persite/sum(phy$edge.length)
  phy$edge.length <- phy$edge.length * clock.geo[i]
  seqs.geo[[i]] <- simSeq(phy,l=1,Q=er.geo,bf=bf.geo)
}
```

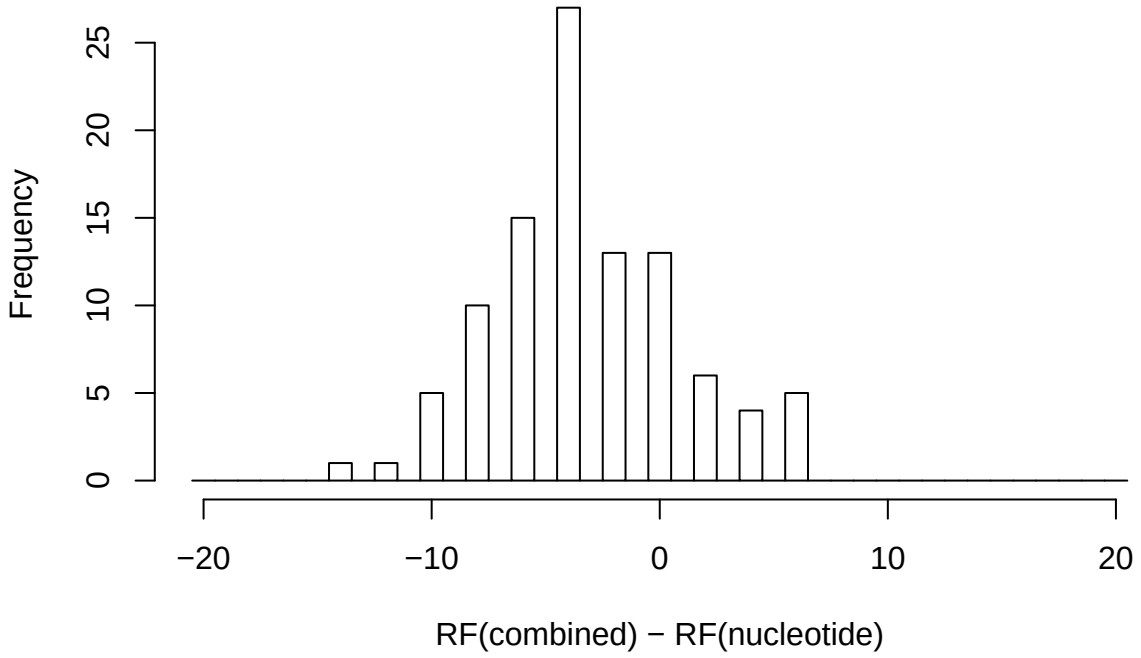
Part 1

First, let's see what happens when we use the real clock rates to make the distances comparable between geography and nucleotide data.

Here and in all sections, we analyze the DNA data with TN93 and the geography data with F81. In other words, we're using mis-specified substitution models for both types of data.

```
est.nuc <- vector("list",length(trees))
est.tot <- vector("list",length(trees))
```

```
for (i in 1:length(trees)) {
  d.nuc <- as.matrix(dist.dna(as.DNAbin(seqs.nuc[[i]]),model="TN93"))
  d.geo <- as.matrix(dist.ml(seqs.geo[[i]],model="F81"))
  rel.rate <- (clock.nuc[i]/clock.geo[i])
  d.tot <- (n.sites/(n.sites+1))*d.nuc + (1/(n.sites+1))*rel.rate*d.geo

  est.nuc[[i]] <- nj(d.nuc)
  est.tot[[i]] <- nj(d.tot)
}

rf.nuc <- numeric(length(trees))
rf.tot <- numeric(length(trees))

for (i in 1:length(trees)) {
  rf.nuc[i] <- dist.topo(unroot(trees[[i]]),est.nuc[[i]])
  rf.tot[i] <- dist.topo(unroot(trees[[i]]),est.tot[[i]])
}

