Analysing diversification with diversitree

Rich FitzJohn

September 23, 2010

Contents

1 Introduction

The diversitree package includes a number of comparative phylogenetic methods, mostly focusing on analysing diversification and character evolution. These methods all share a common set of utility tools for maximum likelihood (ML) parameter estimation, hypothesis testing and model comparison, and Bayesian parameter estimation through Markov chain Monte Carlo (MCMC). Included methods include:

- **Diversification**
  - Constant rate birth-death models (Nee et al., 1994)
- **Character evolution**
  - Discrete trait evolution (Pagel, 1994)
  - Univariate Brownian motion parameter estimation for continuous characters
- **Joint character evolution and diversification**

  - BiSSE: Binary trait speciation and extinction (Maddison et al., 2007), and extensions for incompletely resolved phylogenies (FitzJohn et al., 2009).
  - MuSSE: Multiple State Speciation and Extinction.

In addition, variants of these models are available:

- "Time dependent": different time epochs have different parameters (implemented for BiSSE, MuSSE).
- MEDUSA-style partitioned analyses, where different regions of the tree have different parameters (implemented for birth-death, BiSSE, and MuSSE).
- Marginal ancestral state reconstruction for discrete characters ("Pagel94") and BiSSE.
- Stochastic character mapping for discrete traits (Bollback, 2006)

In the future, new methods will include

- QuaSSE: Quantitative State Speciation and Extinction: (FitzJohn, 2010)
- Reflected Brownian motion for bounded traits
- Stochastic character mapping for discrete traits that affect speciation or extinction rates
- Geographic modes of speciation (GeoSSE: Geographic State Speciation and Extinction: Goldberg et al. *submitted*)

For all methods, inference can be carried out under maximum likelihood, or in Bayesian analyses via MCMC (Markov Chain Monte Carlo). Phylogenies can also be simulated under several of the methods.

This tutorial is designed to give an overview of the features in diversitree. It does not aim to be a compete reference to the package, or claim to always follow best practice. The manual follows the structure above. Many of the examples are just taken from the online documentation; further examples can be found there. Most are fairly contrived – if you have examples you would rather see here, I would welcome data sets.

```
> library(diversitree)
```

2 Constant-rate birth-death models

The ape package already has some support for constant-rate birth-death models, but diversitree duplicates this for completeness. The major differences are (1) the function is not constrained to positive diversification rates (μ can exceed λ), (2) support for both random taxon sampling and unresolved terminal clades (but see ape's `bd.ext`), and (3) run both MCMC and MLE fits to birth death trees. The constant rate birth death model is a special case of the other diversification models implemented in the package, and is the simplest model in diversitree.

Start with a simulated phylogeny, with speciation rate $\lambda = 0.1$ and extinction rate $\mu = 0.03$:

```
> set.seed(2)
> phy <- tree.bd(c(0.1, 0.03), max.taxa = 100)
```

(plotted in figure 1). The first step in any analysis in diversitree is to construct a likelihood function. For constant rate birth death models, this is done with `make.bd`:

```
> lik <- make.bd(phy)
```

To see the argument names of the likelihood function, use the `argnames` function

```
> argnames(lik)
```

```
[1] "lambda" "mu"
```

This shows that `lik` takes a vector of two parameters (λ, μ). It will return the log likelihood of the parameters, following the calculations in (Nee et al., 1994).

```
> lik(c(0.1, 0.03))
```

```
[1] -7.74086
```

Most likelihood functions accept additional arguments; these are documented on their online help pages. The only additional argument accepted for birth-death model is to disable conditioning on survival (by default, the likelihood is conditional on two lineages surviving to the present, following (Nee et al., 1994).

```
> lik(c(0.1, 0.03), condition.surv = FALSE)
```

```
[1] -10.74823
```

To do a ML model fit, pass the likelihood function and a starting point guess to the `find.mle` function:

```
> fit <- find.mle(lik, c(0.1, 0.03), method = "subplex")
```

The final argument here selects the method "`subplex`" for the ML search; other methods are available. (The default for birth-death models is "`nlm`", which may produce warnings about failure to converge for the example here.)

To extract the coefficients from the fitted object, use the `coef` function:

```
> coef(fit)
```

```
    lambda         mu
0.09782550 0.05671838
```

and to extract the log-likelihood value at the ML point, use the $logLik$ function

```
> logLik(fit)
```

```
'log Lik.' -4.148588 (df=2)
```

which extracts the coefficients with some additional information, or extract the `lnLik` element from the list directly:

```
> fit$lnLik
```

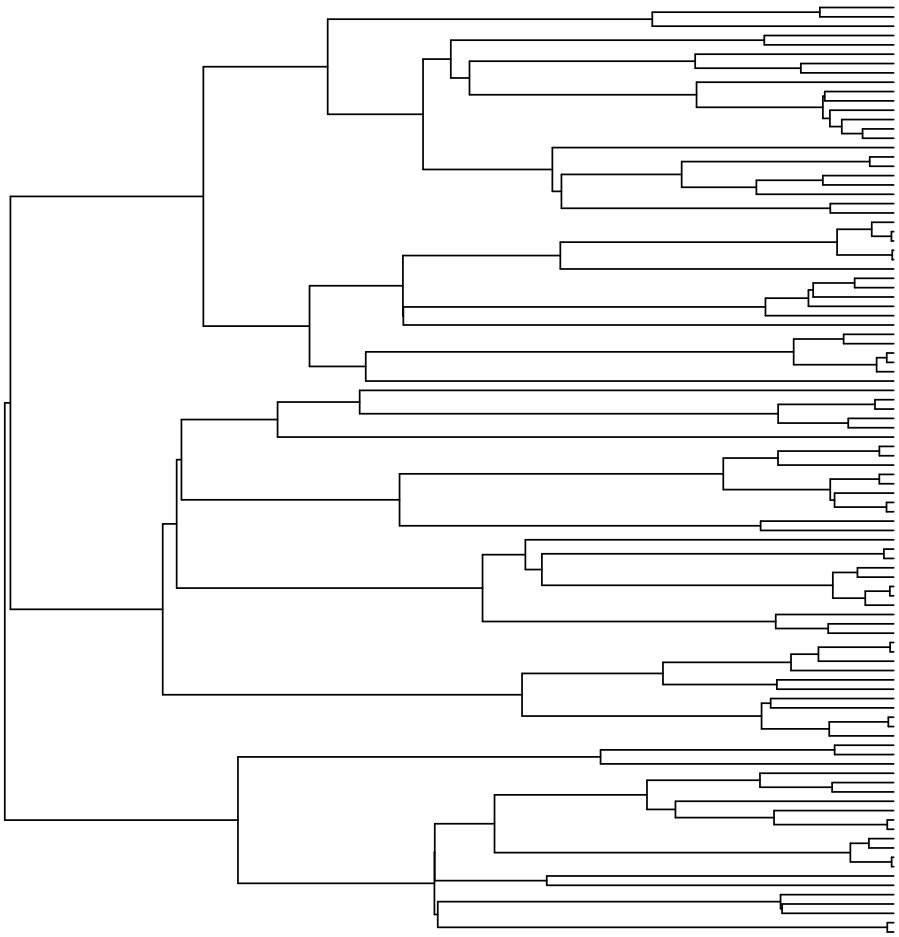

Figure 1: A birth-death tree, with speciation rate $\lambda = 0.1$ and extinction rate $\mu = 0.03$.

```
[1] -4.148588
```

