# Phylogenetic least squares estimation without genetic distances

Extended Vignette

2024-06-08

## Preliminaries

First, two packages need to be installed:

- `robustDist`
- `phyloLSnoDist`

Installation of `robustDist` can be accomplished with the following:

```
install.packages("robustDist", repos=c("http://R-Forge.R-project.org",
                                       "http://cran.at.r-project.org"),dependencies=TRUE))
```

You may also need to include the option `INSTALL_opts = c('--no-lock')`.

Then, the easiest way to install `phyloLSnoDist` will be to use `install_github` from the `devtools` package:

```
install.packages("devtools") # if not already installed
library(devtools)
install_github("peterbchi/phyloLSnoDist")
```

## Phylogenetic Inference: Simple Example

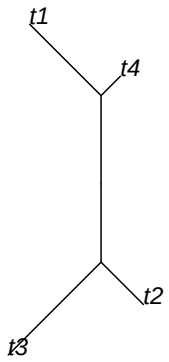
Let us generate some data on which to perform phylogenetic inference. We now load the `phyloLSnoDist` package, which should also load its dependencies:

```
library(phyloLSnoDist)
```

### 4-taxon tree

We generate a 4-taxon tree which we will use for demonstration:

```
set.seed(100)
my_tree <- rtree(4)
plot(my_tree, type='u')
```

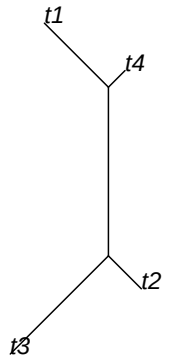



With this tree, let us generate some DNA nucleotide sequence data, and convert it to `phyDat` format for usage by the `phylo.ls.nodist` function:

```
my_DNA <- simSeq(my_tree, 2000)
my_DNA_pD <- as.phyDat(my_DNA)
```

Then, we run our `phylo.ls.nodist` function to run phylogenetic inference with our new loss function. Since there are only three unrooted topologies, we do an exhaustive search by setting `search.all = TRUE`:

```
nodist_tree <- phylo.ls.nodist(my_DNA_pD, search.all = TRUE)
plot(nodist_tree, type='u')
```

