Phylogenetic Inference using RevBayes

Chromosome Evolution

William A. Freyman and Sebastian Höhna

Introduction

A central organizing component of the higher-order architecture of the genome is chromosome number, and changes in chromosome number have long been understood to play a fundamental role in evolution. This tutorial will introduce phylogenetic models of chromosome number evolution, and demonstrate how to use `RevBayes` to estimate the rates of chromosome number change and ancestral chromosome numbers. We will also show how to use the `RevGadgets` R package to make plots of ancestral chromosome number estimates and stochastic character maps of chromosome evolution.

We will begin by providing an overview of the basic ChromEvol model (Mayrose et al. 2010) and an example `RevBayes` analysis. This is followed by a discussion of a number of model extensions that enable joint inference of phylogeny and chromosome numbers, tests for correlated rates of phenotype and chromosome evolution (the BiChroM model; Zenil-Ferguson et al. 2017), and incorporating cladogenetic changes in chromosome number. Next, we will introduce the ChromoSSE model (Freyman and Höhna 2016) which jointly estimates diversification rates and chromosome number evolution. Finally, we will briefly discuss comparing models of chromosome evolution using reversible-jump MCMC and Bayes factors.

If you use `RevBayes` for chromosome evolution analysis, please cite the original papers that describe the chromosome evolution model as well as Freyman and Höhna (2016) which describes in detail the `RevBayes` implementation of these models.

Contents

The Chromosome Number Evolution tutorial contains several sections:

Recommended tutorials

This tutorial assumes the reader is familiar with the content covered in the following `RevBayes` tutorials:

- `Rev` **Basics**
- **Molecular Models of Character Evolution**
- **Discrete Morphology Models of Character Evolution**
- **Running and Diagnosing an MCMC Analysis**

1 Overview of Chromosome Number Evolution Models

Chromosome changes represent major evolutionary mechanisms that have long been a focal point of study. Changes in chromosome number such as the gain or loss of a single chromosome (dysploidy), or the doubling of the entire genome (polyploidy), can have phenotypic consequences, affect the rates of recombination, and increase reproductive isolation among lineages and thus drive diversification (Stebbins 1971). Recently, evolutionary biologists have increasingly studied the macroevolutionary consequences of chromosome changes within a molecular phylogenetic framework, mostly utilizing the likelihood-based ChromEvol models of chromosome number evolution introduced by Mayrose et al. (2010). The ChromEvol models have permitted phylogenetic studies of ancient whole genome duplication events, rapid "catastrophic" chromosome speciation, major reevaluations of the evolution of angiosperms, and new insights into the fate of polyploid lineages (e.g. Pires and Hertweck 2008; Mayrose et al. 2011; Tank et al. 2015). The basic ChromEvol model has been extended to examine the association of phenotype with chromosome evolution (BiChroM; Zenil-Ferguson et al. 2017), and to incorporate cladogenetic changes and diversification rates (ChromoSSE; Freyman and Höhna 2016).

Here we describe the ChromEvol model as implemented in `RevBayes`, which except for one detail noted below is the same mathematical model introduced in Mayrose et al. (2010). In further sections, we will show how to set up extensions such as BiChroM and ChromoSSE which build on the useful but basic ChromEvol model of chromosome number evolution.

1.1 The ChromEvol Model

In ChromEvol the evolution of chromosome number is represented as a continuous-time Markov process, similar to models of molecular evolution and discrete morphological evolution. The continuous-time Markov process is described by an instantaneous rate matrix Q where the value of each element represents the instantaneous rate of change within a lineage from a genome of i chromosomes to a genome of j chromosomes. For all elements of Q in which either $i = 0$ or $j = 0$ we define $Q_{ij} = 0$. For the off-diagonal elements $i \neq j$ with positive values of i and j, Q is determined by:

$$Q_{ij} = \begin{cases} \gamma_a & j = i + 1, \\ \delta_a & j = i - 1, \\ \rho_a & j = 2i, \\ \eta_a & j = 1.5i, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where γ_a, δ_a, ρ_a, and η_a are the rates of chromosome gains, losses, polyploidizations, and demi-polyploidizations.

If we are interested in modeling scenarios in which the probability of fusion or fission events are positively or negatively correlated with the number of chromosomes we can define Q as:

$$Q_{ij} = \begin{cases} \gamma_a e^{\gamma_m (i-1)} & j = i + 1, \\ \delta_a e^{\delta_m (i-1)} & j = i - 1, \\ \rho_a & j = 2i, \\ \eta_a & j = 1.5i, \\ 0 & \text{otherwise}, \end{cases} \tag{2}$$

where γ_m and δ_m are rate modifiers of chromosome gain and loss, respectively, that allow the rates of chromosome gain and loss to depend on the current number of chromosomes. If the rate modifier $\gamma_m = 0$, then $\gamma_a e^{0(i-1)} = \gamma_a$. If the rate modifier $\gamma_m > 0$, then $\gamma_a e^{\gamma_m(i-1)} \geq \gamma_a$, and if $\gamma_m < 0$ then $\gamma_a e^{\gamma_m(i-1)} \leq \gamma_a$. Note that this parameterization differs slightly from the original ChromEvol model; here we assume the rates of chromosome change can vary exponentially as a function of the current chromosome number, whereas ChromEvol as originally described by Mayrose et al. (2010) assumes a linear function. The theoretical reasons for this difference are described in Freyman and Höhna (2016), however in practice on most empirical datasets the difference appears insignificant.

