Body Size Evolution in Monitor Lizards

Ian G. Brennan

08 November, 2019

Contents

Read Me **1**

Data Visualization **2**

Packages and Scripts **5**

Trait and Spatial Evolution of Australian *Varanus* **6**
Spatial Data Processing 6
Fitting Models of Trait Evolution 13

Building and Fitting Coevolutionary Models of Trait Evolution **21**
Coevolutionary Spatial Data Processing 21
Coevolutionary Model Fitting 30

Read Me

This markdown document will walk you through the data and code necessary to repeat the analyses of trait evolution found in our manuscript. All the files and code are available at the *GitHub Repository MonitorPhylogenomics*.

Start off by loading a few packages that we'll need along the way.

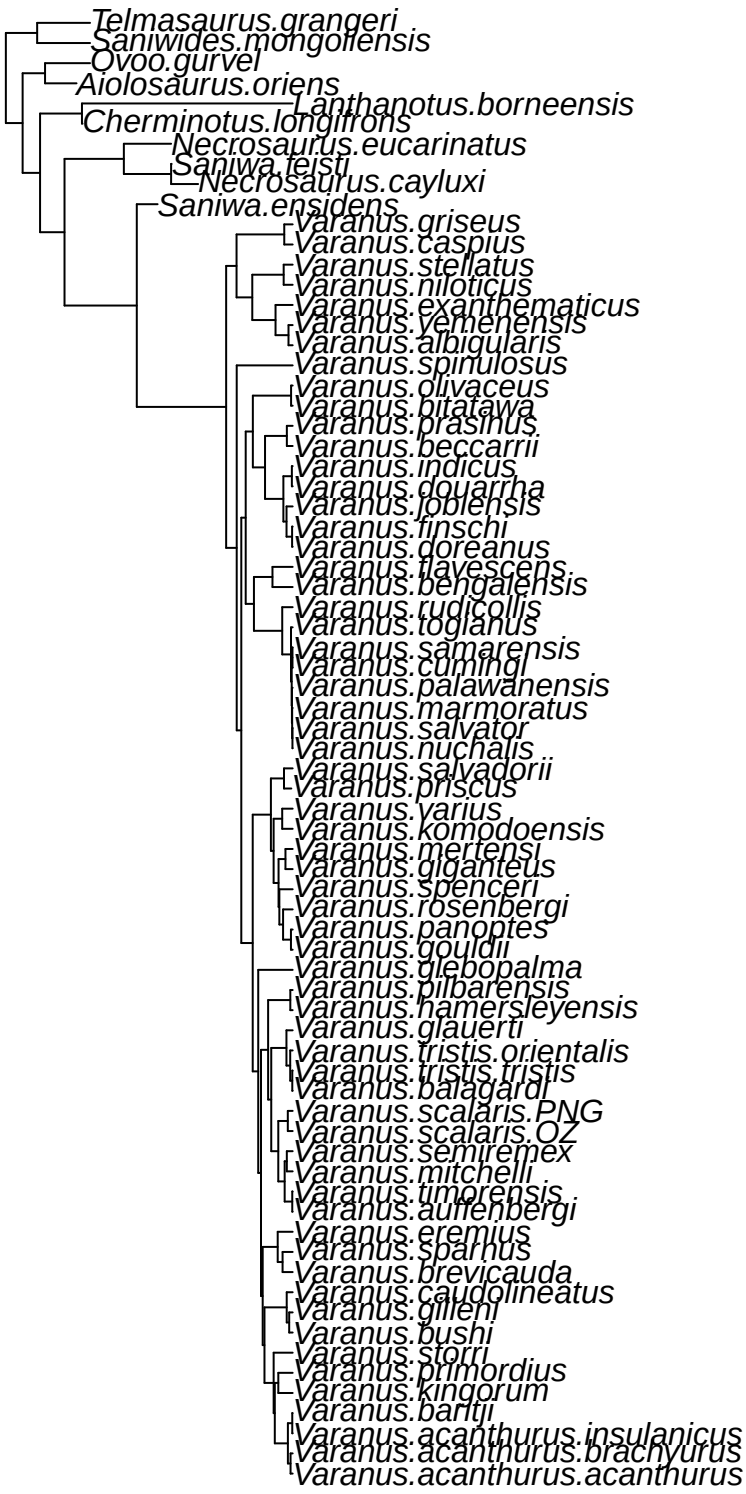
```
library(dplyr)
library(treeplyr)
library(RCurl)
library(phytools)
library(RColorBrewer)
# remember 'plyr' and 'dplyr' conflict, so don't load 'plyr'

#setwd("~/Documents/GitHub")
```

Data Visualization

Now read in and have a quick look at our data.

```
gtree <- read.tree("Varanidae_STRICT_HKY_270_con.newick");
plot.phylo(gtree)
```



```
alldata <- read.csv("All_Size_Data.csv", header=T)
head(alldata)
```

```
##     SVL Tail         Name_in_Tree Location     Group Habitat Status
## 1  90.0   NA Antechinomys.laniger          Marsupial    <NA> Extant
## 2  94.5   NA    Antechinus.agilis          Marsupial    <NA> Extant
## 3 134.5   NA    Antechinus.bellus          Marsupial    <NA> Extant
## 4 129.0   NA  Antechinus.flavipes          Marsupial    <NA> Extant
## 5 133.0   NA   Antechinus.godmani          Marsupial    <NA> Extant
## 6 151.0   NA       Antechinus.leo          Marsupial    <NA> Extant
```

Use treeplyr to combine the data, then remove any missing

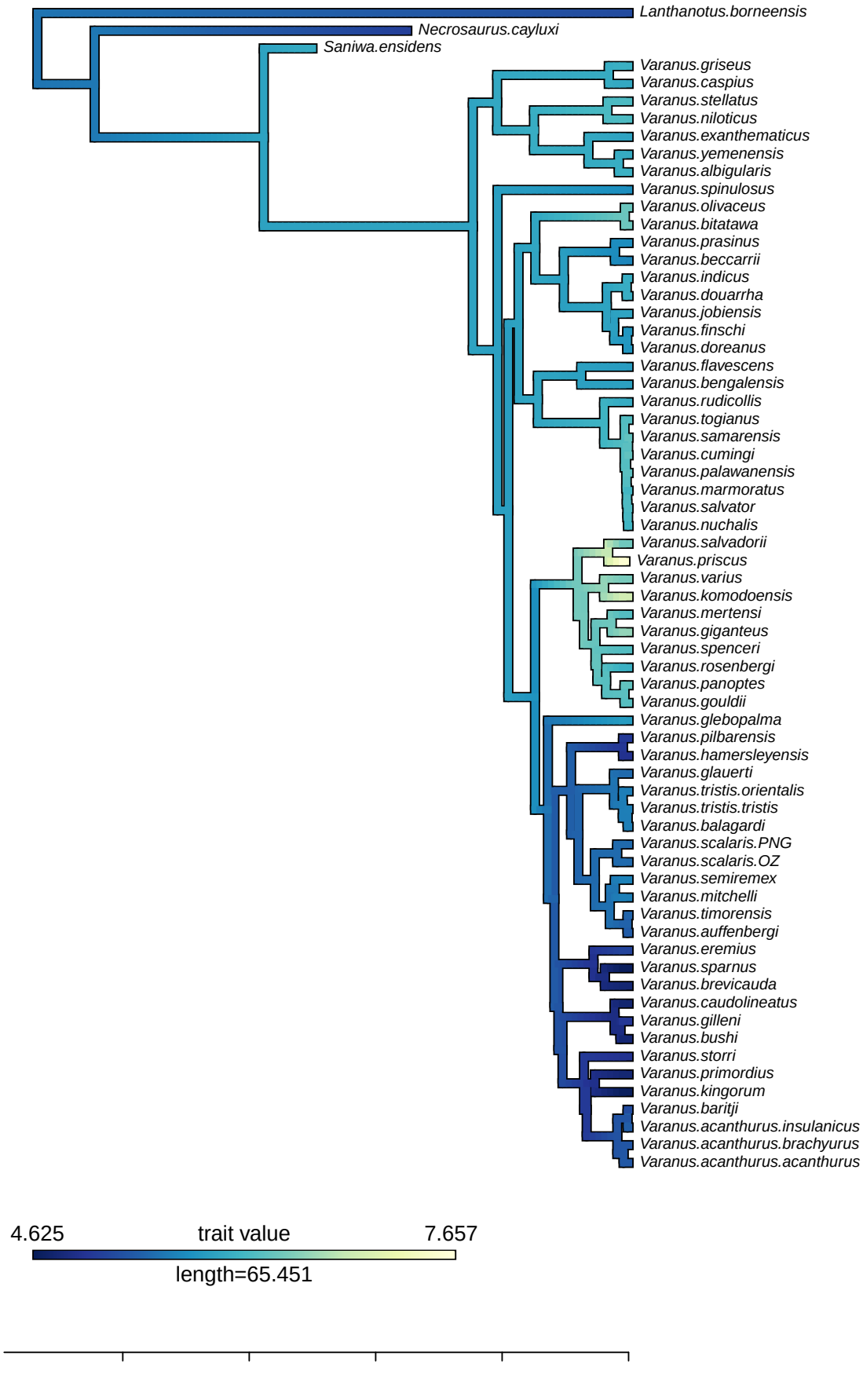
```
goanna <- make.treedata(gtree, alldata); # summary(goanna)
gf <- filter(goanna, is.na(Tail)==FALSE, is.na(SVL)==FALSE);
```

treeplyr is great because we can use all the tidyverse terms we already love.
Log-transform the SVL data, and make it a new column

```
gf <- mutate(gf, logSVL = log(gf$dat$SVL))
```

Start visualizing the trait using 'contMap' from 'phytools'

```
obj1 <- contMap(gf$phy, getVector(gf, logSVL), plot=FALSE, outline=F);
n<-length(obj1$cols);
obj1$cols[1:n] <- rev(colorRampPalette(brewer.pal(9, "YlGnBu"))(n));
plot(obj1, legend=0.7*max(nodeHeights(obj1$tree)),
     fsize=c(0.7,0.9), lwd=5, border=F); axisPhylo(1, backward=T)
```


