Benchmarking APE's ability to handle large DNA alignments *

Samuel D. J. Brown

November 9, 2011

1 Introduction

This note presents the results of a benchmarking test to determine the capacity of APE [2] to handle large datasets.

This note is also available as a Sweave (.Rnw) file from the SPIDER website (http://spider.r-forge.r-project.org/), which allows these tests to be replicated on computing platforms other than that on which it is performed here. To do this, download the document, open up R, install SPIDER, APEand their required packages, set the working directory and run the command `Sweave("benchmarking.Rnw")`. Pass the resulting .tex file through LaTeXto get the finished product.

For more information about Sweave and reproducible research, visit the Sweave web page (http://www.stat.uni-muenchen.de/ leisch/Sweave/) and the ReproducibleResearch.net webpage.

2 Methods

The tests performed in this document involve recording the time taken to create a K2P distance matrix and neighbour joining tree using the APE command `nj(dist.dna(x))`. The effect of sequence length and sequence number are investigated independantly by manipulating a single DNA alignment. Tests are repeated 50 times and the mean and standard deviation is recorded of the elapsed system time by the samples.

The DNA alignment that will be used is an alignment of 37 sequences of the mitochondrial protein-coding gene cytochrome oxidase I from the 4 New Zealand species of the nursery-web spider genus *Dolomedes* (Pisauridae) [3]. These sequences are available on GenBank as accession numbers GQ337328 through GQ337385 and are pre-loaded as a `DNAbin` object in SPIDER as the `dolomedes` dataset.

*This note was supplied as supplementary data for [1].

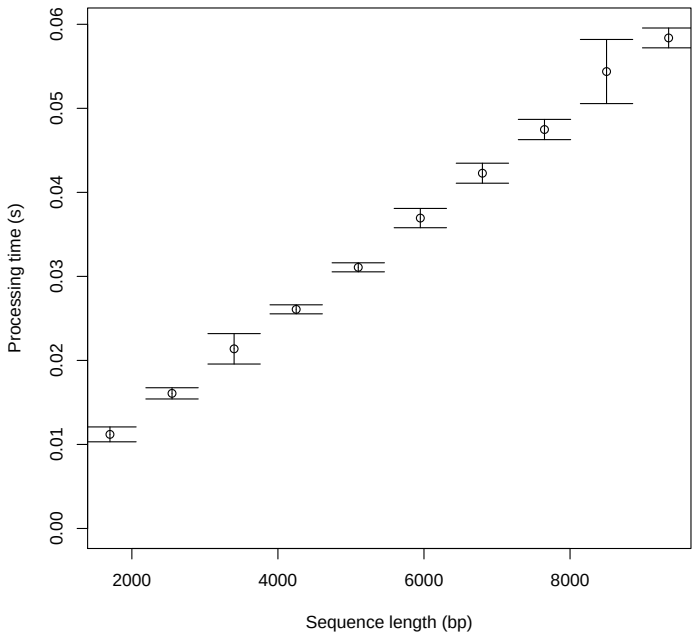
Please note that the .Rnw file takes a while to run to completion. The time taken to process the file is roughly around 10 min, so go and get a cup of tea while the computer thinks about it.