Demi-polyploidization is the union of a reduced and an unreduced gametes that produces a cytotype with 1.5 times the number of chromosomes. The number of chromosomes in a genome must of course be an integer, so for odd values of i, Q_{ij} is set to $\eta/2$ for the two integer values of j resulting when $j = 1.5i$ is rounded up and down.

As in all continuous-time Markov models, the diagonal elements $i = j$ of Q are defined as:

$$Q_{ii} = -\sum_{i \neq j}^{C_m} Q_{ij}. \tag{3}$$

The probability of anagenetically evolving from chromosome number i to j along a branch of length t is then calculated by exponentiation of the instantaneous rate matrix:

$$P_{ij}(t) = e^{-Qt}. \tag{4}$$

Given a phylogeny and chromosome counts of the extant lineages, this model can be used in either a maximum likelihood or Bayesian inference framework to estimate the rates of chromosome change and the ancestral chromosome numbers.

1.2 Next Steps

The basic ChromEvol model as described above can be extended in a number of useful ways that will be covered in further sections. In the next section, however, we'll set up and run a simple `RevBayes` analysis using the ChromEvol model before moving on to the more complex models.

2 Example: A Simple ChromEvol Analysis

In this example, we will use molecular sequence data and chromosome counts from Ohi-Toma et al. (2006) of the plant genus *Aristolochia* (commonly called Dutchman's pipe plants). We will use a simple ChromEvol model to infer rates of chromosome evolution and ancestral chromosome numbers.

2.1 Tutorial Format

This tutorial follows a specific format for issuing instructions and information.

> The boxed instructions guide you to complete tasks that are not part of the `RevBayes` syntax, but rather direct you to create directories or files or similar.

Information describing the commands and instructions will be written in paragraph-form before or after they are issued.

All command-line text, including all `Rev` syntax, are given in **`monotype font`**. Furthermore, blocks of `Rev` code that are needed to build the model, specify the analysis, or execute the run are given in separate shaded boxes. For example, we will instruct you to create a constant node called **`example`** that is equal to `1.0` using the **`<-`** operator like this:

```
example <- 1.0
```

It is important to be aware that some PDF viewers may render some characters given as `Rev commands` differently. Thus, if you copy and paste text from this PDF, you may introduce some incorrect characters. Because of this, we recommend that you type the instructions in this tutorial or copy them from the scripts provided.

2.2 Data and Files

> On your own computer, create a directory called **`RB_Chromosome_Evolution_Tutorial`** (or any name you like).
>
> In this directory download and unzip the archive containing the data files: **`data.zip`**.
>
> This will create a folder called **`data`** that contains the files necessary to complete this exercise.

2.3 Creating the `Rev` File

> Create a new directory (in **`RB_Chromosome_Evolution_Tutorial`**) called **`scripts`**. (If you do not have this folder, please refer to the directions in section 2.2.)

When you execute `RevBayes` in this exercise, you will do so within the main directory you created (**`RB_Chromosome_Evolution_Tutorial`**), thus, if you are using a Unix-based operating system, we recommend that you add the `RevBayes` binary to your path.

For complex models and analyses, it is best to create `Rev` script files that will contain all of the model parameters, moves, and functions. In this first section, you will create a file called `ChromEvol_simple.Rev` from scratch and save it in the **`scripts`** directory.

The full scripts for these examples are also provided in the `RevBayes` tutorial repository[1]. Please refer to these files to verify or troubleshoot your own scripts.

Open your text editor and create the master `Rev` file called `ChromEvol_simple.Rev` in the **`scripts`** directory.

Enter the `Rev` code provided in this section into this file.

The file you will begin in this section will be the one you load into `RevBayes` when you've completed all of the components of the analysis. The file will contain the `Rev` code to load the data files, set up the model, run the MCMC analysis, and summarize the results.

2.4 Reading in Data

First, we'll read in the phylogeny. In this example the phylogeny is assumed known. In further examples we'll jointly estimate chromosome evolution and the phylogeny.

```
phylogeny <- readBranchLengthTrees("data/aristolochia.tree")[1]
```

We need to limit the maximum number of chromosomes allowed in our model, so here we use the largest observed chromosome count plus 10. This is an arbitrary limit on the size of the state space that could be increased if necessary.

```
max_chromo = 26
```

Now we get the observed chromosome counts from a tab-delimited file.

```
chromo_data = readCharacterDataDelimited("data/aristolochia_chromosome_counts.tsv",
    stateLabels=(max_chromo + 1), type="NaturalNumbers", delimiter="\t", headers=FALSE)
```

[1] https://github.com/revbayes/revbayes_tutorial/tree/master/RB_Chromosome_Evolution_Tutorial/scripts

2.5 The Chromosome Evolution Model

We'll use exponential priors to model the rates of polyploidy and dysploidy events along the branches of the phylogeny. `gamma` is the rate of chromosome gains, `delta` is the rate of chromosome losses, and `rho` is the rate of polyploidization.

```
gamma ~ dnExponential(10.0)
delta ~ dnExponential(10.0)
rho ~ dnExponential(10.0)
```

Add MCMC moves for each of the rates.

```
mvi = 1
moves[mvi++] = mvScale(gamma, lambda=1, weight=1)
moves[mvi++] = mvScale(delta, lambda=1, weight=1)
moves[mvi++] = mvScale(rho, lambda=1, weight=1)
```

Now we create the rate matrix for the chromosome evolution model. Here we will use a simple ChromEvol model that includes only the rate of chromosome gain, loss, and polyploidization.

```
Q := fnChromosomes(max_chromo, gamma, delta, rho)
```