Lanthanotus.borneensis
Necrosaurus.cayluxi
Saniwa.ensidens
Varanus.griseus
Varanus.caspius
Varanus.stellatus
Varanus.niloticus
Varanus.exanthematicus
Varanus.yemenensis
Varanus.albigularis
Varanus.spinulosus
Varanus.olivaceus
Varanus.bitatawa
Varanus.prasinus
Varanus.beccarrii
Varanus.indicus
Varanus.douarrha
Varanus.jobiensis
Varanus.finschi
Varanus.doreanus
Varanus.flavescens
Varanus.bengalensis
Varanus.rudicollis
Varanus.togianus
Varanus.samarensis
Varanus.cumingi
Varanus.palawanensis
Varanus.marmoratus
Varanus.salvator
Varanus.nuchalis
Varanus.salvadorii
Varanus.priscus
Varanus.varius
Varanus.komodoensis
Varanus.mertensi
Varanus.giganteus
Varanus.spenceri
Varanus.rosenbergi
Varanus.panoptes
Varanus.gouldii
Varanus.glebopalma
Varanus.pilbarensis
Varanus.hamersleyensis
Varanus.glauerti
Varanus.tristis.orientalis
Varanus.tristis.tristis
Varanus.balagardi
Varanus.scalaris.PNG
Varanus.scalaris.OZ
Varanus.semiremex
Varanus.mitchelli
Varanus.timorensis
Varanus.auffenbergi
Varanus.eremius
Varanus.sparnus
Varanus.brevicauda
Varanus.caudolineatus
Varanus.gilleni
Varanus.bushi
Varanus.storri
Varanus.primordius
Varanus.kingorum
Varanus.baritji
Varanus.acanthurus.insulanicus
Varanus.acanthurus.brachyurus
Varanus.acanthurus.acanthurus
4.625
trait value
7.657
length=65.451

Trim the data just down to Australian goannas

```
oz.g <- filter(gf, Location == "Australia" & Status == "Extant")
#summary(oz.g)
```

Packages and Scripts

We need to load a bunch of additional packages and custom scripts, I'll try and explain what they're for briefly.

```
library(parallel)
library(RPANDA)
library(deSolve)
library(rase)
```

This is a collection of additional functions for RPANDA, and includes the new models we'll use later.

```
source("~/Documents/GitHub/MonitorPhylogenomics/RPANDA_extras.R")
```

These functions make a *Geo Object* from spatial data and your *rase* output.

```
source("~/Documents/GitHub/MonitorPhylogenomics/CreateGeoObject_fromSP.R")
```

A collection of functions for extracting the AIC values and weights for a set of models.

```
source("~/Documents/GitHub/MonitorPhylogenomics/Calculate_AICs.R")
```

A function for plotting distribution maps for a set of taxa. This also translates spatial data into spatial geometries and OWin objects.

```
source("~/Documents/GitHub/MonitorPhylogenomics/plot.distmaps.R")
```

This function processes a *rase* output object and makes distribution objects for ancestral nodes. More on that later...

```
source("~/Documents/GitHub/MonitorPhylogenomics/process.rase.R")
```

The likelihood optimization can be hard given the number of parameters we're estimating, so I've created a function 'search.surface' that uses mclapply to fit the model a number of times with different starting parameters. It starts by creating sets of plausible starting parameters from across the surface, fits them and gives you the output either the best model fit, or all of the model fits. This function/method won't be necessary for simpler models like the BM or OU, and as implemented it won't work on models outside of the RPANDA framework.

```
source("~/Documents/GitHub/MonitorPhylogenomics/search.surface.R")
```

Trait and Spatial Evolution of Australian *Varanus*

We're going to turn our focus over to the Australian radiation of monitor lizards now, so we can load some existing files.

```
tutorial <- readRDS("~/Documents/GitHub/MonitorPhylogenomics/Goanna_Walkthrough.RDS")
# this includes a tree, size data, distribution data, and a processed rase object
names(tutorial)
```

```
## [1] "phy"               "size.data"         "distribution.data"
## [4] "rase.data"         "geo.object"
```

This should include our phylogeny of Australian monitor lizards (**phy**), corresponding body size information (**body.size**), a table of their occurrence records (**distribution.data**), a process *rase* file of spatial evolution (**rase.data**), and a geography object for use with *RPANDA* (**geo.object**).

Spatial Data Processing

Spatial Data at the Tips

Now we can plot the distributions of extant (tip) taxa.
plot.distmaps will loop through each taxon in the distribution dataframe (goanna.dist), and return a list three types of objects:
• **SpatialPoints** which have been made from the lat/longs.
• **ConvexHulls** which are distribution shape objects
• **OWin** objects which are another type of distribution shape object

```
head(tutorial$distribution.data)
```

```
##         Name_in_Tree  Latitude Longitude
## 1 Varanus.acanthurus -20.50569  140.6596
## 2 Varanus.acanthurus -20.48583  140.6087
## 3 Varanus.acanthurus -21.28000  140.5000
## 4 Varanus.acanthurus -20.70217  140.4910
## 5 Varanus.acanthurus -21.70000  140.4717
## 6 Varanus.acanthurus -18.07472  140.4508
```

```
tips <- plot.distmaps(tutorial$distribution.data,
                      new.directory = NULL,
                      point.width = 0.25)
```

```
## plotting Varanus.acanthurus , 1 of 31
## plotting Varanus.balagardi , 2 of 31
## plotting Varanus.baritji , 3 of 31
## plotting Varanus.brevicauda , 4 of 31
## plotting Varanus.bushi , 5 of 31
## plotting Varanus.caudolineatus , 6 of 31
## plotting Varanus.doreanus , 7 of 31
## plotting Varanus.eremius , 8 of 31
## plotting Varanus.giganteus , 9 of 31
```

```
## plotting Varanus.gilleni , 10 of 31
## plotting Varanus.glauerti , 11 of 31
## plotting Varanus.glebopalma , 12 of 31
## plotting Varanus.gouldii , 13 of 31
## plotting Varanus.indicus , 14 of 31
## plotting Varanus.kingorum , 15 of 31
## plotting Varanus.komodoensis , 16 of 31
## plotting Varanus.mertensi , 17 of 31
## plotting Varanus.mitchelli , 18 of 31
## plotting Varanus.panoptes , 19 of 31
## plotting Varanus.pilbarensis , 20 of 31
## plotting Varanus.prasinus , 21 of 31
## plotting Varanus.primordius , 22 of 31
## plotting Varanus.rosenbergi , 23 of 31
## plotting Varanus.scalaris , 24 of 31
## plotting Varanus.semiremex , 25 of 31
## plotting Varanus.sparnus , 26 of 31
## plotting Varanus.spenceri , 27 of 31
## plotting Varanus.storri , 28 of 31
## plotting Varanus.tristis.b , 29 of 31
## plotting Varanus.tristis.a , 30 of 31
## plotting Varanus.varius , 31 of 31
```

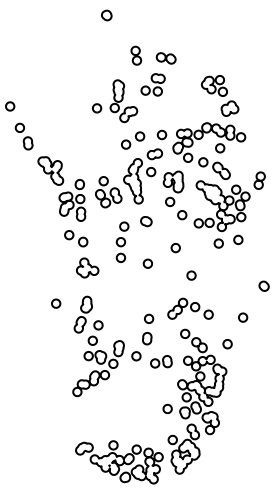
We can quickly look at what these things are. Each object is also indexed for each tip taxon.
First check out the *SpatialPoints* objects

```
head(tips$SpatialPoints$Varanus.eremius)
```

```
## class       : SpatialPoints
## features    : 1
## extent      : -26.96355, -26.96355, 140.9941, 140.9941  (xmin, xmax, ymin, ymax)
## crs         : NA
```

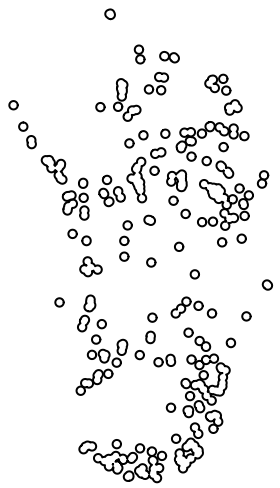
Next we can plot the shape of this distribution, which is an amalgamation of the point data with a buffer (of your choosing) around each point.

```
plot(tips$ConvexHulls$Varanus.eremius)
```



Finally, the *OWin* object for a given species should be identical to that species' *ConvexHull*