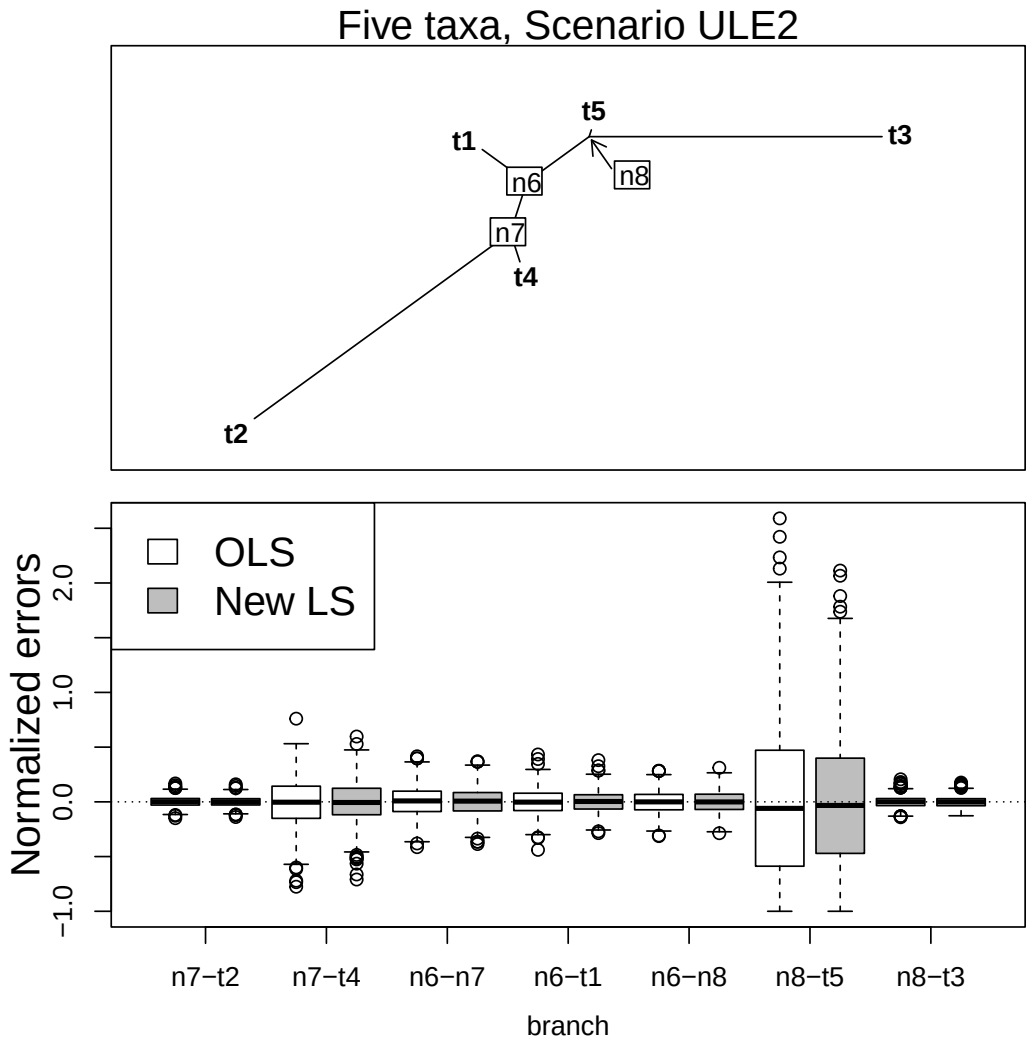


## Manuscript Analysis Replication

We will now demonstrate some of the analyses from our manuscript (Chi and Minin 2024).

### Figure 2: Branch Length estimation on a fixed 5-taxon tree

In the manuscript, this figure is shown:




The full code to run the simulations to create this figure are in the file called `Fig_errors_5taxa.R` in the `analysis` directory of our github repository (`https://github.com/peterbchi/phyloLSnoDist`). We walk through an abbreviated version below.

#### Setup

First, we load the appropriate 5-taxon tree file, contained in the `tree_files` subdirectory within the `analysis` directory in the github repository, or also in the `Vignette` directory. We also get things set up for simulations:

```
my.tree<-read.newick("sim_phylo5-9.tree")
n.br<-length(unroot(my.tree)$edge.length)

# create storage matrices
reps <- 10        # This can be bumped up; 10000 reps will take about 1.5 hours
new.brlen<-matrix(NA,nrow=reps,ncol=n.br)
reg.brlen<-matrix(NA,nrow=reps,ncol=n.br)
new.errors<-matrix(NA,nrow=reps,ncol=n.br)
reg.errors<-matrix(NA,nrow=reps,ncol=n.br)
```