Parameters for demi-polyploidization and rate modifiers could also be added at this step for more complex models. For example, we could have included the rate of demi-polyploidization `eta` and rate modifiers like this:

```
Q := fnChromosomes(max_chromo, gamma, delta, rho, eta, gamma_l, delta_l)
```

Here we assume the frequency of chromosome numbers at the root of the tree are equal. Alternatively, we could have treated the root frequencies as a free variable and estimated them from the observed data. This approach will be illustrated in further examples.

```
root_frequencies := simplex(rep(1, max_chromo + 1))
```

Finally, we create the stochastic node for the chromosome evolution continuous-time Markov chain (CTMC). We also clamp the observed chromosome count data to the CTMC.

```
chromo_ctmc ~ dnPhyloCTMC(Q=Q, tree=phylogeny, rootFreq=root_frequencies, type="
   NaturalNumbers")
chromo_ctmc.clamp(chromo_data)
```

All of the components of the model are now specified, so now we wrap it into a single model object.

```
mymodel = model(phylogeny)
```

2.6 Set Up the MCMC

The next important step for our master `Rev` file is to specify the MCMC monitors. For this, we create a vector called **`monitors`** that will each output MCMC samples. First, a screen monitor that will output every 10 iterations:

```
monitors[1] = mnScreen(printgen=10)
```

Next, an ancestral state monitor which will sample ancestral states and write them to a log file. We could additionally use the `mnStochasticCharacterMap` monitor to sample stochastic character maps of chromosome evolution (see the next section for an example).

```
monitors[2] = mnJointConditionalAncestralState(filename="output/
   ChromEvol_simple_anc_states.log", printgen=10, tree=phylogeny, ctmc=chromo_ctmc,
   type="NaturalNumbers")
```

And another monitor for logging all the model parameters. This will generate a file that can be opened in `Tracer` for checking MCMC convergence and parameter estimates.

```
monitors[3] = mnModel(filename="output/ChromEvol_simple_model.log", printgen=10)
```

Now we set up the MCMC and include code to execute the analysis. In this example we set the chain length to `200`, however for a real analysis you would want to run many more iterations and check for convergence.

```
mymcmc = mcmc(mymodel, monitors, moves)
mymcmc.run(200)
```

2.7 Summarize Ancestral States

Now we need to add Rev code that will summarize the sampled ancestral chromosome numbers. First, read in the ancestral state trace generated by the ancestral state monitor during the MCMC analysis:

```
anc_state_trace = readAncestralStateTrace("output/ChromEvol_simple_anc_states.log")
```

Finally, summarize the values from the traces over the phylogeny. Here we do a marginal reconstruction of the ancestral states, discarding the first 25% of samples as burnin. This will produce the file ChromEvol_simple_final.tree that contains the phylogeny along with estimated ancestral states. We can use that file with the RevGadgets R package to generate a plot of the ancestral states.

```
ancestralStateTree(phylogeny, anc_state_trace, "output/ChromEvol_simple_final.tree",
    burnin=0.25, reconstruction="marginal")
```

Note that we could also have calculated joint or conditional ancestral states instead of (or in addition to) the marginal ancestral states. If we had sampled stochastic character maps, we would summarize them with the characterMapTree function.

And now quit RevBayes:

```
q()
```

You made it! Be sure to save the file.

2.8 Execute the RevBayes Analysis

With all the parameters specified and all analysis components in place, you are now ready to run your analysis. The Rev scripts you just created will all be used by RevBayes and loaded in the appropriate order.

Begin by running the RevBayes executable. In Unix systems, type the following in your terminal (if the RevBayes binary is in your path):

```
rb scripts/ChromEvol_simple.Rev
```

Provided that you started RevBayes from the correct directory (**RB_Chromsome_Evolution_Tutorial**) the analysis should now run. Alternatively, from within RevBayes you could use the **source()** function to feed RevBayes your master script file:

```
source("scripts/ChromEvol_simple.Rev")
```

This will execute the analysis and you should see output similar to this (though not the exact same values):

```
RevBayes

Visit the website www.RevBayes.com for more information about RevBayes.

RevBayes is free software released under the GPL license, version 3. Type 'license()' for details.

To quit RevBayes type 'quit()' or 'q()'.


> source("scripts/ChromEvol_simple.Rev")
   Processing file "scripts/ChromEvol_simple.Rev"
   Attempting to read the contents of file "aristolochia.tree"
   Successfully read file

   Running MCMC simulation
   This simulation runs 1 independent replicate.
   The simulator uses 3 different moves in a random move schedule with 3 moves per iteration

Iter        |      Posterior   |     Likelihood   |          Prior   |    elapsed   |        ETA   |
----------------------------------------------------------------------------------------------------
0           |       -61.8458   |       -63.6701   |        1.82431   |   00:00:00   |   --:--:--   |
10          |       -59.8779   |       -60.1293   |       0.251464   |   00:00:02   |   --:--:--   |
20          |        -59.251   |       -55.6886   |       -3.56249   |   00:00:04   |   00:00:36   |
30          |       -60.1883   |       -53.1068   |       -7.08147   |   00:00:06   |   00:00:34   |
40          |       -61.1943   |       -52.6285   |       -8.56577   |   00:00:09   |   00:00:36   |
50          |       -60.7165   |       -54.7084   |       -6.00804   |   00:00:11   |   00:00:33   |

...
```

When the analysis is complete, `RevBayes` will quit and you will have a new directory called **`output`** that will contain all of the files you specified with the monitors (Sect. 2.6).