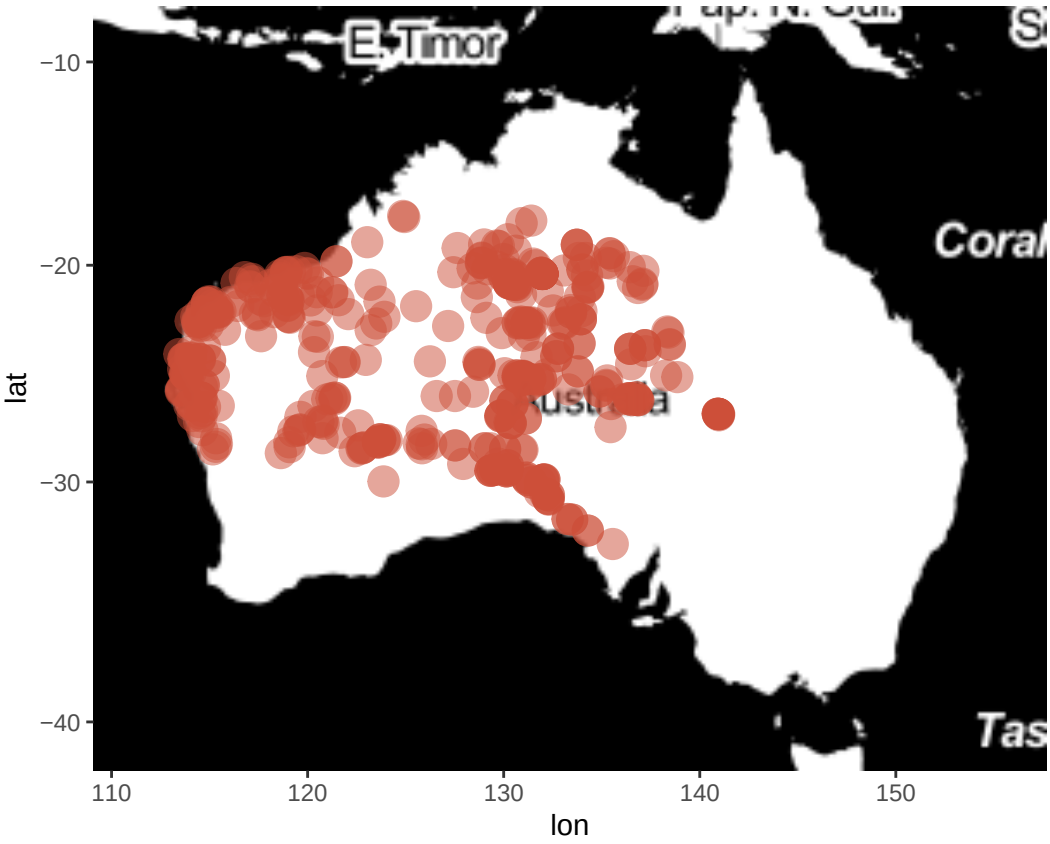
```
plot(tips$OWin$Varanus.eremius)
```



We can see what this actually looks like on a map too:

```
sbbox <- make_bbox(lon = tutorial$distribution.data$Longitude,
                   lat = tutorial$distribution.data$Latitude, f = .1)
sq_map <- get_stamenmap(bbox = sbbox, zoom = 3, maptype = "toner")

fortified.data <- fortify(tips$ConvexHulls$Varanus.eremius)
print((ggmap(sq_map)
      + geom_point(data = filter(tutorial$distribution.data,
                   Name_in_Tree == "Varanus.eremius"),
                   mapping = aes(x=Longitude, y=Latitude),
                   color="tomato3", size=5, alpha=0.5)))
```



Running *rase*

Ok, now we can move on. Quickly sort the data to make sure the order matches the tree appropriately

```
tree_poly <- name.poly(tips$OWin,
                       tutorial$phy,
                       poly.names = unique(tutorial$distribution.data$Name_in_Tree))
```

Now we can run the *rase* MCMC sampler. I'm not going to actually run it here because it would take too long, so instead we'll load an object I've already run.

```
res <- rase(tutorial$phy,
            tree_poly,
            niter=10000,
            logevery = 100)
```

If you have actually run it, then extract and plot the MCMC output to check for convergence.

```
resmc <- mcmc(res, start=(length(res[,1])*.2)) # remove 20% as burnin
par(mar=c(1,1,1,1))
plot(resmc)
```

Processing *rase* Outputs

If/when you do run *rase* you'll need to process the output so we can use it with the *process.rase* function. It will do pretty much the same thing that the *plot.distmaps* function did, but for ancestral nodes/distributions.

I'll explain how you use it here.

```
process.rase <- function(mcmc.object,
                         distribution,
                         new.directory=NULL,
                         remove.extralimital=NULL,
                         range.shape,
                         point.width=1)
```

Here are the basics:
• **mcmc.object**—is your *rase* mcmc output file
• **distribution**—is the data frame of distribution information
• **new.directory**—if you'd like to see the distributions of ancestral nodes, specify an output folder.
• **remove.extralimital**—because *rase* is a Brownian Motion dispersal process you might end up with samples in the ocean. We can remove these if we have a SHP file of the borders of our region, here Australia. Defaults to NULL if you don't have one.
• **range.shape**—if you set '*remove.extralimital*=TRUE', then the function expects the path to a .shp/x file.
• **point.width**—is the buffer size for your ancestral samples.

We've already loaded a processed *rase* object, so no need to make a new one.

```
# these are the objects in the rase list
names(tutorial$rase.data)
```

```
## [1] "DistData"      "SpatialPoints" "ConvexHulls"   "OWin"
```

Ancestral Spatial Data

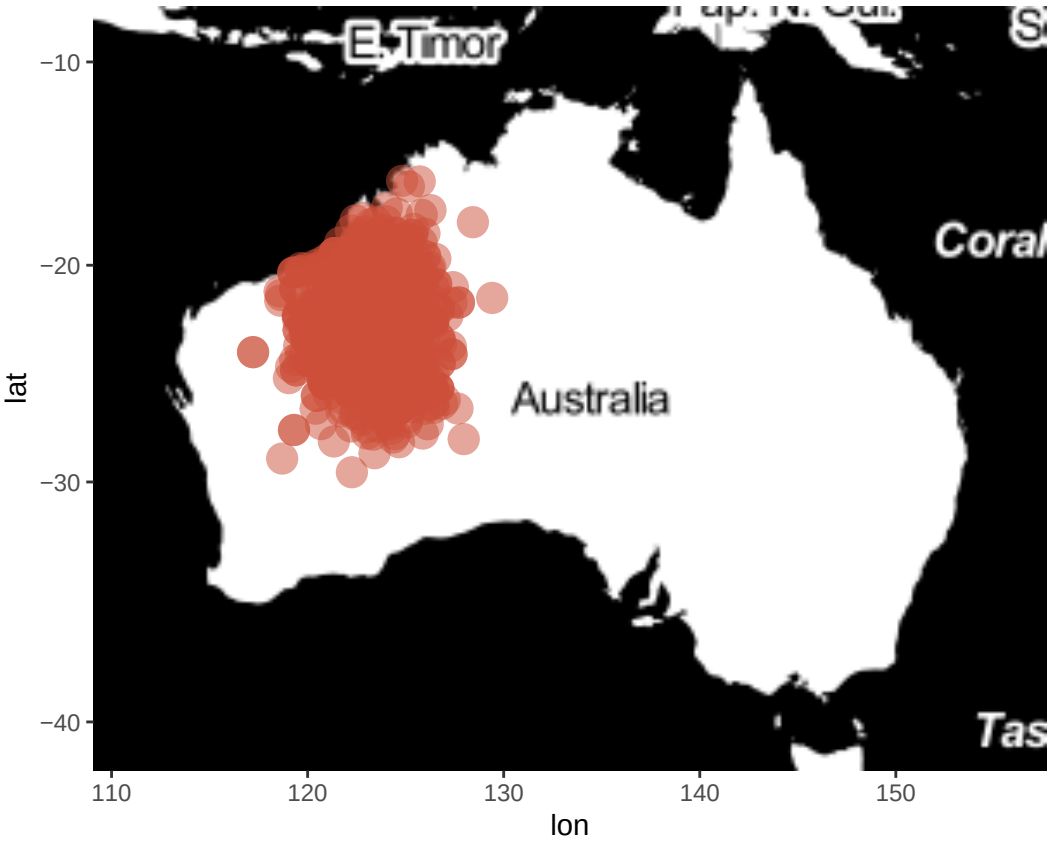
This object has all the same things as the object from *plot.distmaps*, in that you have *SpatialPoints*, *ConvexHulls*, and *OWin* shapes, as well as the raw distribution data *DistData*. The difference is that all of this information is for ancestral nodes (node numbers according to *ape*).
Quickly look at this information if you're interested. Our tree has 31 tips, so the root node should be n32.

```
# get the node number for the MRCA of V.gilleni and V.bushi
getMRCA(tutorial$phy, c("Varanus.gilleni", "Varanus.bushi"))
```

```
## [1] 43
```

We know it's node 43 ('n43' in my notation), so we can plot the distribution of this ancestor.

```
print((ggmap(sq_map)
       + geom_point(data = filter(tutorial$rase.data$DistData,
                                  Name_in_Tree == "n43"),
                    mapping = aes(x=Longitude, y=Latitude),
                    color="tomato3", size=5, alpha=0.5)))
```



Making *GeoObjects*

Great, we're ready to move on to making our *GeoObject*
In the script *CreateGeoObject_fromSP.R* there are actually two functions:
CreateGeoObject_SP works with a single tree, distribution data frame, and processed *rase* object.
CreateCoEvoGeoObject_SP does much the same, but for the co-evolutionary scenario with two trees, a distribution data frame, and two processed *rase* objects.

```
goanna.geo.object <- CreateGeoObject_SP(tutorial$phy,
                                        tutorial$distribution.data,
                                        point.width=0.25)
```

Again, instead of running that, we've already read in an existing file. If you decide to run it, the function shouldn't take more than a minute.

```
names(tutorial$geo.object)
```

```
## [1] "geography.object" "times"            "spans"
## [4] "name.matrix"
```

The *GeoObject* holds a lot of information, so it's not really useful to go through everything, but I'll try to make a short explanation so it isn't a total black box.
• **geography.object** is a list of the interaction matrices for all taxa that exist at a given time in the tree.

```
tutorial$geo.object$geography.object[[3]]
```

```
##                     Varanus.glebopalma .AAB .AAD .AAE
```

```
## Varanus.glebopalma                  1    1    1    1
## .AAB                                1    1    1    1
## .AAD                                1    1    1    1
## .AAE                                1    1    1    1
```

```
# this shows the interaction matrix at the third time point (third cladogenetic event)
```

• **times** are the occurrence times of the cladogenetic events, as time since the root (0).

```
tutorial$geo.object$times
```

```
##  [1]  0.000000  6.936469  9.839077 11.766469 12.616739 14.045740 15.130049
##  [8] 16.267532 16.382653 17.409324 18.781343 19.360130 20.707024 21.615989
## [15] 22.609906 22.877148 23.680182 23.768804 23.879711 23.899351 24.841652
## [22] 24.969668 25.334848 25.498787 25.755498 27.199174 28.789539 29.053123
## [29] 29.643692 29.731313 30.024877
```

```
# goanna.geo.object$times[[3]] is the time of the matrix above
```

• **spans** are the amount of time between each cladogenetic event.

```
tutorial$geo.object$spans
```

```
##  [1] 6.93646905 2.90260825 1.92739208 0.85026958 1.42900062 1.08430957
##  [7] 1.13748316 0.11512033 1.02667170 1.37201875 0.57878699 1.34689416
## [13] 0.90896442 0.99391703 0.26724258 0.80303362 0.08862244 0.11090706
## [19] 0.01963949 0.94230126 0.12801605 0.36518000 0.16393878 0.25671074
## [25] 1.44367643 1.59036531 0.26358380 0.59056876 0.08762114 0.29356387
```

• **name.matrix** is a matrix of all the internal node names and associated information. The left column are node/tip numbers. The center column is the taxon name of the edge which descends from that node, and the right column is the node that the edge ends at. Each node appears in the left column twice because it gives rise to two edges, which end at two different nodes (or tips). This is probably of no interest to anyone.

```
head(tutorial$geo.object$name.matrix)
```

```
##      [,1] [,2]                [,3]
## [1,] "32" ".AAA"              "33"
## [2,] "32" ".AAB"              "40"
## [3,] "33" ".AAC"              "34"
## [4,] "33" ".AAD"              "44"
## [5,] "34" ".AAE"              "35"
## [6,] "34" "Varanus.glebopalma" "0"
```

We now have a *GeoObject* that we can use for the geography-informed models. Next up we'll start fitting some models of trait evolution.