Does this model fit much better than a model without extinction (a Yule, or pure birth, model)? You can constrain parameters of likelihood functions using the `constrain` function. To specify that the extinction rate, μ should be zero:

```
> lik.yule <- constrain(lik, mu ~ 0)
```

Argument names here must match those given by `argnames`. Run the ML search the same way as above, specifying a single starting parameter (I've used the speciation rate from the full model here):

```
> fit.yule <- find.mle(lik.yule, coef(fit)[1], method = "subplex")
```

To perform a likelihood ratio test, use the `anova` function[1] The model with the nonzero extinction estimate is preferred, with $\chi^2_1 = 5.7$.

```
> anova(fit, yule = fit.yule)

     Df   lnLik    AIC  ChiSq Pr(>|Chi|)
full  2 -4.1486 12.297
yule  1 -7.0118 16.024 5.7264    0.01671
```

Alternatively, we can use Markov chain Monte Carlo (MCMC) to perform a Bayesian analysis. Here, I will use a uniform prior on the interval $[0, \infty)$ for both parameters, by not specifying any prior. The `w` parameter is the tuning parameter. Here, it affects how many function evaluations will be needed per sample, but will not generally affect the rate of mixing (see the online help for more information).

```
> samples <- mcmc(lik, fit$par, nsteps = 2000, w = c(0.1, 0.1),
+     lower = 0, upper = Inf, print.every = 1000)

1000: {0.0824, 0.0421} -> -4.64321
2000: {0.0980, 0.0542} -> -4.18160
```

The posterior distribution of these parameters, and the code to generate it, is in figure 2.

Analyses can also use trees where only a fraction of species are present in the phylogeny. To demonstrate this, let's drop 25 of the 100 species from the original tree at random:

```
> set.seed(1)
> phy.sub <- drop.tip(phy, sample(100, 25))
```

When constructing the likelihood function, pass an argument `sampling.f` in, with a value on $(0, 1]$ representing the fraction of species that are descended from the root node that are included in the phylogeny (here, 75/100). Then, run a ML analysis with `find.mle` as before:

```
> lik.sub <- make.bd(phy.sub, sampling.f = 75/100)
> fit.sub <- find.mle(lik.sub, c(0.1, 0.03), method = "subplex")
> coef(fit.sub)
```

[1] This is an unfortunate convention in R: many packages use this function as a general model comparison function, and I've taken their lead here – in a future version, I may add a `lrt` function, which should be clearer. No analysis of variance is performed.

```
> samples$r <- samples$lambda - samples$mu
> col <- c("red", "blue", "green3")
> profiles.plot(samples[c("lambda", "mu", "r")], col.line = col,
+     las = 1, ylab = "Probability density", xlab = "Parameter estimate")
> legend("topright", c("lambda", "mu", "r"), fill = col)
> abline(v = 0, lty = 2)
> abline(v = c(0.1, 0.03, 0.07), col = col)
```

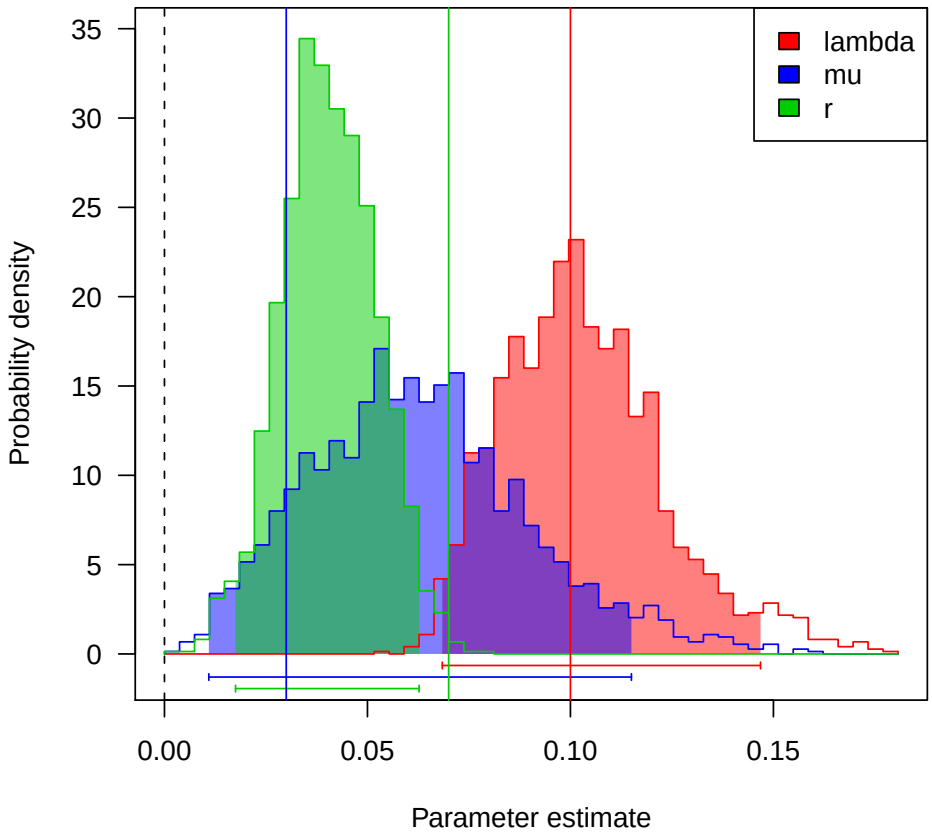

Figure 2: Posterior probability distributions for the parameters of the constant rate birth death model. True values are indicated by the solid vertical lines. The bars at the bottom of the distributions and the shaded areas correspond to the 95% credibility intervals. These include the two parameter λ and μ, though the true diversification rate r lies above the 95% credibility interval for that parameter.

```
    lambda         mu
0.08608870 0.04237603
```

With fewer included species, test to see whether the full model is still preferred over the Yule model:

```
> lik.sub.yule <- constrain(lik.sub, mu ~ 0)
> fit.sub.yule <- find.mle(lik.sub.yule, coef(fit.sub)[1], method = "subplex")
> anova(fit.sub, yule = fit.sub.yule)

     Df   lnLik    AIC  ChiSq Pr(>|Chi|)
full  2 -39.182 82.364
yule  1 -40.501 83.002 2.6388     0.1043
```

Dropping these species has reduced the support from $\chi^2_1 = 5.7$ to $\chi^2_1 = 2.6$, and the difference is no longer significant at the 5% level.

