Tuatara rates analyses

David Winter

08/11/2016

Outline

This working directory contains an example R project. A complete copy of all of the data in this repo is available from a public google drive: https://drive.google.com/open?id=0B-AfkPuOyhRaNHBBd3gzb1EzNkE

We are trying to

- Estimate substution rates across the reptile phylogeny
- Compare rates among the major clades
- Identify genetic sequences with unusually high conservation (if we have time)

To do that we have prepare at 14 spp reptile alignment. The process by which these alignments were produced is described in `~/tutes/msa`. The files included here are

- `alignments/` a series of .maf alignments, one per chromosome in the Anolis (AnoCar2) genome
- `annotation/` annotatoin files including `anoCar2_ens_genes.gtf` retrieved from the ftp://ftp.ensembl.org/pub/release-85/gtf/anolis_carolinensis
- `reps.tr` a reptile tree retrieved from Open Tree of Life
- `contraints.csv` a file containing approximate dates for the splits in `reps.tr`

Extract 4-fold degenerate sites

Step one: work out how to do it once

Our first challange was to identify the fourfold degenerate sites from these alignments. We read the vignette for `rphast` to see a similar analysis and adapted that workflow to our data.

In particular we started by working out how to analyse one chromosome's worth of data. We started by reading in a chromosome and the annotation data:

```
library(rphast)
chrom <- read.msa("alignments/chrLGa.maf")
genes <- read.feat("annotation/anoCar2_ens_genes.gtf")
chrom
```

```
## msa object with 19 sequences and 7101755 columns, stored in R
## $names
##  [1] "anoCar2" "pogVit1" "ophGra1" "anaPla1" "galGal3" "melGal2" "melUnd1"
##  [8] "taeGut3" "ophHan1" "croMit1" "sphPun"  "pytBiv1" "pelSin1" "strCam1"
## [15] "geoFor1" "allMis1" "chrPic1" "gavGan1" "croPor1"
##
## $alphabet
## [1] "ACGT"
##
## $is.ordered
## [1] TRUE
```

```
##
## $offset
## [1] 10429
##
## (alignment output suppressed)
```

If we try to extract the fourfold degenerate sites we hit a stumbling block.

```
get4d.msa(chrom, genes)
```

```
## Error in .Call.rphast("rph_msa_reduce_to_4d", x$externalPtr, features$externalPtr): to obtain 4d sit
```

Though this error message is a little opaque, we were able to work out that we had to (a) extract only those features from the chromosome `chrLGa` in `genes` and (b) update the chromosome names to match the chromosome names in the alignment:

```
LGa_genes <- genes[genes$seqname == "LGa",]
LGa_genes$seqname <- paste0("anoCar2.chr", LGa_genes$seqname)
ffd <- get4d.msa(chrom, LGa_genes)
ffd
```

```
## msa object with 19 sequences and 2947 columns, stored in R
## $names
##  [1] "anoCar2" "pogVit1" "ophGra1" "anaPla1" "galGal3" "melGal2" "melUnd1"
##  [8] "taeGut3" "ophHan1" "croMit1" "sphPun"  "pytBiv1" "pelSin1" "strCam1"
## [15] "geoFor1" "allMis1" "chrPic1" "gavGan1" "croPor1"
##
## $alphabet
## [1] "ACGT"
##
## $is.ordered
## [1] FALSE
##
## (alignment output suppressed)
```

Step two: do that hundreds of times

So we worked out how to do that once, but there are hundreds of chromosomes in the real dataset. One key step to being able to do reproducible research in computational biology is being able to write functions that "abstract out" the core analysis in a pipeline.

In this case we want to think about the sorts of inputs we will have (a chromosome file and a set of features) and the outputs we want to produce (a set of fourfold degenerate sites). Taking that on board we can write a function that takes a chromosome file, identifies the chromosome name, and extracts correct features form the annotations in order to get the degenerate sites.

```
get_ffd <- function(fname, features){
  chrom <- read.msa(fname)
  maf_file <- sub(pattern="alignments/", replacement="", fname)
  chrom_name <- sub(".maf", "", maf_file)
  annotation_name <- paste0("anoCar2.", chrom_name)
  #message(annotation_name)
```

```
  gene_annotation <- features[features$seqname == annotation_name,]
  read.msa(fname, features=gene_annotation)
}
```

We can then use `lapply` to generate a list with the degenerate sites from each chromosome. We can then stick all of the sites into one object usgin `concat.msa`. (Because this analysis computational intensive it is *not* executed as part of this markdown document)

```
fnames <- list.files("alignments")
all_the_ffd_sites <- lapply(fnames, get_ffd, features=genes)
ffd_all <- concat.msa(all_the_ffd_sites)
```