### Run simulation

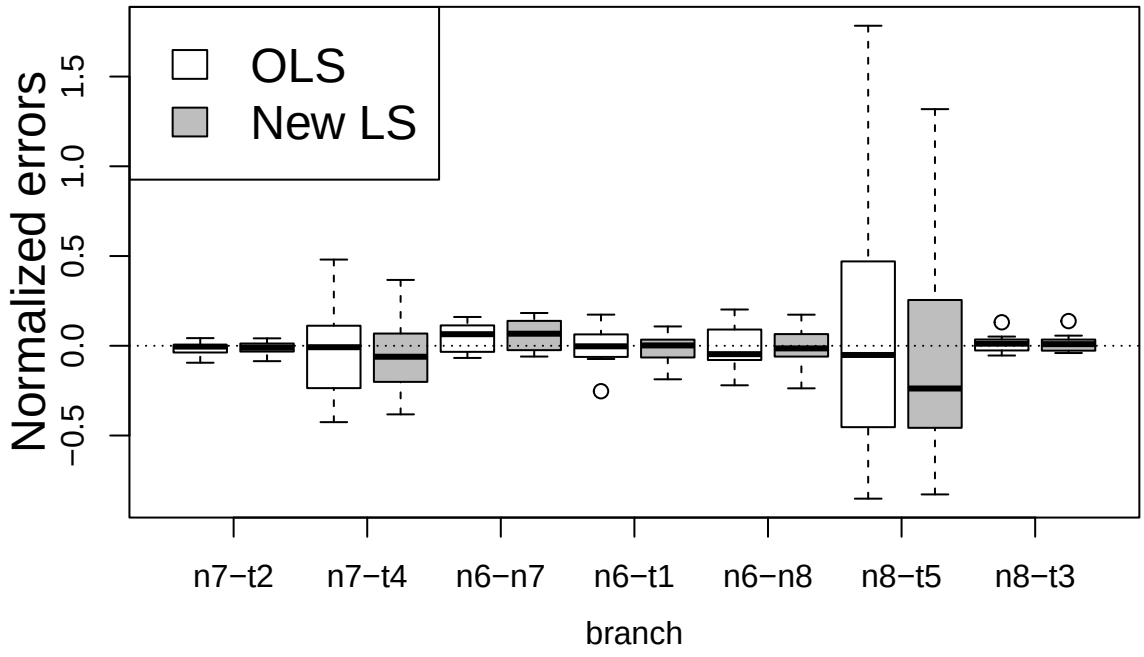
In the manuscript, we simulate nucleotide sequences with a length of 2000 independent sites:

```
for(i in 1:reps){
  # Simulate sequence alignment
  my.align <- as.character(simSeq(my.tree, 2000))
  data.bin<-as.DNAbin(as.alignment(my.align))

  # Run ordinary least squares
  ols.tree <- nnls.tree(as.matrix(dist.dna(data.bin,model="JC69")),unroot(my.tree),trace=0)

  # Extract branch lengths from ordinary least squares
  # Re-order to match output from our routine (below)
  for(j in 1:n.br){
    edge<-unroot(my.tree)$edge[j,]
    ind<-which(apply(apply(ols.tree$edge,1,`==`,edge),2,sum)==2)
    reg.brlen[i,j]<-ols.tree$edge.length[ind]
  }

  # Run branch length estimation with new loss function
  optim.out <- new.ls.fit.optimx(unroot(my.tree), as.phyDat(my.align), rep(0.1, n.br))
  new.brlen[i,] <- optim.out$par.est
}
```

At this point, the simulations are complete. The following code is all for recreating the side-by-side boxplots shown in the figure, from the simulation output.