3 Markov models of discrete character evolution

Again, start with a simulated tree and character distribution. This uses the BiSSEsimulation code, explained later, as we need both a tree and character distribution. This is just a 25 species birth-death tree, with fairly high rates of character evolution of a binary character with an asymmetry in the character transition rates, so that 0 → 1 transitions occur less rapidly than 1 → 0 transitions. The root of the tree starts in state 0.

```
> pars <- c(0.1, 0.1, 0.03, 0.03, 0.1, 0.2)
> set.seed(3)
> phy <- trees(pars, "bisse", max.taxa = 25, max.t = Inf, x0 = 0)[[1]]
> states <- phy$tip.state
> states
```

```
 sp3  sp4  sp5  sp6  sp7  sp8 sp10 sp11 sp12 sp13 sp14 sp15 sp16 sp17 sp18 sp19
   0    1    0    0    0    0    0    0    0    1    0    0    0    1    0    0
sp20 sp21 sp22 sp23 sp24 sp25 sp26 sp27 sp28
   0    1    0    0    0    0    0    0    0
```

The simulation remembers the true character history, which is displayed in figure 3.

Next, build a likelihood function with the `make.mk2` function, and run a ML analysis with `find.mle`, using an initial parameter guess of (0.1, 0.1):

```
> lik.mk2 <- make.mk2(phy, states)
> fit.mk2 <- find.mle(lik.mk2, c(0.1, 0.1), method = "subplex")
> coef(fit.mk2)
```

```
      q01       q10
0.1527806 0.7708167
```

In the fit, the q_{10} parameter is much higher than q_{01}; the difference being larger than in the true model.

See if this difference is statistically justified by running a model where the two q values are constrained to be equal:

```
> lik.mk1 <- constrain(lik.mk2, q10 ~ q01)
> fit.mk1 <- find.mle(lik.mk1, 0.1, method = "subplex")
> anova(fit.mk2, mk1 = fit.mk1)
```

```
     Df   lnLik    AIC  ChiSq Pr(>|Chi|)
full  2 -10.906 25.811
mk1   1 -13.025 28.049 4.2381    0.03953
```

This is marginally significant $\chi^2_1 = 4.2$, so the asymmetric model fits better.

```
> par(mar = rep(0, 4))
> plot(history.from.sim.discrete(phy, 0:1), phy)
```

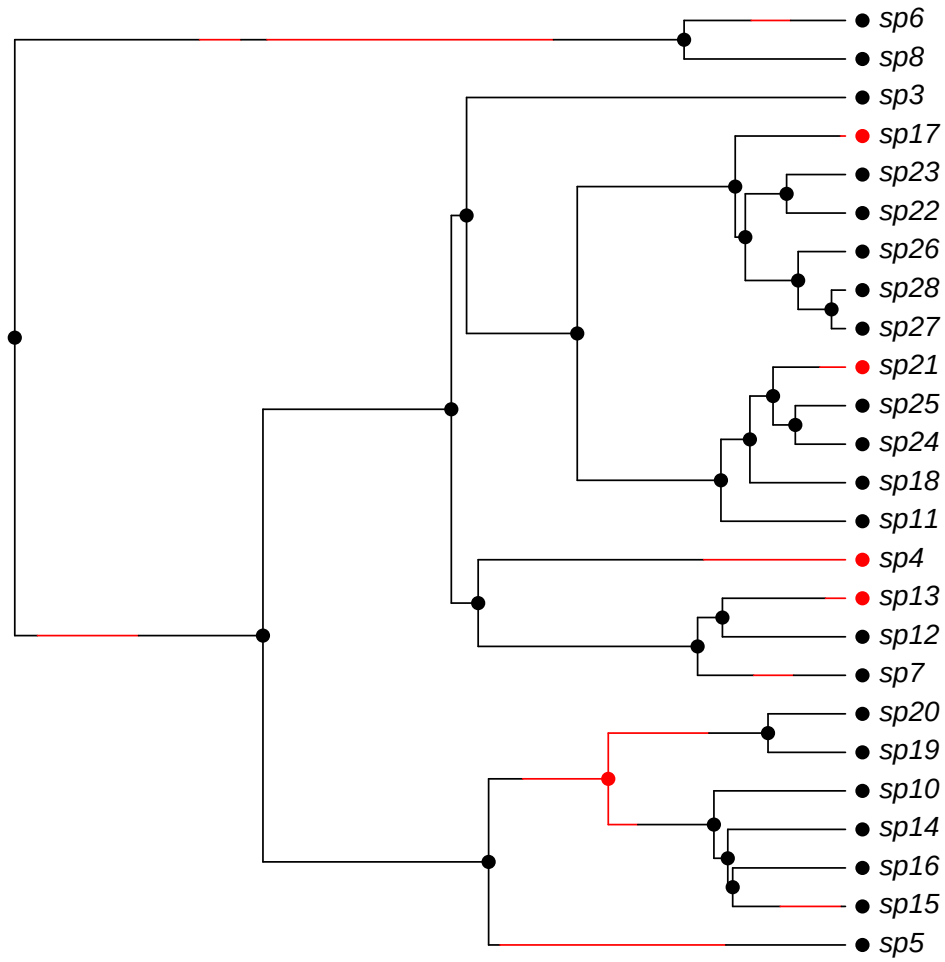

Figure 3: Character history for simulated trait and tree. Black is state 0, red is state 1.

3.1 Drawing samples with MCMC

It is possible to run an MCMC analysis. However, care should be taken to choose priors carefully, as while $q_{10}/(q_{01}+q_{10})$ is usually well characterised by the data, the overall rate $(q_{01}+q_{10})$ is poorly defined. For small trees like this, essentially infinite values of character evolution are consistent with the data, with the tip states just drawn from the stationary distribution of the process.

There are two supplied prior functions, but any function that takes a vector of parameters and returns the log prior probability may be used. First, consider an exponential prior with rate 10, which gives a mean of 1/10, and assume the same prior distribution for both parameters.

```
> prior.exp <- make.prior.exponential(10)
```

To run the MCMC, we need to specify a starting point (again, I have used (.1,.1), but the ML point might be preferable). I have discarded the first 500 samples (10%), which is probably overkill for this model, as the autocorrelation between samples is extremely small.

```
> samples <- mcmc(lik.mk2, c(0.1, 0.1), nsteps = 5000, prior = prior.exp,
+     w = 0.1, lower = 0, print.every = 1000)

1000: {0.0519, 0.2533} -> -9.81083
2000: {0.0360, 0.1206} -> -9.03188
3000: {0.0523, 0.2049} -> -9.45210
4000: {0.0245, 0.0190} -> -8.91244
5000: {0.0478, 0.1906} -> -9.34016

> samples <- subset(samples, i > 500)
```

The marginal distributions of these parameters are shown in figure The two marginal distributions overlap substantially, and there is little support here for the hypothesis that q_{10} is greater than q_{01}.