Fitting Models of Trait Evolution

Our full tree includes a few taxa which are not recognized as full species (*Varanus acanthurus brachyurus*, *V.a.insulanicus*, *et al.*), so we'll drop those from the tree for the trait evolution model fitting.

```
goanna <- make.treedata(tutorial$phy, tutorial$size.data); # summary(goanna)
goanna <- mutate(goanna, logSVL = log(Body_Length))
```

We can have a look at the *search.surface* function quickly to see what it does and how it works.

```
search.surface <- function(model, n.iter = 10, traits, n.proc = 8,
                           no.S=1, results=c("best", "all"), start.params=NULL)
```

The function lets us control a few things via the commands:
• **model**—the model of interest, you must have built this already
• **n.iter**—the number of model fittings you'd like completed, defaults to 10
• **traits**—the input traits for model fitting
• **n.proc**—number of processors. this function will fit the model in parallel.
• **no.S**— number of S parameters in the model, defaults to 1. if your model requires estimating/fitting more than 1 S parameter, say so
• **results**—would you like the function to report just the best fitting run, or all the results from each fit attempt

Now we need to build & fit a set of models just to the goanna data to compare with previous hypotheses about how body size variation evolved.
Let's start with the basics (BM, OU), then move into some more complex models.

Brownian Motion (BM)

Now the standard Brownian Motion (BM) model (Felsenstein, 1985)

```
modelBM <- createModel(goanna$phy,
                       keyword="BM")
```

```
show(modelBM)
```

```
## ******************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "BM"
## *** Parameters of the model : [1] "m0" "v0" "d" "sigma"
## *** Description : Brownian Motion model with linear drift.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving independently after
branching.
## dX_t = d dt + sigma dW_t
## *** Periods : the model is cut into 31 parts.
## For more details on the model, call : print(PhenotypicModel)
## ******************************************************************
```

```
fitBM <- fitTipData(modelBM,
                    treeplyr::getVector(goanna, logSVL),
                    GLSstyle=T)
show(fitBM)
```

Ornstein-Uhlenbeck (OU)

Next an Ornstein-Uhlenbeck (OU) model, which is really just BM with a single "adaptive optimum"

```
modelOU <- createModel(goanna$phy,
                       keyword="OU")
```

```
show(modelOU)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "OU"
## *** Parameters of the model : [1] "m0" "v0" "psi" "theta" "sigma"
## *** Description : Ornstein-Uhlenbeck model.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving independently after
branching.
## dX_t = psi(theta- X_t) dt + sigma dW_t
## *** Periods : the model is cut into 31 parts.
## For more details on the model, call : print(PhenotypicModel)
## ********************************************************************
```

```
fitOU <- fitTipData(modelOU,
                    treeplyr::getVector(goanna, logSVL),
                    GLSstyle=T)
show(fitOU)
```

Multi-Optimum OU (OUM)

Let's fit a model for goannas based on habitat partitioning (Collar et al. 2011).
This is a multi-OU model with different "adaptive optima" per habitat/niche type.

```
library(OUwie)
source("~/Documents/GitHub/MonitorPhylogenomics/make.OUwie.input_Script.R")
goanna.habitat <- make.OUwie.input(data=goanna$dat,
                                   regime=goanna$dat$Habitat,
                                   taxa=goanna$phy$tip.label,
                                   phy=goanna$phy,
                                   trait=goanna$dat$logSVL)
plot(goanna.habitat$regime.simm)
fitOUM <- OUwie(goanna.habitat$regime.simm,
                goanna.habitat$combined,
                model="OUM",
                simmap.tree=T)
fitOUM
```

Phenotypic Matching (PM)

Build the PM model, which estimates a single S parameter

```
modelPM <- createModel(goanna$phy,
                       keyword="PM")
```

```
show(modelPM)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "PM"
## *** Parameters of the model : [1] "m0" "v0" "theta" "psi" "S" "sigma"
## *** Description : Phenotype Matching model.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving then non-independtly
according to the Phenotype Matching expression.
## *** Periods : the model is cut into 31 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
fitPM <- fitTipData(modelPM,
                    treeplyr::getVector(goanna, logSVL),
                    GLSstyle=T)
show(fitPM)
```

The geography-informed models below require a processed rase object, luckily we already have one of those! ### Phenotypic Matching with Geography (PM_{geo}) Build the PM_{geo} model, which estimates a single S parameter, and includes geography

```
modelPM_geo <- createGeoModel(goanna$phy,
                              tutorial$geo.object,
                              keyword="PM+geo")
```

```
show(modelPM_geo)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "PM+geo"
## *** Parameters of the model : [1] "m0" "v0" "theta" "psi" "S" "sigma"
## *** Description : Phenotype Matching model with biogeography.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving then non-independtly
according to the Phenotype Matching expression.
## *** Periods : the model is cut into 31 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
fitPM_geo <- search.surface(modelPM_geo,
                            n.iter=8,
                            traits=treeplyr::getVector(goanna, logSVL),
                            n.proc=4,
                            no.S=1,
                            results="best") # this took ~1 min
show(fitPM_geo)
```

Phenotypic Matching without the OU process ($PMOU_{less}$)

Build the PM OU-less model, which estimates a single S, no alpha, and includes geography

```
modelPMOU_geo <- createGeoModel(goanna$phy,
                                tutorial$geo.object,
                                keyword="PMOU+geo")
```

```
show(modelPMOU_geo)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "PMOU+geo"
## *** Parameters of the model : [1] "m0" "v0" "S" "sigma"
## *** Description : Simplified Phenotype Matching model.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving then non-independtly
according to the Phenotype Matching expression, without the OU term.
## *** Periods : the model is cut into 31 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
fitPMOU_geo <- fitTipData(modelPMOU_geo,
               treeplyr::getVector(goanna, logSVL),
               GLSstyle=T)
show(fitPMOU_geo)
```

Summarizing Model Fit

If you've run the above models, and want to keep the results, you can do that here

```
model.list <- c("BM", "OU", "PM", "PM_geo", "PMOU_geo", "OUM")
fit.res <- lapply(model.list,
                  function(x) get(paste0("fit", x)));
names(fit.res) <- model.list
save.all.models <- paste0("~/Documents/GitHub/MonitorPhylogenomics/",
                          "Varanus_Empirical_ModelObjects.RData")
save(modelBM, fitBM,
     modelOU, fitOU,
     modelPM, fitPM,
     modelPM_geo, fitPM_geo,
     modelPMOU_geo, fitPMOU_geo,
     fitOUM, fit.res,
     file=save.all.models)
```

To save time, I haven't run these models here, so we'll read in an r data file with the models instead.

```
load("~/Documents/GitHub/MonitorPhylogenomics/Varanus_Empirical_ModelObjects.RData")
```

Summarize the model fits

```
multiphy.AIC(prefix="fit", phylo=goanna$phy,
             models=c("BM", "OU", "PM_geo", "PMOU_geo", "OUM"))
```

```
## $results
## logLik no.Param AIC AICc delta.AICc AICcWt
## PMOU_geo -6.503810 4 21.00762 22.54608 0.000000 0.69225324
## BM -7.963686 4 23.92737 25.46583 2.919753 0.16078614
## OU -7.367979 5 24.73596 27.13596 4.589877 0.06975670
## PM_geo -6.006353 6 24.01271 27.51271 4.966625 0.05777981
## OUM -7.096477 6 26.19295 29.69295 7.146873 0.01942411
##
## $best.model
## logLik no.Param AIC AICc delta.AICc AICcWt
```