hist(rf.tot-rf.nuc,breaks=seq(-20.5,20.5,1),main="",xlab="RF(combined) - RF(nucleotide)")
```



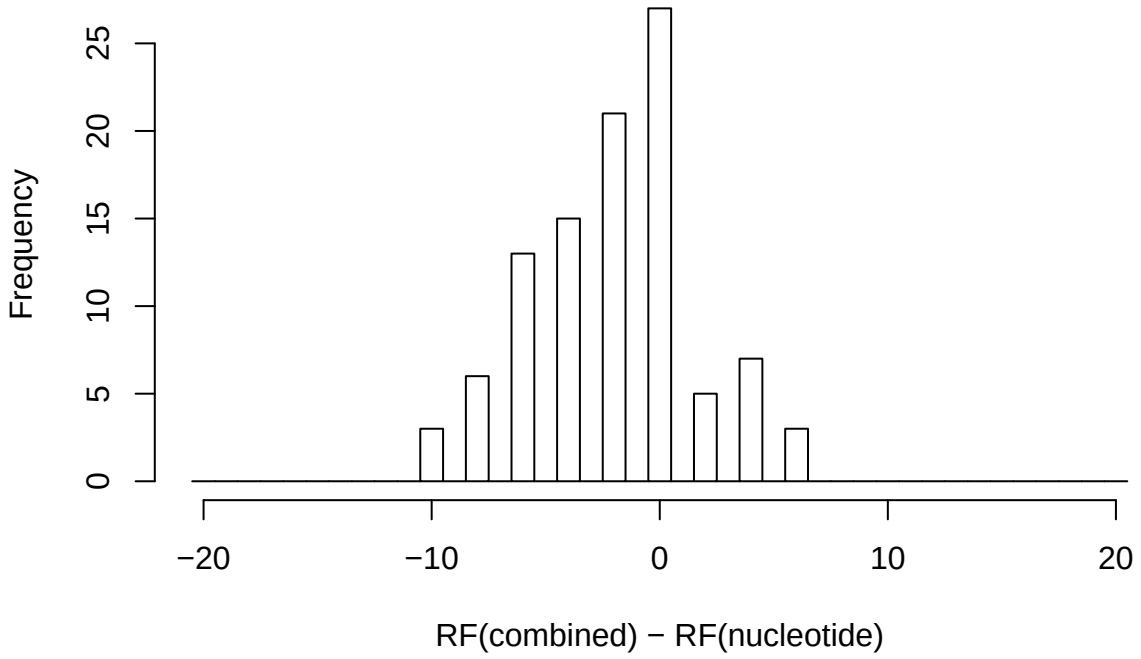
This produces 85 estimated phylogenies from the combined data that are at least as good as the phylogenies from nucleotide data alone.

Part 2

Next, let's see what happens when we use the real clock rates to make the distances comparable between geography and nucleotide data.

```
est.nuc <- vector("list",length(trees))
est.tot <- vector("list",length(trees))

for (i in 1:length(trees)) {
```

```
  d.nuc <- as.matrix(dist.dna(as.DNAbin(seqs.nuc[[i]]),model="TN93"))
  d.geo <- as.matrix(dist.ml(seqs.geo[[i]],model="F81"))
  rel.rate <- sum(d.nuc[upper.tri(d.nuc)])/sum(d.geo[upper.tri(d.geo)])
  d.tot <- (n.sites/(n.sites+1))*d.nuc + (1/(n.sites+1))*rel.rate*d.geo

  est.nuc[[i]] <- nj(d.nuc)
  est.tot[[i]] <- nj(d.tot)
}

rf.nuc <- numeric(length(trees))
rf.tot <- numeric(length(trees))

for (i in 1:length(trees)) {
  rf.nuc[i] <- dist.topo(unroot(trees[[i]]),est.nuc[[i]])
  rf.tot[i] <- dist.topo(unroot(trees[[i]]),est.tot[[i]])
}