The other prior included in diversitree assumes that the mean of the rates is exponentially distributed, and the scaled difference between the two parameters is Beta distributed. If we assume that the Beta distribution is symmetrical, then its two parameters are equal, and we have

$$\bar{q} = \frac{q_{01}+q_{10}}{2} \sim \text{Exp}[r]$$
$$\Delta q = 1 - \frac{q_{01}}{2\bar{q}} \sim \text{Beta}[\beta,\beta]$$

Specifying a exponential distribution for $\bar{q}$ with rate 10 (as before), and a uniform distribution for Δq:

```
> prior.eb <- diversitree:::make.prior.ExpBeta(10, 1)
```

and running the MCMC sampler:

```
> samples.eb <- mcmc(lik.mk2, c(0.1, 0.1), nsteps = 5000, prior = prior.eb,
+     w = 0.1, lower = 0, print.every = 1000)

1000: {0.1160, 0.4310} -> -11.52772
2000: {0.0751, 0.2715} -> -10.72087
3000: {0.0711, 0.3396} -> -10.87517
4000: {0.1404, 0.4576} -> -11.92495
5000: {0.1266, 0.5338} -> -11.97329
```

```
> col <- c("red", "blue")
> profiles.plot(samples[c("q01", "q10")], col.line = col, las = 1,
+     xlab = "Parameter estimate", ylab = "Posterior probability density")
> abline(v = c(0.1, 0.2), col = col)
```

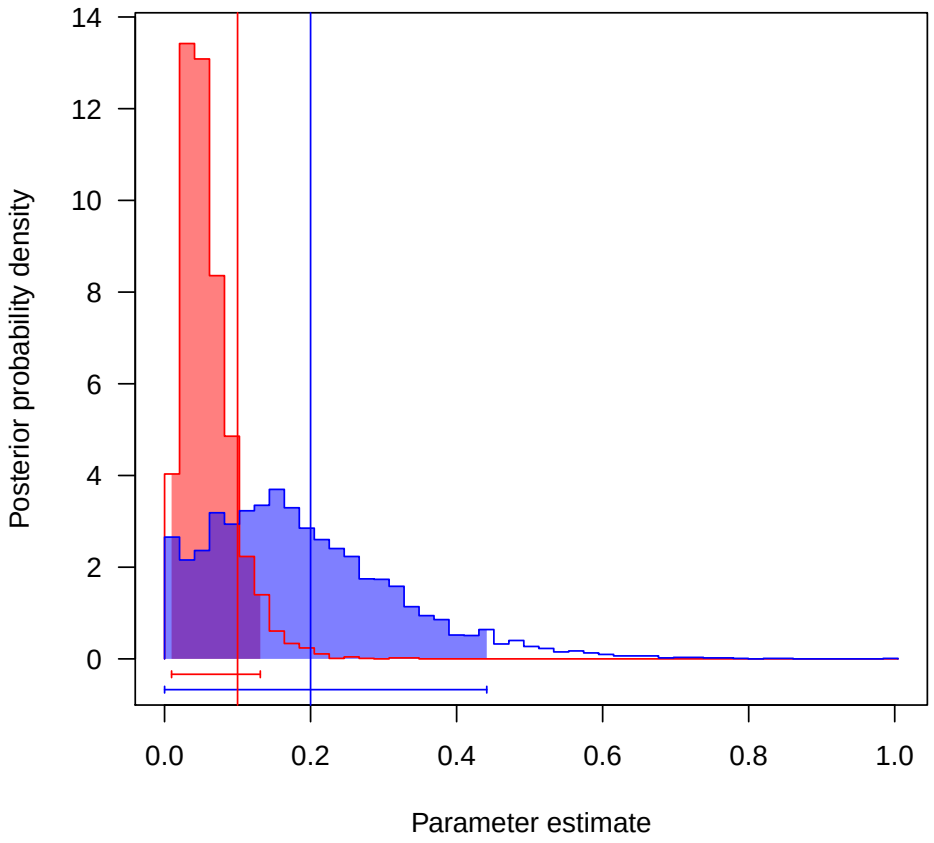

Figure 4: Posterior probability distributions for the parameters of the Mk2 model. True values are indicated by the solid vertical lines.

```
> samples.eb <- subset(samples.eb, i > 500)
```

The results from this are not plotted here, but give similar marginal distributions. However, the proportion of samples for which $q_{10} > q_{01}$ differs between the two sets of samples.

```
> mean(samples$q10 > samples$q01)

[1] 0.9091111

> mean(samples.eb$q10 > samples.eb$q01)

[1] 0.9777778
```

4 Binary traits and diversification: BiSSE

The BISSE (Binary State Speciation and Extinction) model combines the features of the constant-rates birth-death model with the two-state Markov model. Again, start with a simulated tree. The parameters here are in the order λ_0, λ_1, μ_0, μ_1, q_{01}, q_{10}, so the parameters below correspond to an asymmetry in the speciation rate where state 1 speciates at twice the rate as state 0. All other parameters are equal between states.

```
> pars <- c(0.1, 0.2, 0.03, 0.03, 0.01, 0.01)
> set.seed(4)
> phy <- tree.bisse(pars, max.t = 30, x0 = 0)
> states <- phy$tip.state
> head(states)

 sp4  sp8  sp9 sp14 sp15 sp16
   0    1    1    1    1    0
```

This gives a 52 species tree, shown with its true history in figure 5. The character states are now stored in the `states` vector. This vector is named, so that each element can be easily associated with a tip in the tree.

The `make.bisse` takes as its first two arguments a tree and set of character states (these are the only mandatory arguments):

```
> lik <- make.bisse(phy, states)
> lik(pars)

[1] -159.7128
```

To perform an ML, we need a starting point. The `starting.point.bisse` function produces a basic heuristic guess of a sensible starting point, based on the character-independent birth-death fit. There are no guarantees that this is at all close to the ML point, or that the ML point can be reached from this point while climbing uphill only (which most optimisers assume).

```
> p <- starting.point.bisse(phy)
> p
```