3 Benchmarking tests

```
> library(spider)
> #Benchmarking
> data(dolomedes)
> LenDat <- dolomedes
> #Sequence length
> LenOutput <- matrix(NA, ncol = 4, nrow = 10, dimnames = list(NULL, c("Number", "Length", "
> for(i in 1:10){
+         LenDat <- cbind(LenDat, dolomedes)
+         subput <- list()
+         length(subput) <- 50
+         for(j in 1:50){
+                 subput[[j]][1] <- dim(LenDat)[1]
+                 subput[[j]][2] <- dim(LenDat)[2]
+                 subput[[j]][3] <- system.time(nj(dist.dna(LenDat)))[3]
+         }
+         LenOutput[i,1] <- mean(sapply(subput, function(x) x[1]))
+         LenOutput[i,2] <- mean(sapply(subput, function(x) x[2]))
+         LenOutput[i,3] <- mean(sapply(subput, function(x) x[3]))
+         LenOutput[i,4] <- sd(sapply(subput, function(x) x[3]))
+
+ }
> NumDat <- dolomedes
> #Sequence number
> NumOutput <- matrix(NA, ncol = 4, nrow = 10, dimnames = list(NULL, c("Number", "Length", "
> for(i in 1:10){
+         NumDat <- rbind(NumDat, dolomedes)
+         subput <- list()
+         length(subput) <- 50
+         for(j in 1:50){
+                 subput[[j]][1] <- dim(NumDat)[1]
+                 subput[[j]][2] <- dim(NumDat)[2]
+                 subput[[j]][3] <- system.time(nj(dist.dna(NumDat)))[3]
+         }
+         NumOutput[i,1] <- mean(sapply(subput, function(x) x[1]))
+         NumOutput[i,2] <- mean(sapply(subput, function(x) x[2]))
+         NumOutput[i,3] <- mean(sapply(subput, function(x) x[3]))
+         NumOutput[i,4] <- sd(sapply(subput, function(x) x[3]))
+ }
```

4 Effect of sequence length

```
> LenNum <- max(LenOutput[,1])
> LenMax <- max(LenOutput[,2])
> LenMaxTime <- max(mapply(x = LenOutput[,3], y = 1.96 * LenOutput[,4], sum))
> plot(LenOutput[,2], LenOutput[,3], xlab = "Sequence length (bp)", ylab = "Processing time
> arrows(x0 = LenOutput[,2], y0 = LenOutput[,3], y1 = LenOutput[,3] - 1.96 * LenOutput[,4],
> arrows(x0 = LenOutput[,2], y0 = LenOutput[,3], y1 = LenOutput[,3] + 1.96 * LenOutput[,4],
```



From this graph we can see that the processing time increases linearly with sequence length. Processing time is very fast however. In this instance, with 37 sequences and a length of 9350, in 97.5% of instances, the processing time will be less than 0.0595606574439694 seconds.

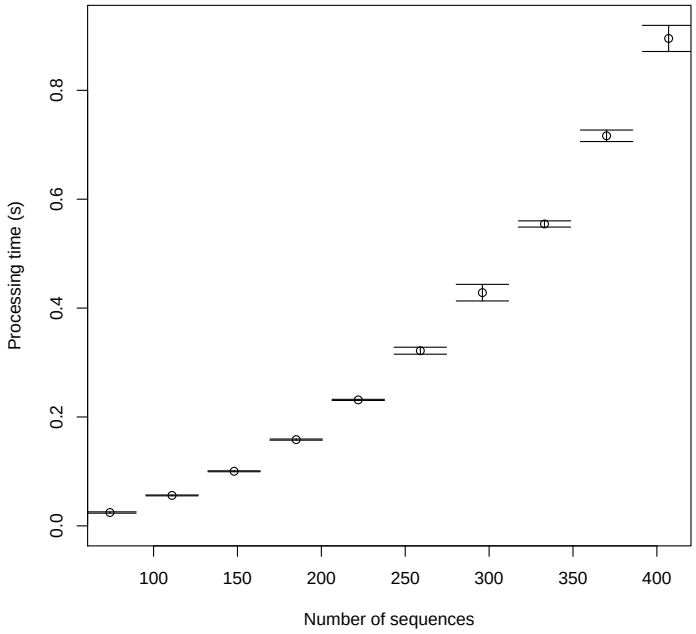
```
> LenOutput
```

```
      Number Length Time mean      Time SD
 [1,]     37   1700   0.01120 0.0004517540
 [2,]     37   2550   0.01608 0.0003404679
 [3,]     37   3400   0.02138 0.0009233921
 [4,]     37   4250   0.02608 0.0002740475
 [5,]     37   5100   0.03108 0.0002740475
 [6,]     37   5950   0.03694 0.0005858885
```

```
 [7,]     37   6800   0.04228 0.0006074369
 [8,]     37   7650   0.04748 0.0006141196
 [9,]     37   8500   0.05438 0.0019471591
[10,]     37   9350   0.05838 0.0006023762
```