2.9 Plotting the Results

Now we will plot the results of the MCMC analysis using the **`RevGadgets`** R package. Start R and set your working directory to the **`RB_Chromsome_Evolution_Tutorial`** directory. Now run the command `source("scripts/plot_ChromEvol_simple.R")` to generate Figure 1 below. There are many options to customize the look of the plot, for options take a look inside the R script.

2.10 Next Steps

There are many extensions to the basic ChromEvol analysis demonstrated here. In the next section we will look at how to set up more complex chromosome number evolution analyses.

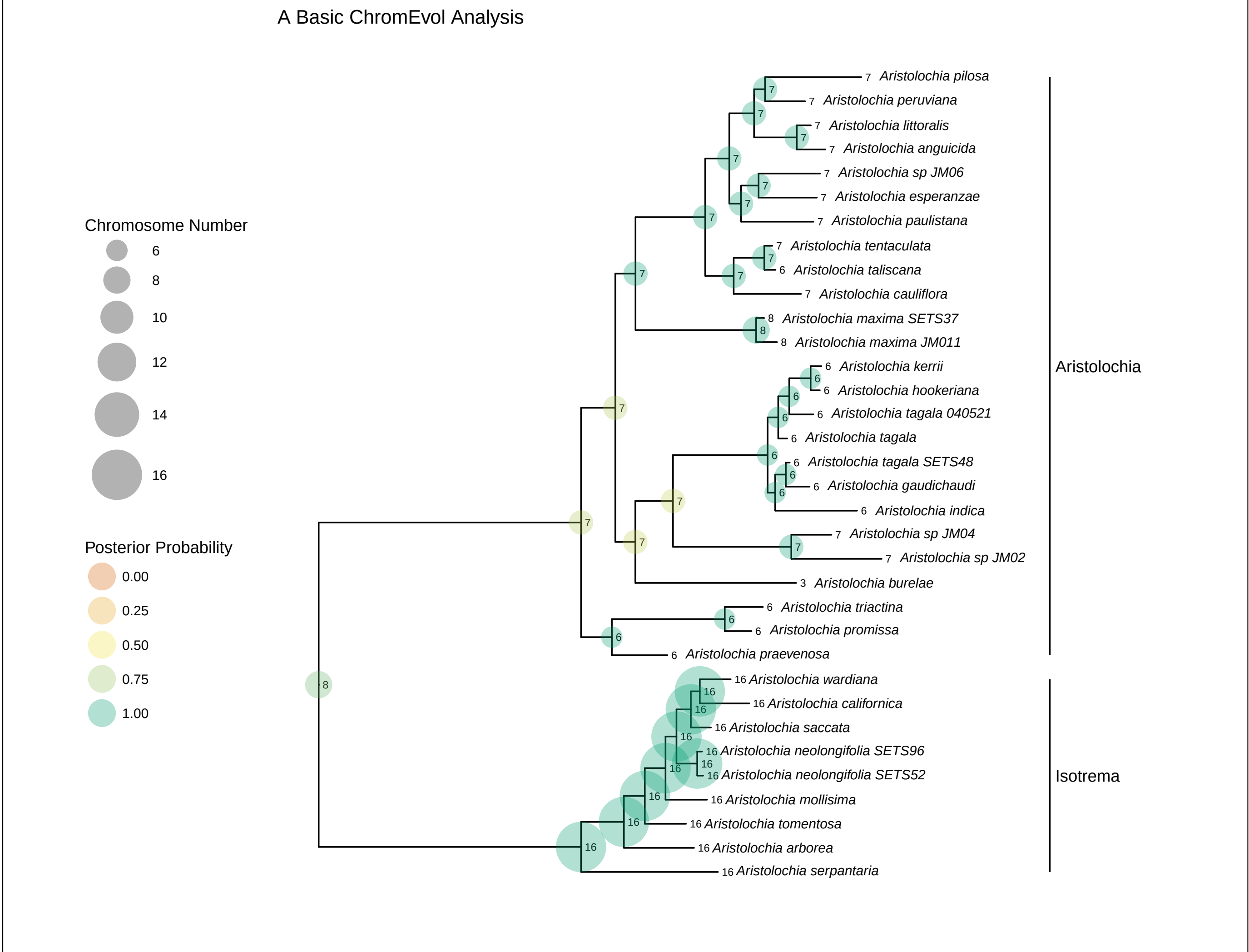

Figure 1: Maximum a posteriori ancestral chromosome number estimates for *Aristolochia*.

3 Basic Extensions of the Chromosome Evolution Model

In this section, we will extend the ChromEvol model in a number of ways. First, we will examine another approach for treating chromosome number root frequencies. This is followed by a brief example applying stochastic character mapping to chromosome evolution models. Then will look at jointly estimating the phylogeny and chromsome evolution, show how to set up a BiChroM analysis, and demonstrate one way to add cladogenetic changes to a chromosome evolution analysis.

Like before, scripts for these examples are also provided in the `RevBayes` tutorial repository[2]. Please refer to these files to verify or troubleshoot your own scripts.

[2] `https://github.com/revbayes/revbayes_tutorial/tree/master/RB_Chromosome_Evolution_Tutorial/scripts`

3.1 Improved Root Frequencies

In the last example we assumed the frequency of chromosome numbers at the root of the tree were equal. This is equivalent to assigning an extremely informative prior that all root states are equally likely. An alternative approach is to treat the root frequencies as free parameters of the model and estimate them from the observed data. In a series of unpublished simulations performed by the authors this resulted in increased accuracy of ancestral root chromosome numbers estimates.