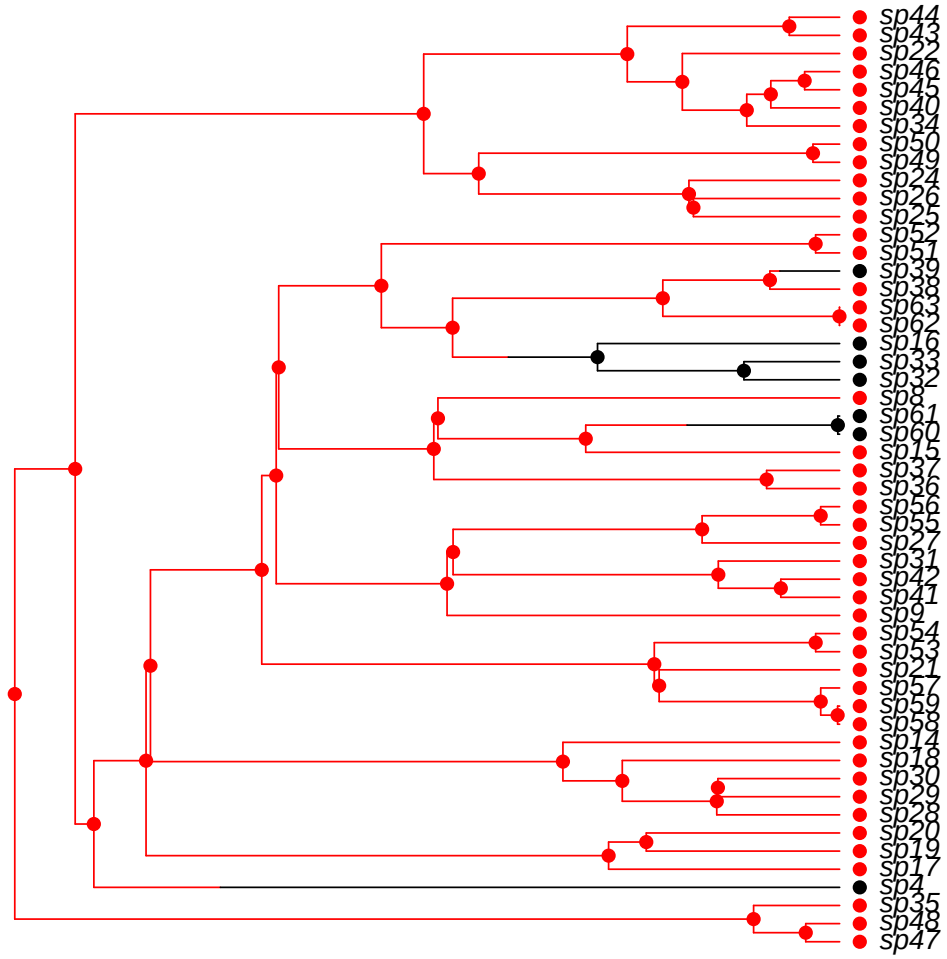

Figure 5: A BiSSE tree, with parameters $\lambda_0 = 0.1$, $\lambda_1 = 0.2$, $\mu_0 = \mu_1 = 0.03$, and $q_{01} = q_{10} = 0.01$. Black is state 0, red is state 1.

```
   lambda0    lambda1        mu0        mu1        q01        q10
0.17172409 0.17172409 0.04158759 0.04158759 0.02602730 0.02602730
```

Start an ML search from this point (this may take some time)

```
> fit <- find.mle(lik, p)
```

The `fit.mle` object has an element `lnLik` with the log-likelihood value

```
> fit$lnLik

[1] -158.6874
```

and coefficients may be extracted with `coef` (rounded for clarity):

```
> round(coef(fit), 3)

lambda0 lambda1     mu0     mu1     q01     q10
  0.134   0.178   0.121   0.036   0.000   0.019
```

Let's test the hypothesis that the speciation rates are different. We can use `constrain` to enforce equal speciation rates to be equal across character states:

```
> lik.l <- constrain(lik, lambda1 ~ lambda0)
```

and then start the ML search again:

```
> fit.l <- find.mle(lik.l, p[argnames(lik.l)])
> fit.l$lnLik

[1] -158.7357
```

(the statement "`p[argnames(lik.l)]`" drops the λ_1 element from the starting parameter vector). This fit has quite different parameters to the full model (compare μ_0)

```
> round(rbind(full = coef(fit), equal.l = coef(fit.l, TRUE)), 3)

        lambda0 lambda1   mu0   mu1 q01   q10
full      0.134   0.178 0.121 0.036   0 0.019
equal.l   0.173   0.173 0.173 0.027   0 0.022
```

(the `TRUE` argument forces `coef` to return values for constrained parameters). However, the difference in fits is not statistically supported, with $chi_1^2 = 0.1$:

```
> anova(fit, equal.l = fit.l)

        Df   lnLik    AIC    ChiSq Pr(>|Chi|)
full     6 -158.69 329.37
equal.l  5 -158.74 327.47 0.096609      0.756
```

4.1 Analysis with MCMC

Because we are fitting six parameters to a tree with only 52 species, priors will be needed so that the posterior distribution is proper. I will use an exponential prior with rate $1/(2r)$, where r is the character independent diversification rate:

```
> prior <- make.prior.exponential(1/(2 * (p[1] - p[3])))
```

The MCMC sampler in diversitree uses slice sampling (Neal, 2003) for parameter updates. The "step size" (argument `w`) does not need to be carefully tuned as it does not affect the rate of mixing – just the number of function evaluations per update. Ideally it will be on the same order as the width of the "high probability region". An easy way of setting this is to run a short chain (say, 100 steps) and use the range of observed samples as a measure of this.

```
> set.seed(1)
> tmp <- mcmc(lik, fit$par, nsteps = 100, prior = prior, lower = 0,
+     w = rep(1, 6), print.every = 0)
> w <- diff(sapply(tmp[2:7], range))
> w
```

```
       lambda0   lambda1       mu0       mu1       q01       q10
[1,] 0.4106028 0.2527382 0.6724629 0.2909625 0.5790865 0.1191832
```

Run the chain for 10,000 steps (this will take a while)

```
> samples <- mcmc(lik, fit$par, nsteps = 10000, w = w, lower = 0,
+     prior = prior, print.every = 0)
```

The marginal distributions for the two speciation rates are shown in figure 6, which shows the 95% credibility intervals for λ_0 completely overlapping those for λ_1.

4.2 Incomplete taxonomic sampling

Not all phylogenies are complete, but the basic BiSSE calculations assume that they are. If given tree contains a random sample of all extant species, the calculations can be corrected. To demonstrate this, we will generate a larger tree (150 taxa), and drop 50 taxa from it. Here, I am using the same parameters as earlier.

```
> pars <- c(0.1, 0.2, 0.03, 0.03, 0.01, 0.01)
> set.seed(4)
> phy <- tree.bisse(pars, max.taxa = 150, x0 = 0)
> states <- phy$tip.state
> phy.s <- drop.tip(phy, setdiff(seq_len(150), sample(150, 50)))
> states.s <- states[phy.s$tip.label]
```

Calculate what the sampling fraction is for this tree. You can either assume that the sampling fraction is independent of the character state:

```
> sampling.f <- 50/150
```

```
> col <- c("blue", "red")
> profiles.plot(samples[c("lambda0", "lambda1")], col.line = col,
+     las = 1, xlab = "Speciation rate", ylab = "Posterior probability density")
> legend("topright", c("lambda0", "lambda1"), col = col, lty = 1)
> abline(v = c(0.1, 0.2), col = col)
```