Estimate rates on the tree

For our next trick, we want to estimate rates of evolution on the reptile tree. If you read the vignette you will see `phyloFit` is the function for this. To make this step a little quicker I've included a saved model that is very close to maximum likelihood estiamte for this data, using this as the `init.mod` (i.e. the starting values for the search) should speed things up, but it will still take a long time (the

```
library(ape)
```

```
##
## Attaching package: 'ape'
```

```
## The following object is masked from 'package:rphast':
##
##     complement
```

```
load("starter_model.Rds")
neutral_mod <- phyloFit(msa=ffd, init.mod=starter_mod)
```

Compare rates among groups

Make a pretty plot

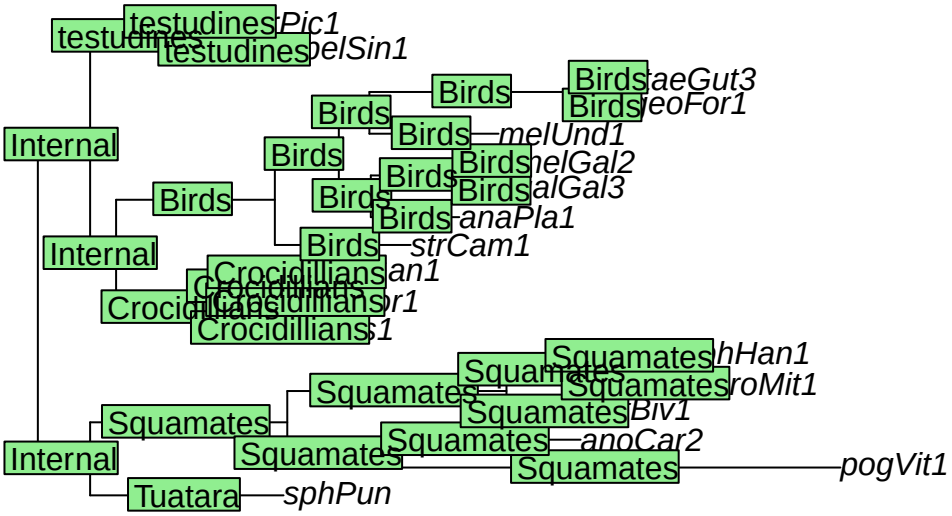
Our goal here was to compare rates of evolution among the major groups in the reptile tree. To do that we need to read the tree as an `R` object (with `read.tree` and attach group labels edges:

```
library(ape)
```

```
##
## Attaching package: 'ape'
```

```
## The following object is masked from 'package:rphast':
##
##     complement
```

```
tr <- ape::read.tree(text=neutral_mod$tree)
edge_grps <- scan("tree_data/edge_groups.txt", "\n")
plot(tr)
edgelabels(edge_grps)
```



At the moment the length of the branches in the tree are in expected number substitutions per site. We are interested in the *rate* of evolution, so need to put some dates on some of the nodes in the tree. Some rough estimates of those dates are in the file `tree_data/contraints.csv` (note how `kable` produces a nice table output in the compiled document!):

```
cons <- read.csv("tree_data/constraints.csv")
knitr::kable(tail(cons))
```

	node	age.min	age.max	soft.bounds	spp
12	30	85.0	95.0	FALSE	anaPla1-galGal3-melGal2-melUnd1-geoFor1-taeGut3
13	31	55.0	75.0	FALSE	anaPla1-galGal3-melGal2
14	32	19.9	49.7	FALSE	galGal3-melGal2
15	33	50.0	60.0	FALSE	melUnd1-geoFor1-taeGut3
16	34	15.0	35.0	FALSE	geoFor1-taeGut3
17	35	110.0	115.0	FALSE	pelSin1-chrPic1