To use this approach, the `root_frequencies` parameter must be redefined as a stochastic node in our graphical model instead of a deterministic node. Remove the following line from your `Rev` script:

```
root_frequencies := simplex(rep(1, max_chromo + 1))
```

We will instead use an uninformative flat Dirichlet prior for the root frequencies. First, we create a vector to hold the concentration parameters for the Dirichlet distribution. Here set all concentration parameters to 1, which results in all sets of probabilities being equally likely. We then pass the vector of concentration parameters into the Dirichlet distribution and create the stochastic node representing root frequencies.

```
root_frequencies_prior <- rep(1, max_chromo + 1)
root_frequencies ~ dnDirichlet(root_frequencies_prior)
```

Next, we must specify MCMC moves for the root frequencies. When the maximum number of chromosomes is high these parameters can have difficulty converging. Therefore, we use two different MCMC moves. The first is Beta Simplex move, which selects one element of the `root_frequencies` vector and proposes a new value for it drawn from a Beta distribution. The second is Element Swap Simplex move, which selects two elements of the `root_frequencies` vector and simply swaps their values.

```
moves[mvi++] = mvBetaSimplex(root_frequencies, alpha=0.5, weight=10)
moves[mvi++] = mvElementSwapSimplex(root_frequencies, weight=10)
```

You can experiment with different weights for each MCMC move.

3.2 Stochastic Character Mapping of Chromosome Evolution

In `RevBayes` both ancestral states and stochastic character maps can be sampled from continuous-time Markov chain (CTMC) and state-dependent speciation and extinction (SSE) models of character evolution. Stochastic character maps show the timing and number of transitions along the branches of the phylogeny, so they can be particularly useful for chromosome evolution estimates where the timing of, for example, whole genome duplication events might be of interest. This example is performed on a non-ultrametric tree, but obviously the same analysis could be performed on time-calibrated trees.

We have already shown how to sample ancestral states above, and here we show the few extra lines of `Rev` code needed to sample stochastic character maps. Stochastic character maps are drawn during the MCMC, so we need to include the `mnStochasticCharacterMap` monitor.

```
monitors[4] = mnStochasticCharacterMap(ctmc=chromo_ctmc, filename="output/
    ChromEvol_maps.log", printgen=10)
```

This monitor will create the `output/ChromEvol_maps.log` file. Just like the other log files, each row in this file represents a different sample from the MCMC. Each column in the file, though, is the character history for a different node in the phylogeny. The last column of the file is the full stochastic character map of the entire tree in **SIMMAP** (Bollback 2006) format. These can be plotted using the **phytools** R package (Revell 2012).

After the MCMC, we can calculate the maximum a posteriori marginal, joint, or conditional character history. This process is similar to the ancestral state summaries. First we read in the stochastic character map trace.

```
anc_state_trace = readAncestralStateTrace("output/ChromEvol_simple_maps.log")
```

Then we use the `characterMapTree` function. This generates two SIMMAP formatted files: 1) the maximum a posteriori character history, and 2) the posterior probabilities of the entire character history.

```
characterMapTree(phylogeny, anc_state_trace, character_file="output/character.tree",
    posterior_file="output/posterior.tree", burnin=5, reconstruction="marginal")
```

Figure 2 is an example stochastic character map of our *Aristolochia* analysis plotted using **phytools**.

3.3 Joint Estimation of Phylogeny and Chromosome Evolution

In `RevBayes` the chromosome evolution models can be used jointly with a model of molecular evolution enabling joint inference of the phylogeny and chromosome number evolution. This enables the chromosome number analysis to take into account phylogenetic uncertainty and allows the chromosome numbers to help inform the phylogeny.

Setting up a model that jointly infers chromosome evolution and phylogeny is mostly simply combining elements covered in the **Molecular Models of Character Evolution** tutorial with the what has already been covered in Section 2 of this tutorial. We will not repeat how to set up the chromosome model component, but show what must be added to the example in Section 2 above. However, we have provided a full working example script `scripts/ChromEvol_joint.Rev`.

3.3.1 Reading in Molecular Data and Setting Clade Constraints

The first major difference from the basic ChromEvol example shown above is that we must additionally read in molecular sequence data:

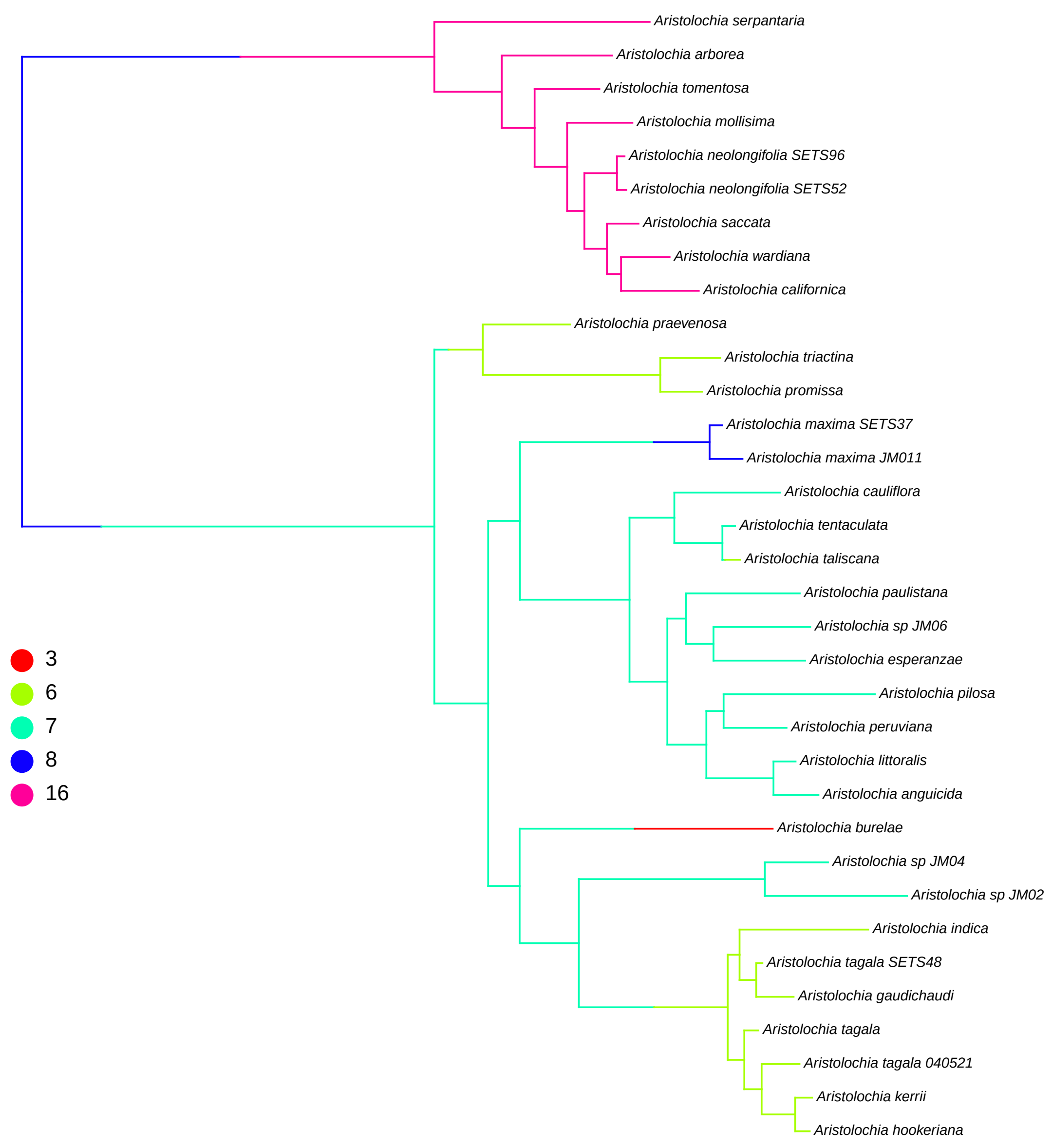

Figure 2: An example of stochastic character mapping applied to chromosome number evolution using RevBayes. Shown is the marginal maximum a posteriori chromosome evolution history of *Aristolochia* using the simple ChromEvol analysis from Section 2.

```
dna_seq = readDiscreteCharacterData("data/aristolochia_matK.fasta")
```

We will need some useful information about this data as well:

```
n_species = dna_seq.ntaxa()
n_sites = dna_seq.nchar()
```

```
taxa = dna_seq.names()
n_branches = 2 * n_species - 2
```

Since we want to jointly infer ancestral states, we need to set an a priori rooting constraint on our phylogeny. So here we set an ingroup and outgroup.

```
outgroup = ["Aristolochia_serpantaria", "Aristolochia_arborea", "Aristolochia_wardiana
    ",
           "Aristolochia_californica", "Aristolochia_saccata", "Aristolochia_mollisima
               ",
           "Aristolochia_tomentosa", "Aristolochia_neolongifolia_SETS52", "
               Aristolochia_neolongifolia_SETS96"]
```

Here we loop through each taxon and if it is not present in the outgroup defined above we add it to the ingroup.

```
i = 1
for (j in 1:taxa.size()) {
    found = false
    for (k in 1:outgroup.size()) {
        if (outgroup[k] == taxa[j].getSpeciesName()) {
            found = true
            break
        }
    }
    if (found == false) {
        ingroup[i] = taxa[j].getSpeciesName()
        i += 1
    }
}
```

And now we make the vector of clade objects to constrain our tree topology.

```
clade_ingroup = clade(ingroup)
clade_outgroup = clade(outgroup)
clade_constraints = [clade_ingroup, clade_outgroup]
```

3.3.2 Tree Model

We will specify a uniform prior on the tree topology, and add a MCMC move on the topology.

```
topology ~ dnUniformTopology(taxa=taxa, constraints=clade_constraints, rooted=TRUE)
moves[mvi++] = mvNNI(topology, weight=10.0)
```

Next, we create a stochastic node for each branch length. Each branch length prior will have an exponential distribution with rate 1.0. We'll also add a simple scaling move for each branch length.

```
for (i in 1:n_branches) {
    br_lens[i] ~ dnExponential(10.0)
    moves[mvi++] = mvScale(br_lens[i], lambda=2, weight=1)
}
```

Finally, build the tree by combining the topology with the branch lengths.

```
phylogeny := treeAssembly(topology, br_lens)
```

3.3.3 Molecular Substitution Model

We'll specify the GTR substitution model applied uniformly to all sites. Use a flat Dirichlet prior for the exchange rates.

```
er_prior <- v(1,1,1,1,1,1)
er ~ dnDirichlet(er_prior)
moves[mvi++] = mvSimplexElementScale(er, alpha=10, weight=3)
```