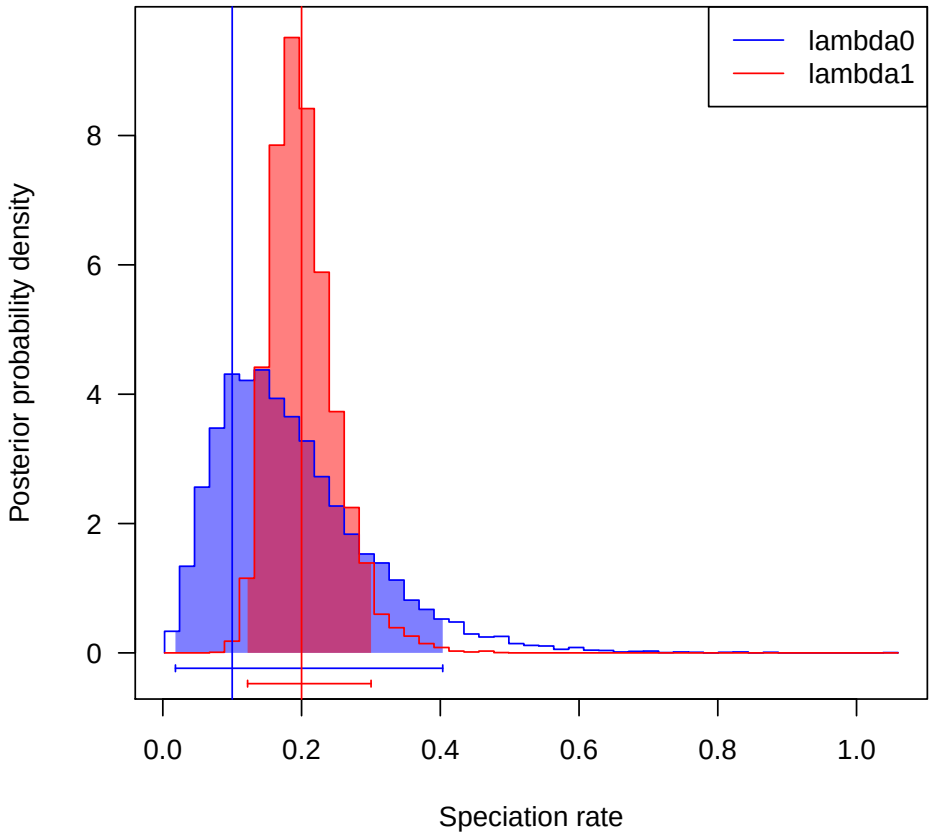

Figure 6: Posterior probability distributions for λ_0 and λ_1 for a BISSEmodel. True values are indicated by the solid vertical lines. The bars at the bottom of the distributions and the shaded areas correspond to the 95% credibility intervals.

or you can assume that it varies with character state

```
> sampling.f <- table(states.s)/table(states)
> sampling.f

states.s
        0         1
0.3846154 0.3284672
```

Pass this in to `make.bisse` and construct a new likelihood function that accounts for the sampling:

```
> lik.s <- make.bisse(phy.s, phy.s$tip.state, sampling.f = sampling.f)
```

This can then be optimised, as before:

```
> p <- starting.point.bisse(phy)
> fit.s <- find.mle(lik.s, p)
> fit.s[1:2]

$par
     lambda0      lambda1          mu0          mu1          q01          q10
9.411696e-02 2.260985e-01 1.002924e-08 9.340778e-02 1.121791e-06 5.542030e-03

$lnLik
[1] -170.2969
```

4.3 Terminally unresolved trees

Another way that phylogenies might be incompletely resolved is that higher level relationships may be known (say, genera), but little or nothing is known about species relationships within these groups. This results in trees where some "taxa" represent a number of species – "terminal clades". There are a couple of different ways that unresolved clade information may be specified. To demonstrate this, I will use an example of sexual dimorphism in shorebirds; this is the same example as in (FitzJohn et al., 2009). The phylogeny is a supertree constructed by Thomas et al. (2004), and the data on sexual size dimorphism are derived from Lislevand et al. (2007). The required files can be downloaded from `http://www.zoology.ubc.ca/prog/diversitree/files/` Read in the phylogenetic tree

```
> tree <- read.nexus("data/Thomas-tree.nex")
```

The tree contains many polytomies; the original tree, and a simplified tree with polytomies converted into clades are shown in figure 7. The character states are stored as the size of the difference of mass between sexes, divided by the mean across sexes (see FitzJohn et al., 2009).

```
> states <- read.csv("data/Lislevand-states.csv", as.is = TRUE)
> states <- structure(states$dimorph, names = states$species)
> states <- states[tree$tip.label]
> names(states) <- tree$tip.label
> head(states)
```

```
Catoptrophorus_semipalmatus         Calidris_ferruginea
                  -0.0988858                  -0.1022280
           Calidris_canutus           Calidris_maritima
                  -0.1605839                  -0.1512605
         Calidris_acuminata              Calidris_mauri
                   0.1016442                  -0.1445783
```

These will need converting to a binary character for use, for example – to convert this into a binary character where an absolute relative difference of 10% would be considered "dimorphic":

```
> head((abs(states) > 0.1) + 0)
```

```
Catoptrophorus_semipalmatus         Calidris_ferruginea
                          0                           1
           Calidris_canutus           Calidris_maritima
                          1                           1
         Calidris_acuminata              Calidris_mauri
                          1                           1
```

The simplest way of working with this tree is to use the `clades.from.polytomies` function. This collapses all daughters of any polytomy into a clade. (Be careful - if you have a polytomy at the base of your tree, the entire tree will collapse into a single clade!) This can be visualised with `plot` functions as normal – see `?plot.clade.tree` for more information. This tree can then be passed into `make.bisse`, along with a plain vector of state names. The catch is that every taxon still needs state information, not just those at the tips. Our state vector `states` includes states for all 350 species, so we are OK to use this.

```
> states.15 <- (abs(states) > 0.15) + 0
> tree.clade <- clades.from.polytomies(tree)
> lik <- make.bisse(tree.clade, states.15)
```

A sensible starting point can still be computed with `starting.point.bisse`; this takes into account the clade information automatically.

```
> p <- starting.point.bisse(tree.clade)
```

You can then perform a ML search or MCMC analysis as usual. However, unresolved clades slow down the analysis considerably, and this will take several minutes.

```
> fit.full <- find.mle(lik, p)
```

In the full model, speciation rates for dimorphic species (state 1) are greater than those for monomorphic species, but character transition rates away from dimorphism (q_{10}) are greater than the reverse transition (q_{10}):

```
> round(coef(fit.full), 3)
```

```
lambda0 lambda1     mu0     mu1     q01     q10
  0.054   0.202   0.000   0.000   0.041   0.277
```