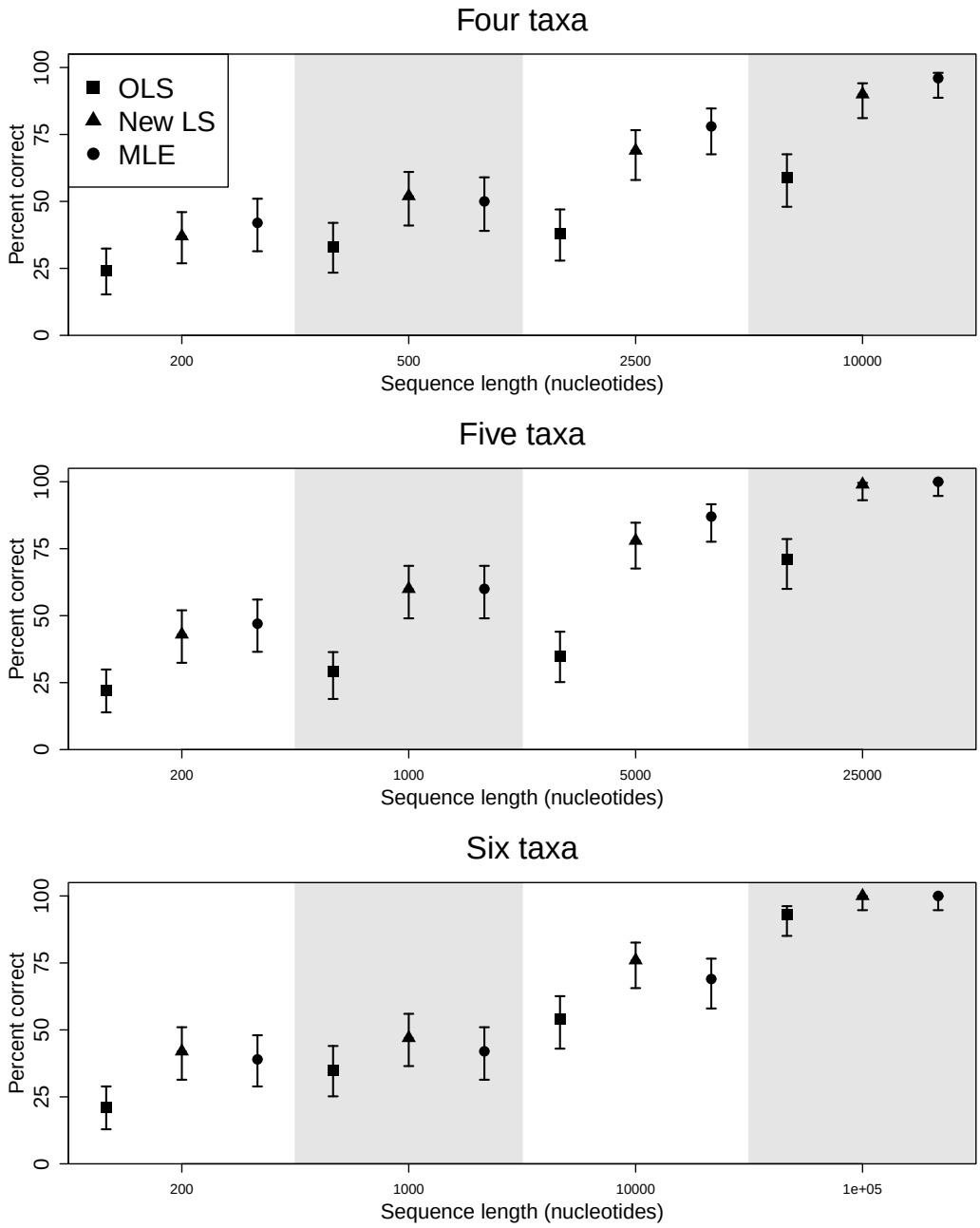
```
# Extract branch names
br.names<-rep(NA,n.br)
n.tips<-length(my.tree$tip.label)

for(k in 1:n.br){
  nodes<-unroot(my.tree)$edge[k,]
  first<-ifelse(nodes[1]<=n.tips,my.tree$tip.label[nodes[1]],paste("n",nodes[1],sep=""))
  second<-ifelse(nodes[2]<=n.tips,my.tree$tip.label[nodes[2]],paste("n",nodes[2],sep=""))
  br.names[k]<-paste(first,second,sep="-")
}


# Calculate normalized errors, order them side-by-side for figure
true<-unroot(my.tree)$edge.length
```

```
all.errors<-matrix(NA,nrow=reps,ncol=14)

for(i in 1:reps){
  all.errors[i,c(1,3,5,7,9,11,13)]<-(reg.brlen[i,]-true)/true
  all.errors[i,c(2,4,6,8,10,12,14)]<-(new.brlen[i,]-true)/true
}

# reorder them to match manuscript figure
br.names<-br.names[c(2,3,1,7,4,6,5)]
errors.reord<-as.data.frame(all.errors[,c(3,4,5,6,1,2,13,14,7,8,11,12,9,10)])


boxplot(errors.reord, col=c("white", "gray"),xlab="",ylab="",xaxt="n")
# legend("topleft",c(expression(L[1]),expression(L[2])),fill=c("white","gray"))
legend("topleft",c("OLS", "New LS"),fill=c("white","gray"), cex=1.5)
axis(1,line=0,at=c(1.5,3.5,5.5,7.5,9.5,11.5,13.5),br.names)
mtext("Normalized errors",side=2,line=2, cex=1.5)
abline(h=0,lty=3)
mtext("branch",side=1,line=2.5)
```