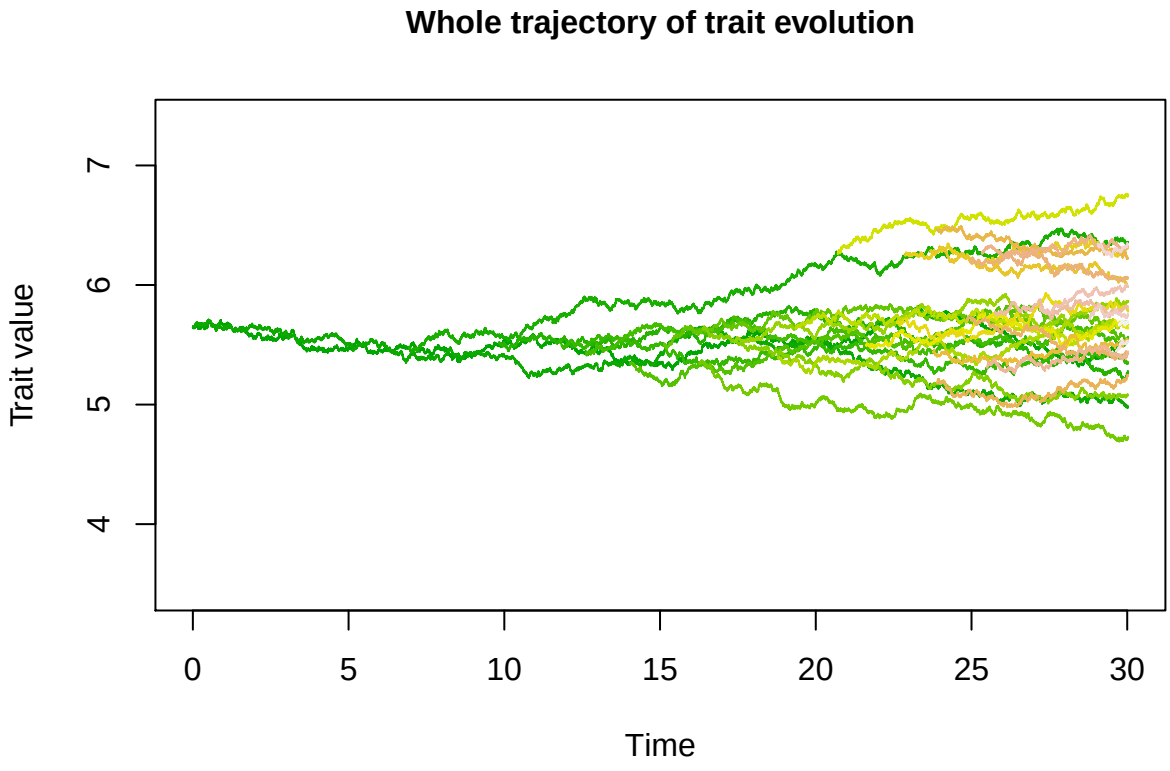
```
## PMOU_geo -6.50381 4 21.00762 22.54608 0 0.6922532
##
## $parameter.estimates
## m0 v0 S sigma
## 5.648863e+00 1.287049e-08 -4.020301e-02 7.506412e-02
##
## $model.description
## [1] "Simplified Phenotype Matching model.\nStarts with two lineages having
the same value X_0 ~ Normal(m0,v0).\nOne trait in each lineage, all lineages
evolving then non-independtly according to the Phenotype Matching expression,
without the OU term."
```

What we find is pretty strong support (70% AICc weight) for the geographically-informed Phenotypic Matching model (also known as Matching Competition), without the OU parameter of the original model. That's interesting because the interaction parameter S is negative, suggesting that body size of Australian monitor lizards has evolved as result of lineage interactions.

If you wanted to simulate some data to see what it looks like under the inferred parameters. Yours might look a little different from the one in this PDF, no worries.

```
simulateTipData(modelPMOU_geo, fitPMOU_geo$inferredParams, method=2)
```

```
## *** Simulation of tip trait values ***
## Simulates step-by-step the whole trajectory, plots it, and returns tip data.
```




```
## Computation time : 1.960836 secs
```

```
##                          [,1]
## Varanus.acanthurus    5.538582
## Varanus.prasinus      4.982906
## Varanus.giganteus     6.351320
## Varanus.glebopalma    5.275955
```

```
## Varanus.semiremex     5.354585
## Varanus.brevicauda    5.567002
## Varanus.bushi         5.552166
## Varanus.balagardi     5.687102
## Varanus.pilbarensis   4.730463
## Varanus.doreanus      5.075709
## Varanus.kingorum      5.865409
## Varanus.scalaris      5.690750
## Varanus.storri        5.676833
## Varanus.komodoensis   6.757443
## Varanus.eremius       5.638671
## Varanus.primordius    5.776869
## Varanus.gouldii       6.315999
## Varanus.spenceri      5.986733
## Varanus.sparnus       5.541467
## Varanus.varius        6.220352
## Varanus.indicus       5.239479
## Varanus.glauerti      5.425278
## Varanus.rosenbergi    6.053572
## Varanus.mertensi      6.317327
## Varanus.mitchelli     5.437683
## Varanus.caudolineatus 5.813990
## Varanus.gilleni       5.988605
## Varanus.tristis.b     5.756668
## Varanus.panoptes      6.335370
## Varanus.baritji       5.529475
## Varanus.tristis.a     5.681471
```

Model Recoverability

We could also quickly check that the inferred params result in simulated data that can recover the correct model:

```
# simulate under PMOUgeo model
testPMOUg <- simulateTipData(modelPMOU_geo, fitPMOU_geo$inferredParams, method=1)
head(testPMOUg)
# create and fit the BM model to the simulated data
test_modelBM <- createModel(goanna$phy,
                            keyword="BM")
test_fitBM <- fitTipData(test_modelBM,
                         testPMOUg,
                         GLSstyle=T)
show(test_fitBM)

# create and fit the PMOUgeo model to the simulated data
test_modelPMOU_geo <- createGeoModel(goanna$phy,
                                     tutorial$geo.object,
                                     keyword="PMOU+geo")
test_fitPMOU_geo <- fitTipData(test_modelPMOU_geo, testPMOUg, GLSstyle=T)
show(test_fitPMOU_geo)

# Summarize the model fits
multiphy.AIC(prefix="test_fit", goanna$phy, c("BM", "PMOU_geo"))
```

We see that from simulated data the PMOU_geo model is definitely distinguishable from Brownian Motion (99.8% AICc weight for the data I simulated).

Reminder: you can check the composition of a model using '@', e.g.:

```
modelPM_geo@aAGamma(3, fitPM_geo$inferredParams);
```

```
## $a
## function (t)
## return(params[3] * params[4] * vectorU)
## <bytecode: 0x7fec35ced8c8>
## <environment: 0x7fec3bd79430>
##
## $A
##                              .AAE          .AAB          .AAD
## .AAE                -0.006791909  0.029934810  0.029934810
## .AAB                 0.029934810 -0.006791909  0.029934810
## .AAD                 0.029934810  0.029934810 -0.006791909
## Varanus.glebopalma  0.029934810  0.029934810  0.029934810
##                     Varanus.glebopalma
## .AAE                         0.029934810
## .AAB                         0.029934810
## .AAD                         0.029934810
## Varanus.glebopalma          -0.006791909
##
## $Gamma
## function (t)
## return(params[6] * diag(vectorU))
## <bytecode: 0x7fec35cede40>
## <environment: 0x7fec3bd79430>
##
## $u
## [1] 1 1 1 1
##
## $OU
## [1] TRUE
```

And see what the interaction matrices would look like under estimated parameters:

```
modelPM_geo@aAGamma(3, fitPM_geo$inferredParams)$A / fitPM_geo$inferredParams[5]
```

```
##                            .AAE       .AAB       .AAD Varanus.glebopalma
## .AAE                  0.0567225 -0.2500000 -0.2500000         -0.2500000
## .AAB                 -0.2500000  0.0567225 -0.2500000         -0.2500000
## .AAD                 -0.2500000 -0.2500000  0.0567225         -0.2500000
## Varanus.glebopalma -0.2500000 -0.2500000 -0.2500000          0.0567225
```

Building and Fitting Coevolutionary Models of Trait Evolution

---

In this manuscript we also further developed the methods of Manceau et al. (201X) that allow the users to account for the influence of two clades on trait evolution. This can be imagined as phenotypic matching like in some butterflies and flowers (positive S) or as phenotypic competition like is expected in local organismal assemblages (negative S).

In general the steps follow the same process as with one clade. We need to: 1. Read in trees, trait, and distributional data. 2. Run *rase* to get ancestral distributions. 3. Determine overlapping taxa and build interaction matrices. 4. Fit models of trait evolution.

We already have our monitor lizard tree and data, but we'll start somewhere near scratch again, in case you've just jumped to this point.

```
coevo <- readRDS("~/Documents/GitHub/MonitorPhylogenomics/CoEvo_Walkthrough.RDS")
names(coevo)
```

```
## [1] "goanna.tree"            "marsupial.tree"
## [3] "size.data"              "goanna.distribution"
## [5] "marsupial.distribution" "goanna.rase"
## [7] "marsupial.rase"         "joint.geo.object"
```

This includes two tree files (**goanna.tree**, **marsupial.tree**), corresponding body size information (**size.data**), occurrence records for both clades (**goanna.distribution**, **marsupial.distribution**), processed *rase* objects (**goanna.rase**, **marsupial.rase**), and a joint geography object for use with *RPANDA* (**joint.geo.object**).