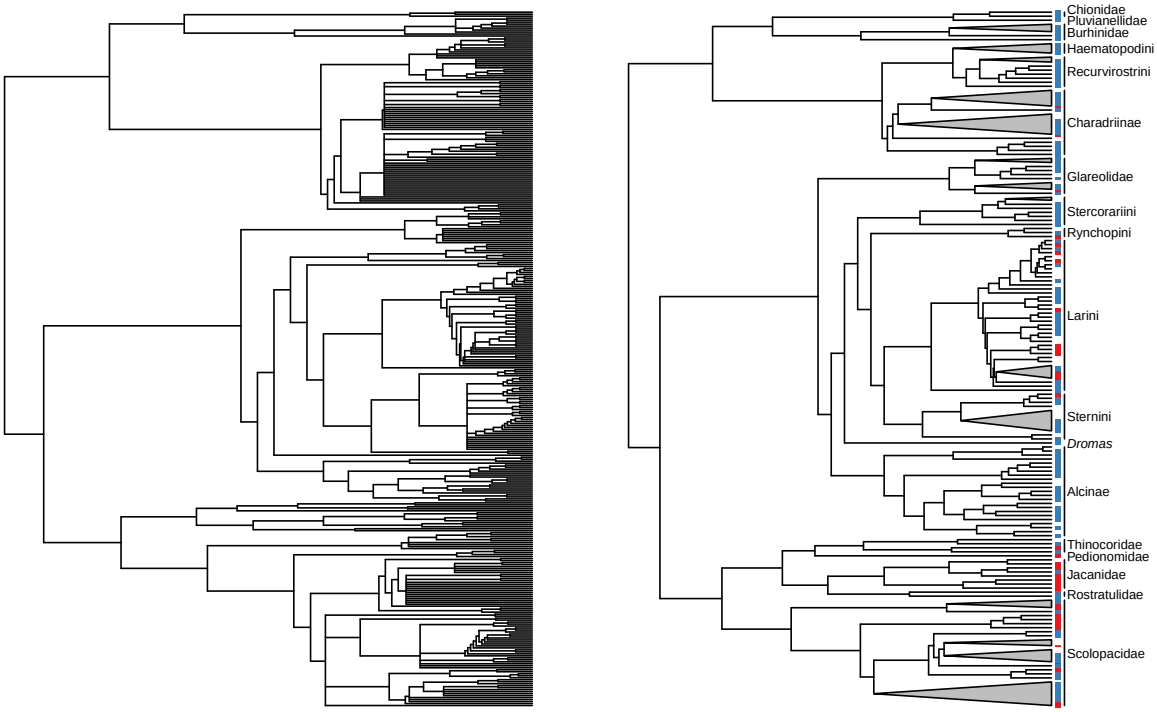

Figure 7: The shorebird supertree. On the left, the original tree with all polytomies and 350 species. On the right, the polytomies have been collapsed, to leave 135 tips, with tips representing from 1 to 47 species. The tips are colour coded to indicate state: red is sexually dimorphic, blue is monomorphic, and white is missing data.

To test whether these differences are significant, we can use a likelihood ratio test. First, construct the reduced models, constraining $\lambda_1 \sim \lambda_0$ or $q_{10} \sim q_{01}$:

```
> lik.l <- constrain(lik, lambda1 ~ lambda0)
> lik.q <- constrain(lik, q10 ~ q01)
```

Then, rerun the

```
> fit.l <- find.mle(lik.l, p[argnames(lik.l)])
> fit.q <- find.mle(lik.q, p[argnames(lik.q)])
```

These constrained models are significantly worse fits than the full model ($p \approx 0.04$ for both).

```
> anova(fit.full, equal.l = fit.l, equal.q = fit.q)
```

```
        Df   lnLik    AIC  ChiSq Pr(>|Chi|)
full     6 -633.14 1278.3
equal.l  5 -635.21 1280.4 4.1470    0.04171
equal.q  5 -635.21 1280.4 4.1477    0.04169
```

The analysis can also be run with MCMC. Here, I am using an exponential prior with rate $1/(2r)$, as earlier. This takes a very long time to run (approximately 4–5 s/sample)

```
> prior <- make.prior.exponential(1/(2 * p[1]))
> samples <- mcmc(lik, coef(fit.full), 10000, w = 0.3, prior = prior,
+     print.every = 0)
```

Setting up the likelihood function above assumed that the tree being used contained polytomies, and this is the source of the unresolved clades. However, it is probably more common to have an "exemplar" tree, where the unresolved species were never included in the first place. The tree `Thomas-tree-exemplar.nex` (which was derived from the tree above) does not contain any reference the species that are contained within the unresolved clades.

```
> tree.ex <- read.nexus("data/Thomas-tree-exemplar.nex")
> states.ex <- states.15[tree.ex$tip.label]
> names(states.ex) <- tree.ex$tip.label
```

We need to define a `data.frame` with information about the unresolved clades. The file `Thomas-unresolved.csv` contains the information in the correct format:

```
> unresolved <- read.csv("data/Thomas-unresolved.csv", as.is = TRUE)
> head(unresolved)
```

```
       tip.label Nc n0 n1
1 Catoptrophorus 47 34 10
2      Gallinago 16 10  1
3       Scolopax  6  1  1
4       Numenius  8  4  4
5         Sterna 37 20  1
6          Larus 17  5  9
```

All the columns here are required:

- `tip.label`: the tip label within the tree
- `Nc`: the total number of species that the tip represents
- `n0`: the number of species known to be in state `0`
- `n1`: the number of species known to be in state 1

(additional columns are fine and will be silently ignored). Note that `Nc` can be greater than `n0 + n1`: this allows for species with unknown state.

This `unresolved` object is passed into the `make.bisse` function:

```
> lik.ex <- make.bisse(tree.ex, states.ex, unresolved = unresolved)
```

This likelihood function should be identical to the one created above.

```
> lik.ex(coef(fit.full))

[1] -633.1378

> lik(coef(fit.full))

[1] -633.1378
```

5 Multiple state characters and diversification: MuSSE

MuSSE (Multiple State Speciation and Extinction) generalises the BiSSE model to allow characters with more than two states. Following Pagel (1994), this can also allow for multiple characters, each of which might be binary, by recoding the states.

To illustrate, we'll start with a simple simulated example with a three-level state. The tree is simulated where character evolution is only possible among neighbouring states (i.e., $1 \to 3$ and $3 \to 1$ transitions are disallowed). All other transitions are equal, and both speciation and extinction rates increase as the character number increases. For a three state case, the parameter vector is in the order $(\lambda_1, \lambda_2, \lambda_3, \mu_1, \mu_2, \mu_3, q_{12}, q_{13}, q_{21}, q_{23}, q_{31}, q_{32})$. This order can be seen here (sorry – clunky at the moment)

```
> diversitree:::argnames.musse(NULL, 3)

 [1] "lambda1" "lambda2" "lambda3" "mu1"     "mu2"     "mu3"     "q12"
 [8] "q13"     "q21"     "q23"     "q31"     "q32"
```

(the order of the q parameters is row-wise through the transition rate matrix, skipping diagonal elements).