And also a flat Dirichlet prior for the stationary base frequencies.

```
pi_prior <- v(1,1,1,1)
pi ~ dnDirichlet(pi_prior)
moves[mvi++] = mvSimplexElementScale(pi, alpha=10, weight=2)
```

Now create a deterministic variable for the nucleotide substitution rate matrix.

```
Q_mol := fnGTR(er, pi)
```

Create a stochastic node for the sequence evolution continuous-time Markov chain (CTMC) and clamp the sequence data. Note we should have two CTMC objects in this model: one for the model of molecular evolution and one for the model of chromosome evolution.

```
dna_ctmc ~ dnPhyloCTMC(tree=phylogeny, Q=Q_mol, branchRates=1.0, type="DNA")
dna_ctmc.clamp(dna_seq)
```

3.3.4 MCMC and Summarizing Results

We set up the MCMC just as before, except here we need to add a file monitor to store the sampled trees.

```
monitors[2] = mnFile(filename="output/ChromEvol_joint.trees", printgen=10, phylogeny)
```

Summarizing the results of the MCMC analysis are a little different. First we will calculate the maximum a posteriori (MAP) tree.

```
treetrace = readAncestralStateTreeTrace("output/ChromEvol_joint.trees", treetype="non-
    clock")
map_tree = mapTree(treetrace, "output/ChromEvol_joint_map.tree")
```

Now we'll summarize the ancestral chromosome numbers over the MAP tree. Read in the ancestral state trace:

```
anc_state_trace = readAncestralStateTrace("output/ChromEvol_joint_states.log")
```

Finally, calculate the marginal ancestral states from the traces over the MAP tree. Note that this time we have to pass both the tree trace and the ancestral state trace to the `ancestralStateTree` function. Since we sampled a joint distribution of ancestral state histories and trees, we sampled some ancestral states for nodes that do not exist in the MAP tree. Therefore the ancestral state probabilities being calculated for the MAP tree are conditional to the probability of the node existing.

```
ancestralStateTree(map_tree, anc_state_trace, treetrace, "output/ChromEvol_joint_final
    .tree" burnin=0.25, reconstruction="marginal")
```

Like before, we can plot the results using the `RevGadgets` R package.

3.4 Associating Chromosome Evolution with Phenotype: BiChroM

```
```

3.5 Incorporating Cladogenetic and Anagenetic Chromosome Changes

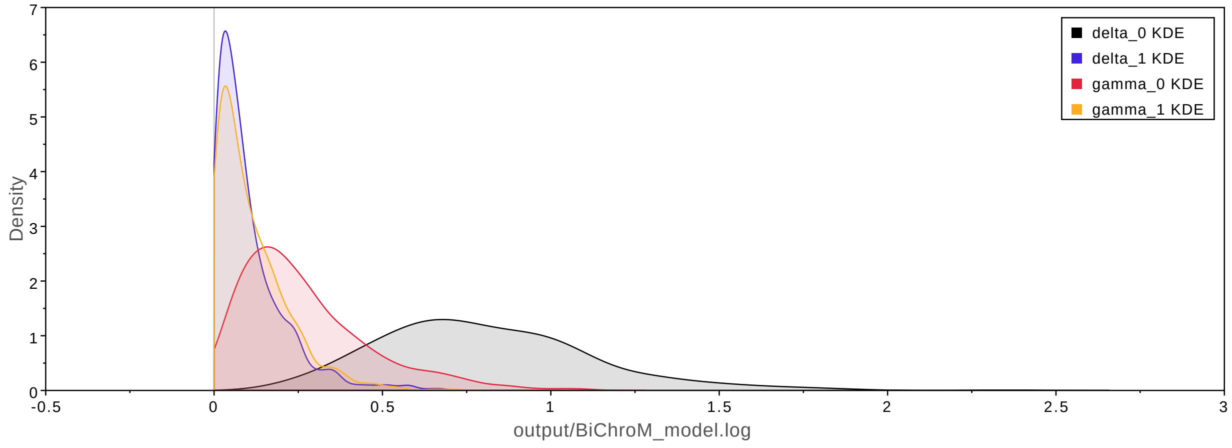

Figure 3: The rates of chromosome gains (`gamma`) and losses (`delta`) are higher for *Aristolochia* lineages with complex gynostemium subdivided into 5 to 24 lobes (state 0) compared to lineages with simple 3 lobed gynostemium (state 1).