5 Effect of sequence number

```
> NumLen <- max(NumOutput[,2])
> NumMax <- max(NumOutput[,1])
> NumMaxTime <- max(mapply(x = NumOutput[,3], y = 1.96 * NumOutput[,4], sum))
> plot(NumOutput[,1], NumOutput[,3], xlab = "Number of sequences", ylab = "Processing time
> arrows(x0 = NumOutput[,1], y0 = NumOutput[,3], y1 = NumOutput[,3] - 1.96 * NumOutput[,4],
> arrows(x0 = NumOutput[,1], y0 = NumOutput[,3], y1 = NumOutput[,3] + 1.96 * NumOutput[,4],
```



From this graph we can see that the processing time increases exponentially as the number of sequences in the alignment increases. In this instance, with 407 sequences and a length of 850, in 97.5% of instances, the processing time will be less than 0.919310101958933 seconds.

```
> NumOutput
```

```
      Number Length Time mean      Time SD
 [1,]     74    850   0.02452 0.0006141196
```

```
 [2,]    111    850   0.05592 0.0003958973
 [3,]    148    850   0.10024 0.0004314191
 [4,]    185    850   0.15844 0.0005405968
 [5,]    222    850   0.23138 0.0005303060
 [6,]    259    850   0.32172 0.0032703991
 [7,]    296    850   0.42834 0.0077764952
 [8,]    333    850   0.55454 0.0029081869
 [9,]    370    850   0.71648 0.0053955386
[10,]    407    850   0.89532 0.0122398479
```

6 Large datasets

```
> bigDat <- dolomedes[sample(1:dim(dolomedes)[1], 3000, replace=TRUE),]
> bigDistRes <- system.time(dist.dna(bigDat))
> bigTreeRes <- system.time(nj(dist.dna(bigDat)))
>
```

This dataset represents one that might be obtained from an environmental DNA project using next-generation sequencing technology. Creating the distance matrix is fairly rapid, taking around 49.296 seconds. Building the tree is a much more intensive job, the whole process from alignment through distance matrix to NJ tree taking around 221.647 seconds.

Testing larger datasets is left as an exercise for the user ;)

7 System information

The following is the information of the system this file has been run on, for purposes of comparison between machines.

```
> sessionInfo()

R version 2.14.0 (2011-10-31)
Platform: i486-pc-linux-gnu (32-bit)

locale:
 [1] LC_CTYPE=en_NZ.utf8       LC_NUMERIC=C
 [3] LC_TIME=en_NZ.utf8        LC_COLLATE=en_NZ.utf8
 [5] LC_MONETARY=en_NZ.utf8    LC_MESSAGES=en_NZ.utf8
 [7] LC_PAPER=C                LC_NAME=C
 [9] LC_ADDRESS=C              LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_NZ.utf8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
```

```
other attached packages:
[1] spider_1.1-0   pegas_0.4      adegenet_1.3-1 ade4_1.4-17    MASS_7.3-16
[6] ape_2.8

loaded via a namespace (and not attached):
[1] gee_4.13-17    grid_2.14.0    lattice_0.20-0 nlme_3.1-102   tools_2.14.0
```

8 Conclusion

Although it is an interpreted language, the time APE takes for basic DNA sequence manipulation is generally within the realms of acceptability. Using the process from alignment to distance matrix to neighbour-joining tree as a standard, the time taken increases linearly with increasing sequence length, but exponentially as the number of sequences in the alignment increases.

References

[1] Samuel David James Brown, Rupert A Collins, Stephane Boyer, Marie-Caroline Lefort, Jagoba Malumbres-Olarte, Cor J Vink, and Robert H Cruickshank. SPIDER: An R package for the analysis of species identity and evolution, with particular reference to DNA barcoding. *Molecular Ecology Resources*, In press.

[2] E. Paradis, J. Claude, and K. Strimmer. APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, 20:289–290, 2004.

[3] Cor J Vink and N Dupérré. Pisauridae (Arachnida: Araneae). *Fauna of New Zealand*, 64:1–54, 2010.