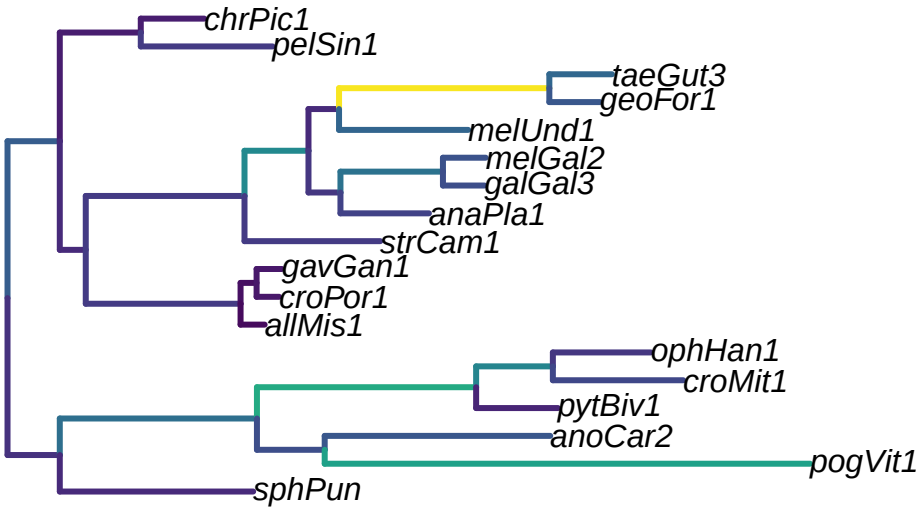
... and the `ape` function `chronos` lets us fit "smoothed" rates of evolution to the branches on the phylogeny

```
smoothed_tr <- chronos(tr, calibration=cons, lambda=1)
```

```
##
## Setting initial dates...
## Fitting in progress... get a first set of estimates
##          Penalised log-lik = -9.28541
## Optimising rates... dates... -9.28541
##
## Done.
```

Making a nice plot of these tree requires some helper functions (ask me if you want to know how these work).

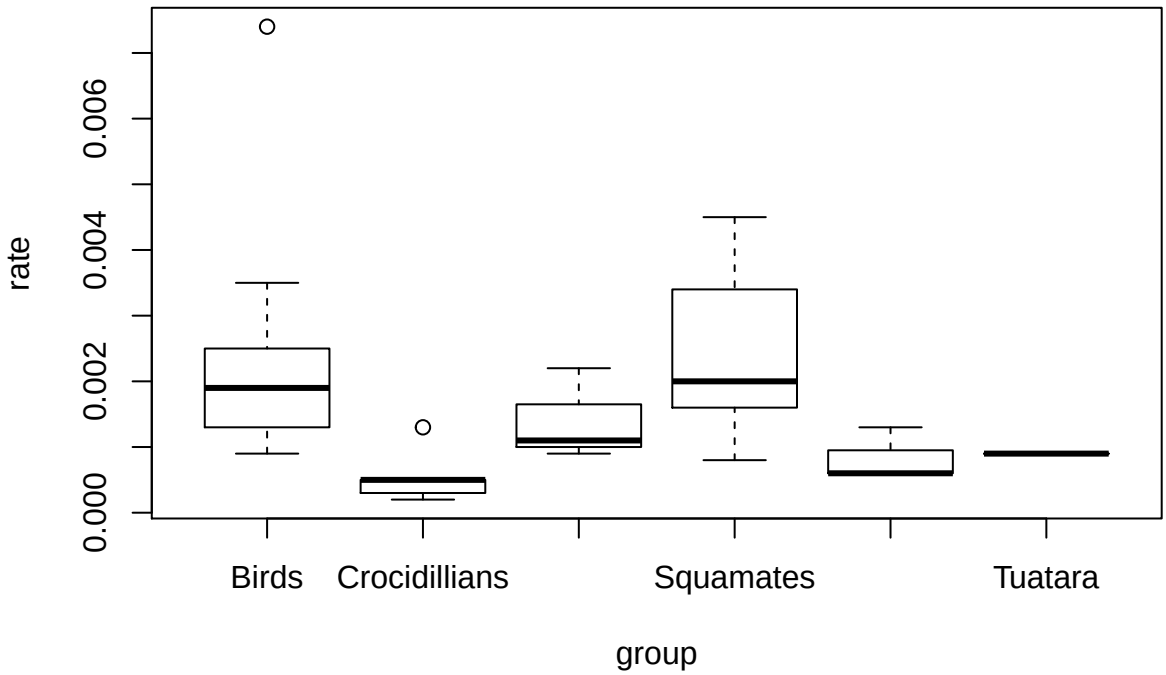
```
library(viridis)

get_rates <- function(x, digits=4)  round(attr(x, "rates"), digits)

make_palette <- function(R){
 cuts <- as.numeric(cut(R, seq(0, max(R), length.out=128),  include.lowest=TRUE)  )
}

plot_rate_tree <- function(smoothed, pal=viridis(128)){
 R <- get_rates(smoothed)
 cuts <- make_palette(R)
 plot(tr, edge.color=pal[cuts], edge.width=3)
}

plot_rate_tree(smoothed_tr)
```



Compare groups

The tree allows us to compare rates as the move down through time. It's also helpful to think about among-clade differences. We can produce a box plot using the rates and "edge group" data. Note in R the tilde (~) is usually used to set up a formula, you can think of it as saying "responds to". So in this case we are saying the rate of evolution responds to the group in which an edge lives).