## Figure 4: Accuracy of Topology Estimation

In the manuscript, Figure 4 is the following:




The code to run the simulations for this figure is included in the files called `Fig_topol_4taxa.R`, `Fig_topol_5taxa.R` and `Fig_topol_6taxa.R`, all in the `analysis` directory of our github repository (`https://github.com/peterbchi/phyloLSnoDist`). We walk through the 4-taxa simulations below.

First we set the number of reps and create some items for storage of the results:

```
reps<-10 # This can be bumped up; 10000 reps will take about 11 hours

# Number of nucleotide sites to consider
```

```
n.sites<-c(200,500,2500,10000)

# Create storage vectors
n.scen <- length(n.sites)
correct.tp.ls<-rep(0,n.scen)
correct.tp.nodist<-rep(0,n.scen)
correct.tp.ml<-rep(0,n.scen)
```

Then we load the tree used for simulations:

```
my.tree<-unroot(read.tree("sim_phylo4-6.tree"))
```

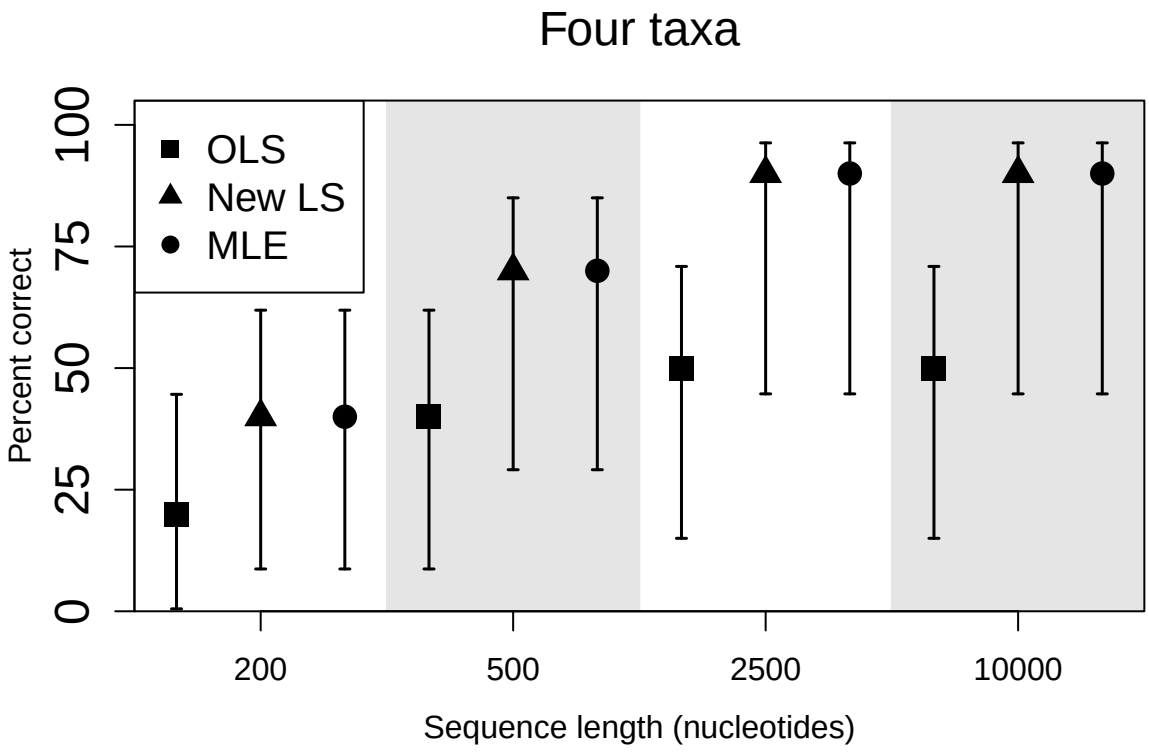
Finally we run simulations through each sequence length:

```
for(s in 1:length(n.sites)){
  for(i in 1:reps){
    my.align <- simSeq(x = my.tree, l = n.sites[s])

    ols.tree <- phylo.ls(my.align, search.all=TRUE)
    nodist.tree <- phylo.ls.nodist(my.align, search.all=TRUE)
    ml.tree <- phylo.ML(my.align, search.all=TRUE)

    correct.tp.ls[s] <- ifelse(all.equal(my.tree, ols.tree, use.edge.length = F),
                               correct.tp.ls[s]+1, correct.tp.ls[s])
    correct.tp.nodist[s] <- ifelse(all.equal(my.tree, nodist.tree, use.edge.length = F),
                                   correct.tp.nodist[s]+1, correct.tp.nodist[s])
    correct.tp.ml[s] <- ifelse(all.equal(my.tree, ml.tree, use.edge.length = F),
                               correct.tp.ml[s]+1, correct.tp.ml[s])
  }
}
```