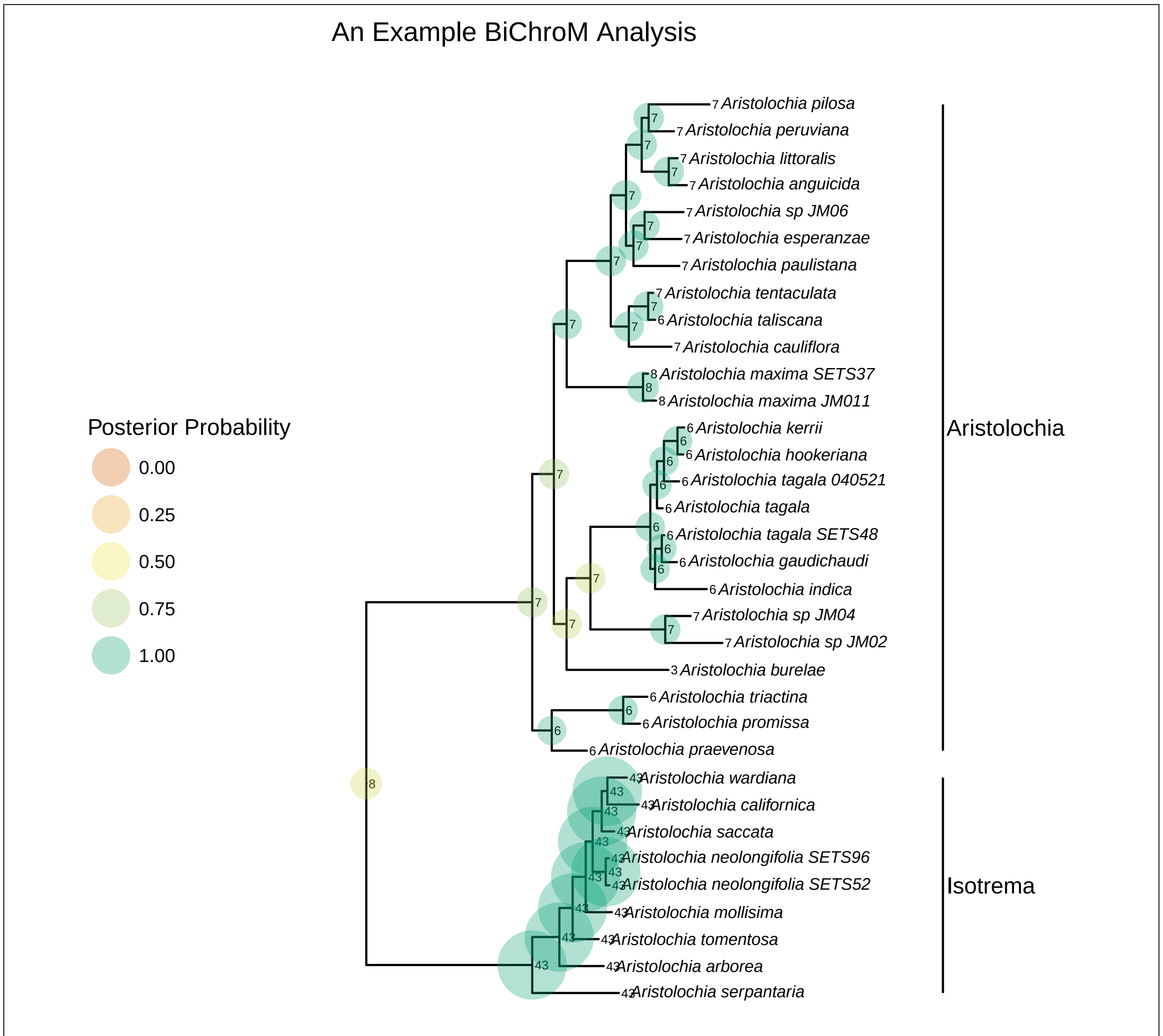

Figure 4: Maximum a posteriori estimates of ancestral chromosome number and gynostemium morphology for *Aristolochia*. States 1-26 represent the haploid number n of chromosome for lineages with gynostemium subdivided in 5 to 24 lobes. States 27-52 represent the haploid number $n + 27$ of chromosomes for lineages with simple 3 lobed gynostemium. The ancestral state for the common ancestor of all *Aristolochia* had a haploid number $n = 8$ and more complex 5 to 24 lobed gynostemium. An evolutionary reduction to 3 lobes is inferred in the lineage leading to the extant Isotrema clade.

References

Bollback, J. P. 2006. Simmap: stochastic character mapping of discrete traits on phylogenies. BMC bioinformatics 7:88.

Freyman, W. A. and S. Höhna. 2016. Cladogenetic and anagenetic models of chromosome number evolution:

a bayesian model averaging approach. bioRxiv Page 086629.

Mayrose, I., M. S. Barker, and S. P. Otto. 2010. Probabilistic models of chromosome number evolution and the inference of polyploidy. Systematic Biology 59:132–144.

Mayrose, I., S. H. Zhan, C. J. Rothfels, K. Magnuson-Ford, M. S. Barker, L. H. Rieseberg, and S. P. Otto. 2011. Recently formed polyploid plants diversify at lower rates. Science 333:1257–1257.

Ohi-Toma, T., T. Sugawara, H. Murata, S. Wanke, C. Neinhuis, and J. Murata. 2006. Molecular phylogeny of aristolochia sensu lato (aristolochiaceae) based on sequences of rbcl, matk, and phya genes, with special reference to differentiation of chromosome numbers. Systematic Botany 31:481–492.

Pires, J. C. and K. L. Hertweck. 2008. A renaissance of cytogenetics: Studies in polyploidy and chromosomal evolution. Annals of the Missouri Botanical Garden 95:275–281.

Revell, L. J. 2012. phytools: an r package for phylogenetic comparative biology (and other things). Methods in Ecology and Evolution 3:217–223.

Stebbins, G. L. 1971. Chromosomal evolution in higher plants. Edward Arnold Ltd., London.

Tank, D. C., J. M. Eastman, M. W. Pennell, P. S. Soltis, D. E. Soltis, C. E. Hinchliff, J. W. Brown, E. B. Sessa, and L. J. Harmon. 2015. Nested radiations and the pulse of angiosperm diversification: increased diversification rates often follow whole genome duplications. New Phytologist 207:454–467.

Zenil-Ferguson, R., J. M. Ponciano, and J. G. Burleigh. 2017. Testing the association of phenotypes with polyploidy: An example using herbaceous and woody eudicots. Evolution .

Version dated: May 3, 2017