Coevolutionary Spatial Data Processing

Spatial Data at the Tips

Now we can plot the distributions of extant (tip) taxa.
plot.distmaps will loop through each taxon in the distribution dataframes, and return a list three types of objects:
- **SpatialPoints** which have been made from the lat/longs.
- **ConvexHulls** which are distribution shape objects
- **OWin** objects which are another type of distribution shape object

```
head(coevo$marsupial.distribution)
```

```
##           Name_in_Tree  Latitude Longitude
## 1 Antechinomys.laniger -28.76670  114.6167
## 2 Antechinomys.laniger -25.62306  114.6942
## 3 Antechinomys.laniger -27.81722  115.6233
## 4 Antechinomys.laniger -27.08333  116.1667
## 5 Antechinomys.laniger -28.75556  116.2667
## 6 Antechinomys.laniger -31.76667  116.3833
```

```
marsupial.tips <- plot.distmaps(coevo$marsupial.distribution,
                                new.directory = NULL,
                                point.width = 0.25)
```

```
## plotting Antechinomys.laniger , 1 of 62
## plotting Antechinus.agilis , 2 of 62
## plotting Antechinus.bellus , 3 of 62
## plotting Antechinus.flavipes , 4 of 62
## plotting Antechinus.godmani , 5 of 62
## plotting Antechinus.leo , 6 of 62
## plotting Antechinus.minimus , 7 of 62
## plotting Antechinus.sp. , 8 of 62
## plotting Antechinus.stuartii , 9 of 62
## plotting Antechinus.swainsonii , 10 of 62
## plotting Dasykaluta.rosamondae , 11 of 62
## plotting Dasyuroides.byrnei , 12 of 62
## plotting Dasyurus.geoffroii , 13 of 62
## plotting Dasyurus.hallucatus , 14 of 62
## plotting Dasyurus.maculatus , 15 of 62
## plotting Dasyurus.viverrinus , 16 of 62
## plotting Echymipera.rufescens , 17 of 62
## plotting Isoodon.auratus , 18 of 62
## plotting Isoodon.macrourus , 19 of 62
## plotting Isoodon.obesulus , 20 of 62
## plotting Macrotis.lagotis , 21 of 62
## plotting Ningaui.ridei , 22 of 62
## plotting Ningaui.timealeyi , 23 of 62
## plotting Ningaui.yvonnae , 24 of 62
## plotting Parantechinus.apicalis , 25 of 62
## plotting Perameles.bougainville , 26 of 62
## plotting Perameles.gunnii , 27 of 62
## plotting Perameles.nasuta , 28 of 62
## plotting Phascogale.calura , 29 of 62
## plotting Phascogale.tapoatafa , 30 of 62
## plotting Planigale.gilesi , 31 of 62
## plotting Planigale.ingrami , 32 of 62
## plotting Planigale.maculata , 33 of 62
## plotting Planigale.tenuirostris , 34 of 62
## plotting Pseudantechinus.bilarni , 35 of 62
## plotting Pseudantechinus.macdonnellensis , 36 of 62
## plotting Pseudantechinus.mimulus , 37 of 62
## plotting Pseudantechinus.ningbing , 38 of 62
## plotting Pseudantechinus.roryi , 39 of 62
## plotting Pseudantechinus.woolleyae , 40 of 62
## plotting Sarcophilus.harrisii , 41 of 62
## plotting Sminthopsis.aitkeni , 42 of 62
## plotting Sminthopsis.archeri , 43 of 62
## plotting Sminthopsis.bindi , 44 of 62
## plotting Sminthopsis.butleri , 45 of 62
## plotting Sminthopsis.crassicaudata , 46 of 62
## plotting Sminthopsis.dolichura , 47 of 62
## plotting Sminthopsis.douglasi , 48 of 62
## plotting Sminthopsis.gilberti , 49 of 62
## plotting Sminthopsis.granulipes , 50 of 62
## plotting Sminthopsis.griseoventer , 51 of 62
## plotting Sminthopsis.hirtipes , 52 of 62
## plotting Sminthopsis.leucopus , 53 of 62
## plotting Sminthopsis.longicaudata , 54 of 62
```

```
## plotting Sminthopsis.froggatti , 55 of 62
## plotting Sminthopsis.macroura , 56 of 62
## plotting Sminthopsis.murina , 57 of 62
## plotting Sminthopsis.ooldea , 58 of 62
## plotting Sminthopsis.psammophila , 59 of 62
## plotting Sminthopsis.virginiae , 60 of 62
## plotting Sminthopsis.youngsoni , 61 of 62
## plotting Thylacinus.cynocephalus , 62 of 62
```

To speed this all up I won't do it for the goannas, but I'll leave this here.

```
head(coevo$goanna.distribution)
goanna.tips <- plot.distmaps(coevo$goanna.distribution,
                             new.directory = NULL,
                             point.width = 0.25)
```

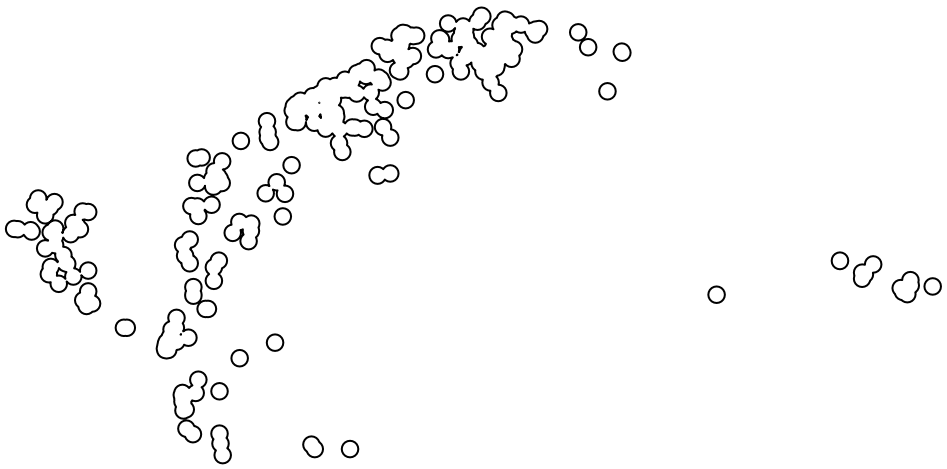
We can quickly look at what these things are. Each object is also indexed for each tip taxon.
First check out the *SpatialPoints* objects

```
head(marsupial.tips$SpatialPoints$Dasyurus.maculatus)
```

```
## class       : SpatialPoints
## features    : 1
## extent      : -36.9333, -36.9333, 140.15, 140.15  (xmin, xmax, ymin, ymax)
## crs         : NA
```

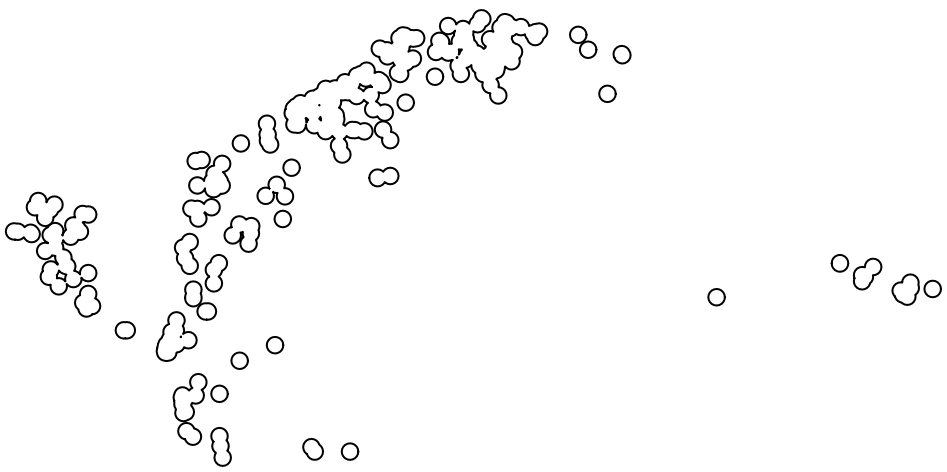
Next we can plot the shape of this distribution, which is an amalgamation of the point data with a buffer (of your choosing) around each point.

```
plot(marsupial.tips$ConvexHulls$Dasyurus.maculatus)
```



Finally, the *OWin* object for a given species should be identical to that species' *ConvexHull*

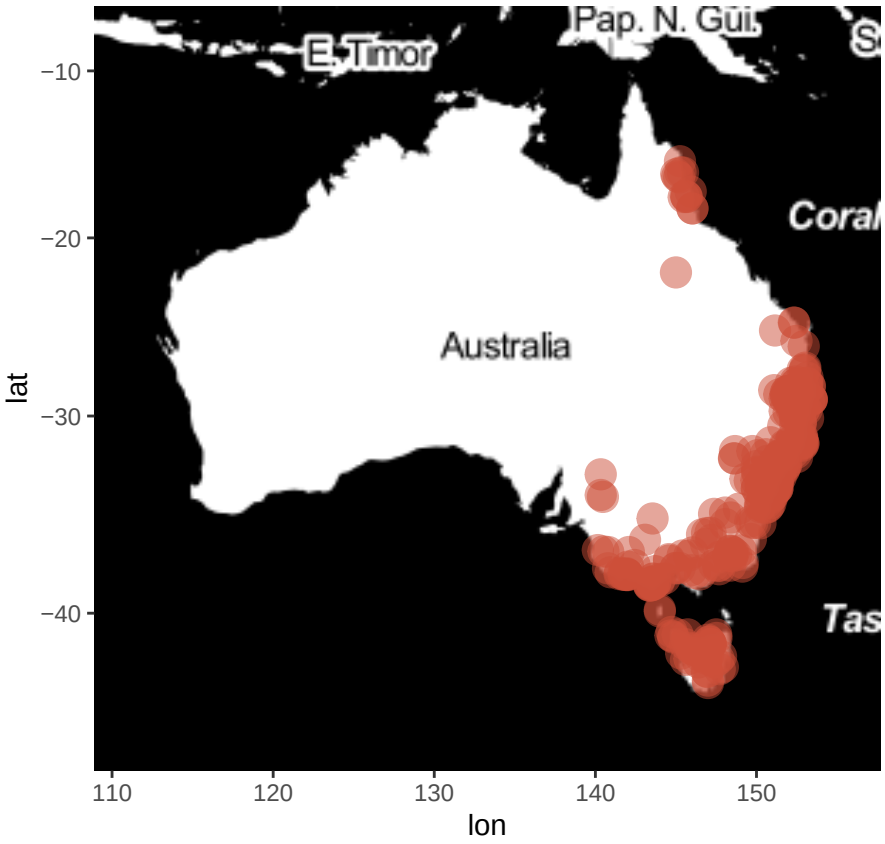
```
plot(marsupial.tips$OWin$Dasyurus.maculatus)
```



We can see what this actually looks like on a map too:

```
sbbox <- make_bbox(lon = coevo$marsupial.distribution$Longitude,
                   lat = coevo$marsupial.distribution$Latitude, f = .1)
sq_map <- get_stamenmap(bbox = sbbox, zoom = 3, maptype = "toner")

fortified.data <- fortify(marsupial.tips$ConvexHulls$Dasyurus.maculatus)
print((ggmap(sq_map)
      + geom_point(data = filter(coevo$marsupial.distribution,
                   Name_in_Tree == "Dasyurus.maculatus"),
                   mapping = aes(x=Longitude, y=Latitude),
                   color="tomato3", size=5, alpha=0.5)))
```



Running *rase*

Ok, now we can move on. Quickly sort the data to make sure the order matches the tree appropriately

```
tree_poly <- name.poly(marsupial.tips$OWin,
                       coevo$marsupial.tree,
                       poly.names = unique(coevo$marsupial.distribution$Name_in_Tree))
```

Now we can run the *rase* MCMC sampler. I'm not going to actually run it here because it would take too long, so instead we'll load an object I've already run.

```
res <- rase(coevo$marsupial.tree,
            tree_poly,
            niter=10000,
            logevery = 100)
```

If you have actually run it, then extract and plot the MCMC output to check for convergence.

```
resmc <- mcmc(res, start=(length(res[,1])*.2)) # remove 20% as burnin
par(mar=c(1,1,1,1))
plot(resmc)
```

If/when you do run *rase* you'll need to process the output so we can use it with the *process.rase* function. It will do pretty much the same thing that the *plot.distmaps* function did, but for ancestral nodes/distributions. I'll explain how you use it here.

```
process.rase <- function(mcmc.object,
                         distribution,
                         new.directory=NULL,
                         remove.extralimital=NULL,
                         range.shape,
                         point.width=1)
```

Here are the basics:
• **mcmc.object**—is your *rase* mcmc output file
• **distribution**—is the data frame of distribution information
• **new.directory**—if you'd like to see the distributions of ancestral nodes, specify an output folder.
• **remove.extralimital**—because *rase* is a Brownian Motion dispersal process you might end up with samples in the ocean. We can remove these if we have a SHP file of the borders of our region, here Australia. Defaults to NULL if you don't have one.
• **range.shape**—if you set '*remove.extralimital*=TRUE', then the function expects the path to a .shp/x file.
• **point.width**—is the buffer size for your ancestral samples.