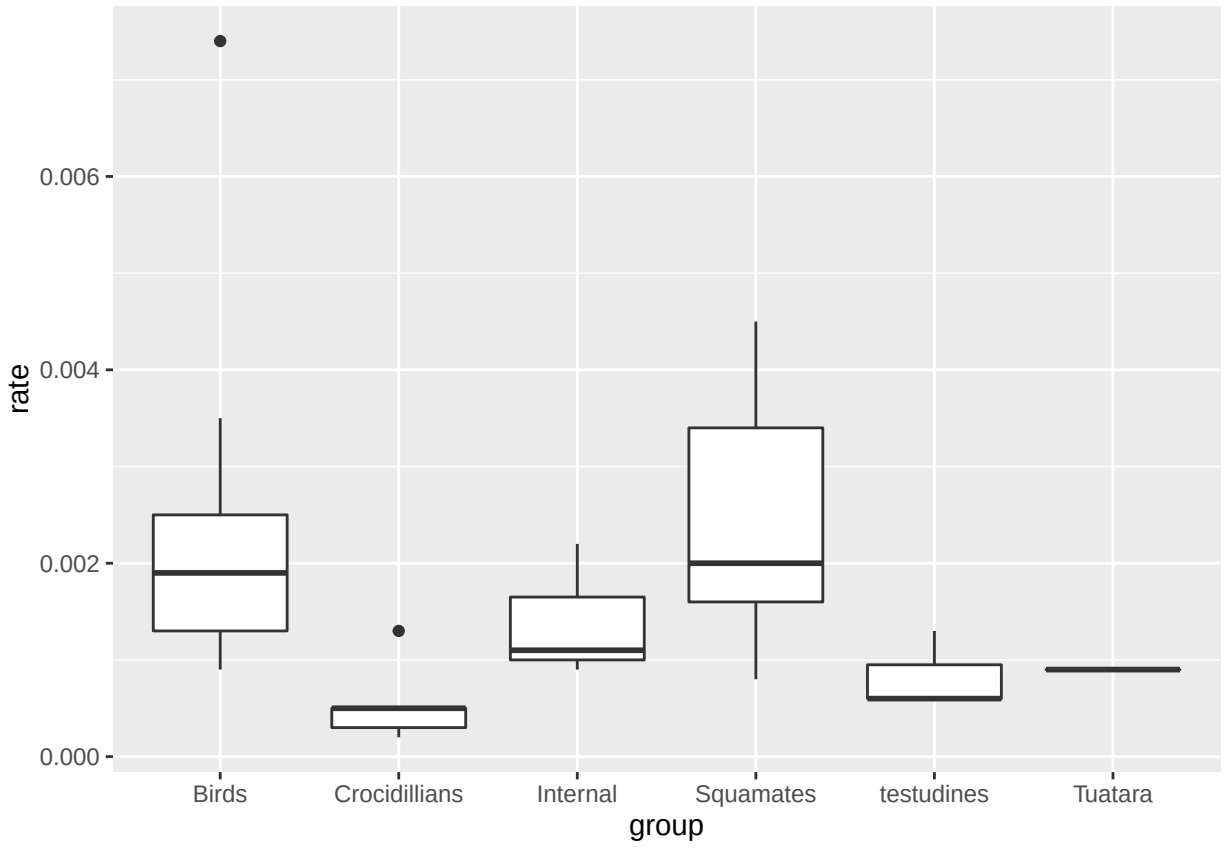
```
rate_data <- data.frame(group=edge_grps, rate=get_rates(smoothed_tr))
plot(rate ~ group, data=rate_data)
```



If you don't like the "base" graphics, there is always `ggplot`. The `if` statement here is designed to only try and make this plot if you have `ggplot2` installed, try `install.packages("ggplot2")` if you get a warning and not plot.

```
if( require(ggplot2) ){
    ggplot(rate_data, aes(group, rate)) + geom_boxplot()
}
```

```
## Loading required package: ggplot2
```


rate
0.006
0.004
0.002
0.000
Birds
Crocidillians
Internal
Squamates
testudines
Tuatara
group