The code to create the figure from these results is `Fig_topol.R`. An abbreviated version is below:

```
# Construct LCO confidence intervals (Schilling and Doi 2014)
source("LCO_generator_all_n.R")
lco.bounds <- LCO.CI(reps,0.95,3)

# Put results into a single vector for figure
results.4<-c(correct.tp.ls[1],correct.tp.nodist[1],correct.tp.ml[1],
             correct.tp.ls[2],correct.tp.nodist[2],correct.tp.ml[2],
             correct.tp.ls[3],correct.tp.nodist[3],correct.tp.ml[3],
             correct.tp.ls[4],correct.tp.nodist[4],correct.tp.ml[4])


res4.box<-boxplot(t(as.matrix(results.4/reps)),plot=F)
res4.box$stats <- sapply(res4.box$stats[1,],function(x) c(lco.bounds[(x*reps),2],x,x,x,lco.bounds[(x*re



plot.new()
plot.window(xlim=c(0.5,12.5),ylim=c(0,1.05),xaxs="i",yaxs="i")
rect(3.5,-1,6.5,1.05,border="gray90",col="gray90")
```

```
rect(9.5,-1,12.5,1.05,border="gray90",col="gray90")
bxp(res4.box,outline=F,medpch=rep(c(15,17,16),4),medcex=1.7,medlty="blank",whisklty=1,whisklwd=1.5,stapl
axis(2,at=seq(0,1,by=0.25),label=c(0,25,50,75,100), cex.axis=1.5)
axis(1,at=seq(2,13,by=3),label=n.sites)
mtext("Percent correct",line=2.5,side=2)
mtext("Sequence length (nucleotides)",side=1,line=2.5)
mtext("Four taxa",cex=1.5,line=1.25)
legend("topleft",pch=c(15,17,16),c("OLS", "New LS","MLE"), cex=1.25)
box()
```




For all analyses in the manuscript not demonstrated above, the full code are provided in the github repository within the `manuscript` directory:

- Figure 3: `Fig_min3_5taxa.R`, `Fig_min3_6taxa.R` and `Fig_min3_7taxa.R` run the simulations, then `Fig_min3.R` within the `figures` subdirectory makes Figure 3, with file name `Fig_min3.pdf`.
- Figure 5: `Fig_kappa_4taxa.R` and `Fig_kappa_5taxa.R` run the simulations, then `Fig_kappa.R` within the `figures` subdirectory makes Figure 5, with file name `Fig_kappa.pdf`.
- Figure 6: `Belgium2014_newBS.R` runs the analysis along with bootstrapping, and creates the figure with file name `Fig_env_2014.pdf`.