We've already loaded a processed *rase* object for the goannas, so no need to make a new one.

```
# these are the objects in the rase list
names(coevo$marsupial.rase)
```

```
## [1] "DistData"      "SpatialPoints" "ConvexHulls"   "OWin"
```

Ancestral Spatial Data

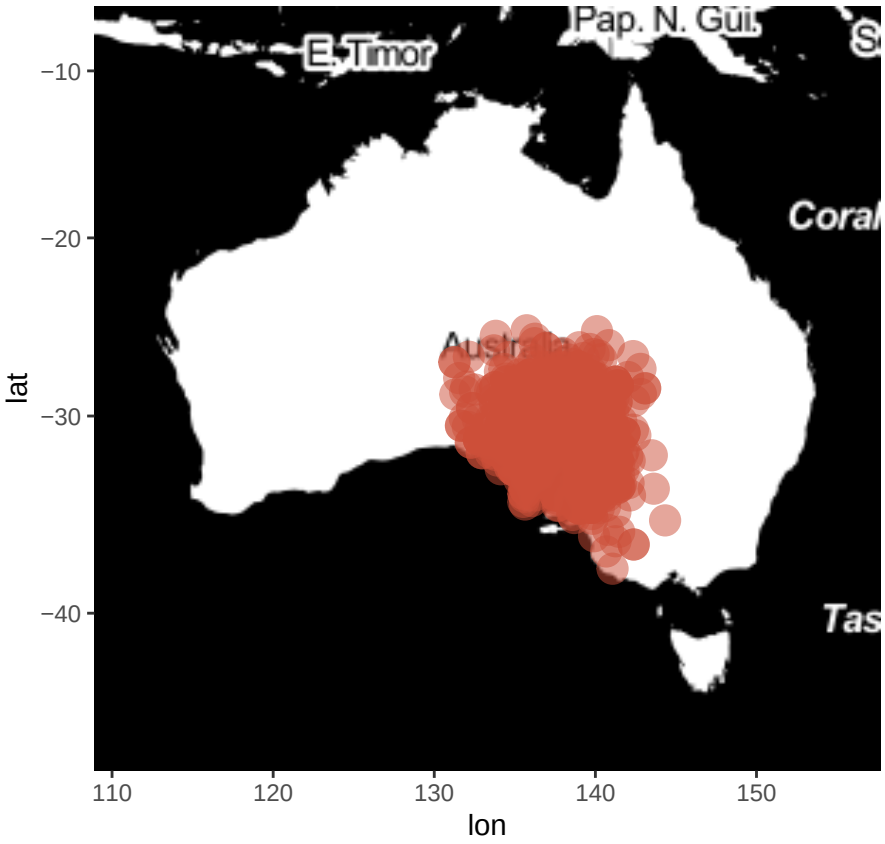
This object has all the same things as the object from *plot.distmaps*, in that you have *SpatialPoints*, *ConvexHulls*, and *OWin* shapes, as well as the raw distribution data *DistData*. The difference is that all of this information is for ancestral nodes (node numbers according to *ape*).
Quickly look at this information if you're interested. Our tree has 31 tips, so the root node should be n32.

```
# get the node number for the MRCA of two quolls D.maculatus and D.viverrinus
getMRCA(coevo$marsupial.tree, c("Dasyurus.maculatus", "Dasyurus.viverrinus"))
```

```
## [1] 115
```

We know it's node 115 ('n115' in my notation), so we can plot the distribution of this ancestor.

```
print((ggmap(sq_map)
       + geom_point(data = filter(coevo$marsupial.rase$DistData,
                                  Name_in_Tree == "n115"),
                    mapping = aes(x=Longitude, y=Latitude),
                    color="tomato3", size=5, alpha=0.5)))
```



Making *CoEvoGeoObjects*

Great, we're ready to move on to making our *CoEvoGeoObject*
In the script *CreateGeoObject_fromSP.R* there are actually two functions:
CreateGeoObject_SP works with a single tree, distribution data frame, and processed *rase* object.
CreateCoEvoGeoObject_SP does much the same, but for the co-evolutionary scenario with two trees, a distribution data frame, and two processed *rase* objects. We'll use the second one here because we're working with two trees.

```
coevo.dist <- rbind(coevo$goanna.distribution, coevo$marsupial.distribution)

joint.geo.object <- CreateCoEvoGeoObject_SP(phy1=coevo$goanna.tree,
                                            phy2=coevo$marsupial.tree,
                                            map=coevo.dist,
                                            rase.obj1=coevo$goanna.rase,
                                            rase.obj2=coevo$dasyurid.rase,
                                            point.width=0.25)
```

Again, instead of running that, we've already read in an existing file. If you decide to run it, the function shouldn't take more than a minute.

```
names(coevo$joint.geo.object)
```

```
## [1] "geography.object" "times"            "spans"
## [4] "startingTimes"    "endTimes"
```

The *CoEvoGeoObject* holds a lot of information, so it's not really useful to go through everything, but I'll try to make a short explanation so it isn't a total black box.

• **geography.object** is a list of the interaction matrices for all taxa that exist at a given time in the tree.

```
coevo$joint.geo.object$geography.object[[3]]
```

```
##                          .AAC Thylacinus.cynocephalus .AAB .ACJ .ACK
## .AAC                        1                        1    1    1    1
## Thylacinus.cynocephalus     1                        1    1    1    1
## .AAB                        1                        1    1    1    1
## .ACJ                        1                        1    1    1    1
## .ACK                        1                        1    1    1    1
```

```
# this shows the interaction matrix at the third time point (third cladogenetic event)
```

• **times** are the occurrence times of the cladogenetic events, as time since the root (0).

```
coevo$joint.geo.object$times
```

```
##  [1]  0.00000 15.21714 17.27353 22.15361 22.58436 23.38706 25.05622
##  [8] 26.98361 27.83388 28.37720 28.70770 29.26288 30.34719 31.48467
## [15] 31.59979 31.84629 32.62647 32.66434 32.90066 33.27401 33.62018
## [22] 33.99848 34.20244 34.24645 34.37417 34.57727 35.11213 35.29442
## [29] 35.29806 35.34686 35.75646 35.92417 36.14921 36.24887 36.25303
## [36] 36.61034 36.83313 37.26644 37.27494 37.47381 37.73709 37.75295
## [43] 37.82705 38.09429 38.50951 38.53092 38.58034 38.79270 38.81964
## [50] 38.89732 38.90005 38.98595 39.09685 39.10717 39.11649 39.40590
## [57] 39.64926 39.65153 39.74652 39.92211 39.95703 40.02344 40.04657
## [64] 40.05879 40.10314 40.18681 40.55199 40.65216 40.68454 40.71593
## [71] 40.74244 40.91928 40.97264 41.14217 41.17334 41.18834 41.59870
## [78] 41.67885 41.71021 41.97283 42.27931 42.39674 42.41631 42.80926
## [85] 43.03377 43.17359 43.55509 44.00668 44.27026 44.86083 44.94845
## [92] 45.24202
```

```
# goanna.geo.object$times[[3]] is the time of the matrix above
```

• **spans** are the amount of time between each cladogenetic event.

```
coevo$joint.geo.object$spans
```

```
##  [1] 15.217141  2.056389  4.880080  0.430746  0.802705  1.669157  1.927393
##  [8]  0.850269  0.543321  0.330496  0.555184  1.084309  1.137484  0.115120
## [15]  0.246499  0.780173  0.037878  0.236317  0.373347  0.346175  0.378301
## [22]  0.203955  0.044015  0.127717  0.203100  0.534860  0.182284  0.003647
## [29]  0.048795  0.409605  0.167703  0.225041  0.099666  0.004153  0.357317
## [36]  0.222788  0.433311  0.008502  0.198869  0.263278  0.015861  0.074096
## [43]  0.267242  0.415221  0.021409  0.049419  0.212362  0.026936  0.077687
## [50]  0.002725  0.085898  0.110907  0.010317  0.009322  0.289407  0.243357
## [57]  0.002277  0.094992  0.175582  0.034926  0.066410  0.023131  0.012219
## [64]  0.044345  0.083671  0.365180  0.100173  0.032379  0.031387  0.026510
## [71]  0.176840  0.053361  0.169529  0.031171  0.014999  0.410365  0.080146
## [78]  0.031365  0.262615  0.306481  0.117429  0.019576  0.392948  0.224511
## [85]  0.139819  0.381499  0.451589  0.263583  0.590569  0.087621  0.293564
```

• **startingTimes**

```
head(coevo$joint.geo.object$startingTimes)
```

```
## [1]  0.00000 17.27353 23.38706 28.37720 31.84629 32.90066
```

• **endTimes**

```
head(coevo$joint.geo.object$endTimes)
```

```
## [1] 17.27353 23.38706 28.37720 31.84629 32.90066 33.62018
```

We now have a *CoEvoGeoObject* that we can use for the geography-informed models. So, we'll start fitting some models of trait evolution.