Simulate a 30 species tree, with the tree starting in state 1.

```
> pars <- c(0.1, 0.15, 0.2, 0.03, 0.045, 0.06, 0.05, 0, 0.05, 0.05,
+     0, 0.05)
> set.seed(2)
> phy <- tree.musse(pars, 30, x0 = 1)
```

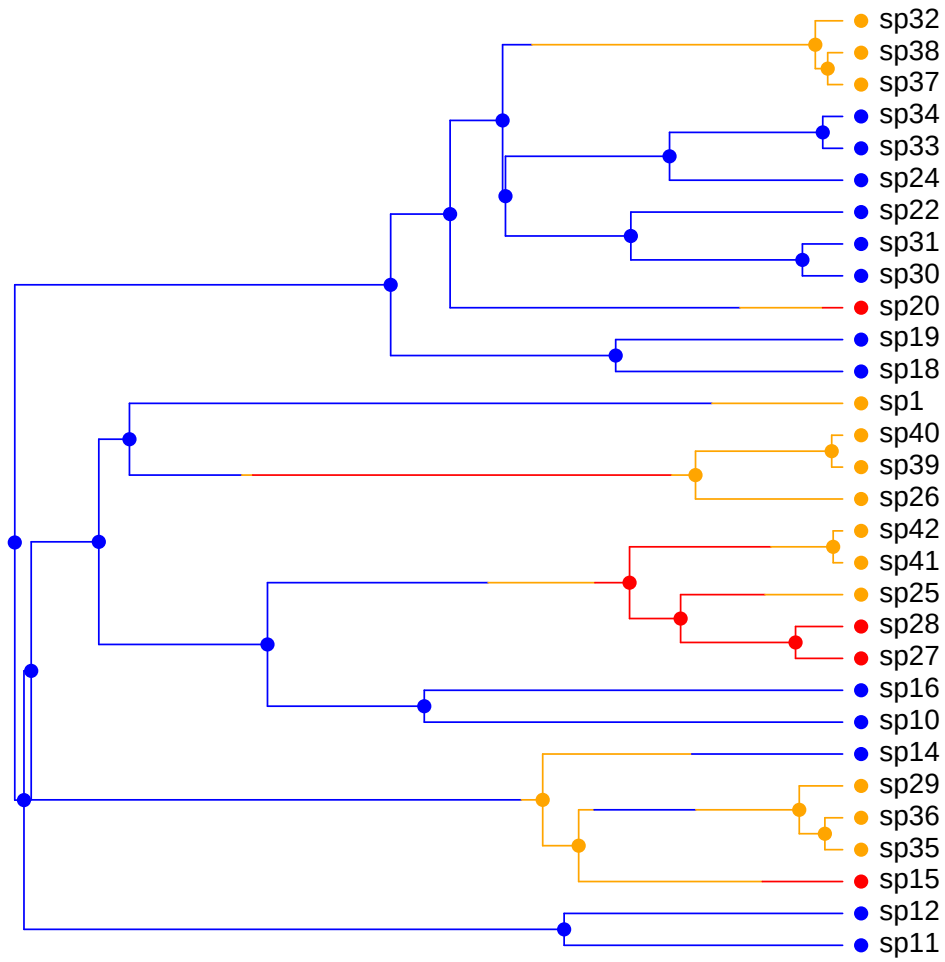

Figure 8: Simulated MUSSE tree. Blue is state 1, Orange is state 2, and red is state 3

The tree and its real character history are shown in figure
The states are numbered $1, 2, 3$, rather than $0, 1$ in BiSSE.

```
> states <- phy$tip.state
> table(states)

states
 1  2  3
13 13  4
```

Making a likelihood function is basically identical to BiSSE. The third argument needs to be the number of states. In a future version this will probably be `max(states)`, but there are some pitfalls about this that I am still worried about.

```
> lik <- make.musse(phy, states, 3)
```

The argument names here are in the same order as for the simulation. Just adding one more state (compared with BiSSE) has moved us up to 10 parameters.

```
> argnames(lik)

 [1] "lambda1" "lambda2" "lambda3" "mu1"     "mu2"     "mu3"     "q12"
 [8] "q13"     "q21"     "q23"     "q31"     "q32"
```

Rather than start with the full model, and constrain things, here I will start with a very simple model and expand. This model has all λ_i, μ_i, and q_i the same (except for q_{13} and q_{31}, which are still zero).

```
> lik.base <- constrain(lik, lambda2 ~ lambda1, lambda3 ~ lambda1,
+     mu2 ~ mu1, mu3 ~ mu1, q13 ~ 0, q21 ~ q12, q23 ~ q12, q31 ~
+         0, q32 ~ q12)
> argnames(lik.base)

[1] "lambda1" "mu1"     "q12"
```

Find the ML point for this model

```
> p <- starting.point.musse(phy, 3)
> fit.base <- find.mle(lik.base, p[argnames(lik.base)])
```

Now, allow the speciation rates to vary

```
> lik.lambda <- constrain(lik, mu2 ~ mu1, mu3 ~ mu1, q13 ~ 0, q21 ~
+     q12, q23 ~ q12, q31 ~ 0, q32 ~ q12)
> fit.lambda <- find.mle(lik.lambda, p[argnames(lik.lambda)])
```

There is very little improvement here (this is a small tree)

```
> anova(fit.base, free.lambda = fit.lambda)

            Df   lnLik    AIC  ChiSq Pr(>|Chi|)
full         3 -110.64 227.27
free.lambda  5 -110.11 230.22 1.0466     0.5926
```

6 Topics not covered

I have not covered several included models here, but information can be found in the online documentation.

- MEDUSA-style "split" models. Following (Alfaro et al., 2009), these models allow different regions of the tree to have different parameters. This is implemented for birth-death models (`make.bd.split`, which is identical to MEDUSA), BiSSE(`?make.bisse.split`), and MuSSE (?make.musse.split).
- Time-dependent models. In these, time is divided into "epochs", each of which may have different parameters. This is implemented for BiSSE (`?make.bisse.td`) and MuSSE (`?make.musse.td`).
- Brownian motion. A very simple-minded Brownian motion likelihood calculation is included. This allows estimation of the diffusion parameter of a Brownian motion process for the evolution of a single continuous trait (`?make.bm`).

References

Alfaro M.E., Santini F., Brock C., Alamillo H., Dornburg A., Rabosky D.L., Carnevale G., and Harmon L.J. 2009. Nine exceptional radiations plus high turnover explain species diversity in jawed vertebrates. Proceedings of the National Acadamy of Sciences, USA 106:13410–13414.

Bollback J.P. 2006. Simmap: Stochastic character mapping of discrete traits on phylogenies. BMC Bioinformatics 7:88.

FitzJohn R.G. 2010. Quantitative traits and diversification. Systematic Biology 59:in press. doi:10.1093/sysbio/syq053.

FitzJohn R.G., Maddison W.P., and Otto S.P. 2009. Estimating trait-dependent speciation and extinction rates from incompletely resolved phylogenies. Systematic Biology 58:595–611.

Lislevand T., Figuerola J., and Székely T. 2007. Avian body sizes in relation to fecundity, mating system, display behavior, and resource sharing. Ecology 88:1605.

Maddison W.P., Midford P.E., and Otto S.P. 2007. Estimating a binary character's effect on speciation and extinction. Syst. Biol. 56:701–710.

Neal R.M. 2003. Slice sampling. The Annals of Statistics 31:705–767.

Nee S., May R.M., and Harvey P.H. 1994. The reconstructed evolutionary process. Philos. Trans. R. Soc. Lond. B Biol. Sci. 344:305–311.

Pagel M. 1994. Detecting correlated evolution on phylogenies: a general method for the comparative analysis of discrete characters. Proceedings of the Royal Society of London, B 255:37–45.

Thomas G.H., Wills M.A., and Székely T. 2004. A supertree approach to shorebird phylogeny. BMC Evolutionary Biology 4:28.