hist(rf.tot-rf.nuc,breaks=seq(-20.5,20.5,1),main="",xlab="RF(combined) - RF(nucleotide)")
```



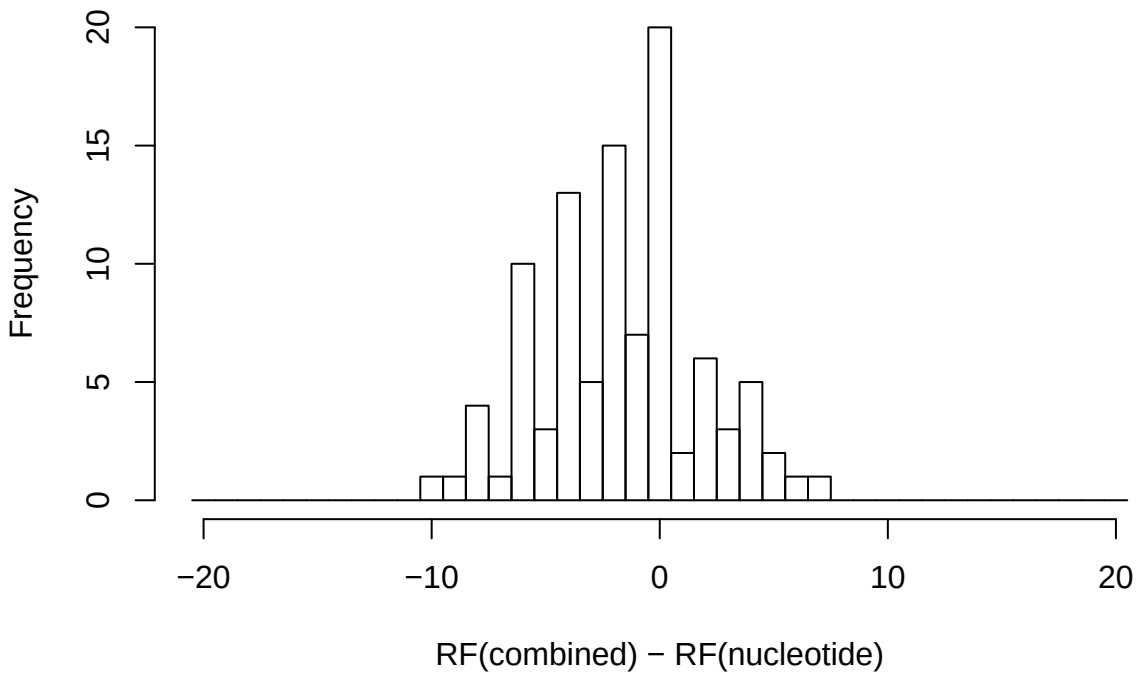
This produces 85 estimated phylogenies from the combined data that are at least as good as the phylogenies from nucleotide data alone.

Part 3

Lastly, we can also consider collapsing branches under some threshold. Here we use 1e-8, which mostly collapses 0-length and negative branches. Here again we use an estimate of the relative rate/adjustment factor between geography and nucleotide data.

```
# Function to collapse short branches
collapseBranches <- function(phy,min.len=0) {
  # recover()
  ntaxa <- length(phy$tip.label)
```

```
  toremove <- which(phy$edge.length < min.len & phy$edge[,2] > ntaxa)
  while ( length(toremove) > 0 ) {
    idx <- toremove[1]
    parent <- phy$edge[idx,1]
    child <- phy$edge[idx,2]
    grandchildren_indices <- which(phy$edge[,1] == child)
    phy$edge[grandchildren_indices,1] <- parent
    phy$edge <- phy$edge[-idx,]
    phy$edge.length  <- phy$edge.length[-idx]
    toremove <- which(phy$edge.length < min.len & phy$edge[,2] > ntaxa)
  }
  phy$Nnode <- length(phy$edge.length) - length(phy$tip.label) + 1
  new.nums <- length(phy$tip.label) + 1:phy$Nnode
  old.nums <- sort(unique(phy$edge[,1]))
  for (i in 1:length(old.nums)) {
    phy$edge[phy$edge[,1] == old.nums[i],1] <- new.nums[i]
    phy$edge[phy$edge[,2] == old.nums[i],2] <- new.nums[i]
  }
  return(phy)
}

est.nuc <- vector("list",length(trees))
est.tot <- vector("list",length(trees))

for (i in 1:length(trees)) {
  d.nuc <- as.matrix(dist.dna(as.DNAbin(seqs.nuc[[i]]),model="TN93"))
  d.geo <- as.matrix(dist.ml(seqs.geo[[i]],model="F81"))
  # rel.rate <- (clock.nuc[i]/clock.geo[i])
  rel.rate <- sum(d.nuc[upper.tri(d.nuc)])/sum(d.geo[upper.tri(d.geo)])
  d.tot <- (n.sites/(n.sites+1))*d.nuc + (1/(n.sites+1))*rel.rate*d.geo

  est.nuc[[i]] <- collapseBranches(nj(d.nuc),1e-8)
  est.tot[[i]] <- collapseBranches(nj(d.tot),1e-8)
}

rf.nuc <- numeric(length(trees))
rf.tot <- numeric(length(trees))

for (i in 1:length(trees)) {
  rf.nuc[i] <- dist.topo(unroot(trees[[i]]),est.nuc[[i]])
  rf.tot[i] <- dist.topo(unroot(trees[[i]]),est.tot[[i]])
}

hist(rf.tot-rf.nuc,breaks=seq(-20.5,20.5,1),main="",xlab="RF(combined) - RF(nucleotide)")
```



This produces 80 estimated phylogenies from the combined data that are at least as good as the phylogenies from nucleotide data alone.

Conclusion

While the parameters used here may not be the most realistic, there seems to be hope for both using neighbor joining to try resolving polytomies and to combine nucleotide and geographic data before estimating time trees.