Coevolutionary Model Fitting

Combine the trait data for both phylogenies

```
joint.traits <- log(coevo$size.data$Body_Length)
names(joint.traits) <- coevo$size.data$Name_in_Tree
head(joint.traits)
```

```
## Antechinomys.laniger    Antechinus.agilis    Antechinus.bellus
##             4.499810             4.548600             4.901564
##  Antechinus.flavipes   Antechinus.godmani       Antechinus.leo
##             4.859812             4.890349             5.017280
```

Combine the trees too.

```
joint.tree <- c(coevo$goanna.tree, coevo$marsupial.tree);
class(joint.tree) <- "multiPhylo"
```

Co-Brownian Motion (CoBM)

Design then fit BM model with a shared sigma (rate) parameter. **Warning**: This is much slower than fitting BM to two trees with the *ratebytree* function in *phytools*. I'd suggest you use that one, and it's included later on.

```
modelCoBM <- createModelCoevolution(coevo$marsupial.tree,
                                    coevo$goanna.tree,
                                    keyword="CoBM")
```

```
show(modelCoBM)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "CoBM"
## *** Parameters of the model : [1] "m0" "v0" "d" "sigma"
## *** Description : Brownian Motion model with linear drift.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving independently after
branching.
## dX_t = d dt + sigma dW_t
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ********************************************************************
```

```
ssCoBM <- fitTipData(modelCoBM,
                     joint.traits,
                     GLSstyle=T)
```

Co-Ornstein Uhlenbeck (CoOU)

Design then fit the OU model with shared sigma (rate) and alpha (constraint) parameters. **Warning**: This is much slower than fitting OU to two trees with the *ratebytree* function in *phytools*. I'd suggest you use that one, and it's included later on.

```
modelCoOU <- createModelCoevolution(coevo$marsupial.tree,
                                    coevo$goanna.tree,
                                    keyword="CoOU")
```

```
show(modelCoOU)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "CoOU"
## *** Parameters of the model : [1] "m0" "v0" "psi" "theta" "sigma"
## *** Description : Ornstein-Uhlenbeck model.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving independently after
branching.
## dX_t = psi(theta- X_t) dt + sigma dW_t
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssCoOU <- fitTipData(modelCoOU,
                     joint.traits,
                     GLSstyle=T)
```

Generalist Matching Mutualism (GMM)

Design the GMM model, then fit it to our Trees/Data (assumes interaction only between taxa in DIFFERENT trees)

```
modelGMM <- createModelCoevolution(coevo$marsupial.tree,
                                   coevo$goanna.tree,
                                   keyword="GMM")
```

```
## [1] "The Generalist Matching Mutualism model. Assumes equal interaction (S)
between all inter-clade lineages, but no interaction (0) among lineages within
a tree (intra-clade)"
```

```
show(modelGMM)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "GMM"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S" "sigma"
## *** Description : The Generalist Matching Mutualism model. Assumes equal
interaction (S) between all inter-clade lineages, but no interaction (0) among
lineages within a tree (intra-clade)
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssGMM <- search.surface(modelGMM,
                        n.iter=8,
                        traits=joint.traits,
                        n.proc=4,
                        results="best") # this took ~1min
```

Generalist Matching Mutualism–All (GMM_{all})

Design the GMM_{all} model, then fit it to our Trees/Data (assumes interaction between ALL taxa in both trees)

```
modelGMM_all <- createModelCoevolution(coevo$marsupial.tree,
                                       coevo$goanna.tree,
                                       keyword="GMM_all")
```

```
## [1] "The Generalist Matching Mutualism 'ALL' model. Assumes equal
interaction (S) between all taxa in both trees (inter- and intra-clade)"
```

```
show(modelGMM_all)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "GMM_all"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S" "sigma"
## *** Description : The Generalist Matching Mutualism 'ALL' model. Assumes
equal interaction (S) between all taxa in both trees (inter- and intra-clade)
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssGMM_all <- search.surface(modelGMM_all,
                            n.iter=8,
                            traits=joint.traits,
                            n.proc=4,
                            results="best") # this took ~1min
```

CoEvolution (CoEvo)

Design the CoEvo model, (correct implementation of geography with the GMM model, but no intra-clade competition, meaning geography only influences interactions of taxa in opposing trees)

```
modelCoEvo <- createModelCoevolution(coevo$marsupial.tree,
                                     coevo$goanna.tree,
                                     geo.object = coevo$joint.geo.object,
                                     keyword="CoEvo")
```

```
## [1] "The CoEvo model. An extension of the GMM model, accounting for
interactions only between geographic co-occurring lineages. As with the GMM
model, it only estimates interaction (S) between taxa across trees
(inter-clade, NOT intra-clade). This model also properly accounts for the
number of co-occuring lineages by dividing S (Pk/1) using rowsums (see Manceau
et al. pg.559, equation 7)."
```

```
show(modelCoEvo)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "CoEvo"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S" "sigma"
## *** Description : The CoEvo model. An extension of the GMM model, accounting
for interactions only between geographic co-occurring lineages. As with the GMM
model, it only estimates interaction (S) between taxa across trees
(inter-clade, NOT intra-clade). This model also properly accounts for the
```

```
number of co-occuring lineages by dividing S (Pk/l) using rowsums (see Manceau
et al. pg.559, equation 7).
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssCoEvo <- search.surface(modelCoEvo,
                          n.iter=8,
                          traits=joint.traits,
                          n.proc=4,
                          results="best") # this took ~16 minutes (12 another time)
```

CoEvolution–All ($\mathbf{CoEvo_{all}}$)

Design the $\text{CoEvo}_{\text{all}}$ model, (correct implementation of geography with the GMM model, but no intra-clade competition)

```
modelCoEvo_all <- createModelCoevolution(coevo$marsupial.tree,
                                         coevo$goanna.tree,
                                         geo.object = coevo$joint.geo.object,
                                         keyword="CoEvo_all")
```

```
## [1] "The CoEvo 'ALL' model. An extension of the GMM model, accounting for
interactions only between geographic co-occurring lineages. It is also an
extension of the CoEvo model, estimating interaction (S) between all
co-occurring taxa (inter-clade AND intra-clade). This model also properly
accounts for the number of co-occuring lineages by dividing S (Pk/l) using
rowsums (see Manceau et al. pg.559, equation 7)."
```

```
show(modelCoEvo_all)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "CoEvo_all"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S" "sigma"
## *** Description : The CoEvo 'ALL' model. An extension of the GMM model,
accounting for interactions only between geographic co-occurring lineages. It
is also an extension of the CoEvo model, estimating interaction (S) between all
co-occurring taxa (inter-clade AND intra-clade). This model also properly
accounts for the number of co-occuring lineages by dividing S (Pk/l) using
rowsums (see Manceau et al. pg.559, equation 7).
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssCoEvo_all <- search.surface(modelCoEvo_all,
                              n.iter=8,
                              traits=joint.traits,
                              n.proc=4,
                              results="best") # this took ~
```

CoEvolution–Split ($\mathbf{CoEvo_{split}}$)

Build the $\text{CoEvo}_{\text{split}}$ model with split intra/inter-clade S (correct implemenation of geography with the GMM model, intra (S2) and interclade (S1) interactions are different)

```
modelCoEvo_Split <- createModelCoevolution(coevo$marsupial.tree,
                                    coevo$goanna.tree,
                                    geo.object = coevo$joint.geo.object,
                                    keyword="CoEvo_Split")
```

```
## [1] "The CoEvo 'SPLIT' model. Again, an extension of the GMM model,
accounting for interactions only between geographic co-occurring lineages. It
accounts for interactions between all taxa like the CoEvo_all model, but
estimates a different interaction parameter for intra-clade (S2) and
inter-clade (S1) interactions. This model also properly accounts for the number
of co-occuring lineages by dividing S (Pk/l) using rowsums (see Manceau et al.
pg.559, equation 7)."
```

```
show(modelCoEvo_Split)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "CoEvo_Split"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S1" "sigma" "S2"
## *** Description : The CoEvo 'SPLIT' model. Again, an extension of the GMM
model, accounting for interactions only between geographic co-occurring
lineages. It accounts for interactions between all taxa like the CoEvo_all
model, but estimates a different interaction parameter for intra-clade (S2) and
inter-clade (S1) interactions. This model also properly accounts for the number
of co-occuring lineages by dividing S (Pk/l) using rowsums (see Manceau et al.
pg.559, equation 7).
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ********************************************************************
```

```
ssCoEvo_Split <- search.surface(modelCoEvo_Split,
                                n.iter=8,
                                traits=joint.traits,
                                n.proc=4, no.S=2,
                                results="best")
# this took ~23 minutes (13 another time)
```

Co-Phenotypic Matching (CoPM)

Build the CoPM model, which estimates the PM model on two trees (no interclade interaction), but fits only a single interaction parameter (S) two both trees

```
modelCoPM <- createModelCoevolution(coevo$marsupial.tree,
                                    coevo$goanna.tree,
                                    keyword="CoPM")
```

```
## [1] "The CoPM model. This is a joint estimation of the PM model for two
trees. It estimates single interaction (S) and rate (sigma) values for both
trees, but S is estimated solely from intra-clade interactions (no interaction
between trees). All lineages in a tree are assumed to interact with ALL other
lineages in that tree"
```

```
show(modelCoPM)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
```

```
## *** Name of the model : [1] "CoPM"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S" "sigma"
## *** Description : The CoPM model. This is a joint estimation of the PM model
for two trees. It estimates single interaction (S) and rate (sigma) values for
both trees, but S is estimated solely from intra-clade interactions (no
interaction between trees). All lineages in a tree are assumed to interact with
ALL other lineages in that tree
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssCoPM <- search.surface(modelCoPM,
                         n.iter=8,
                         traits=joint.traits,
                         n.proc=4,
                         results="best") # this took ~
```

Co-Phenotypic Matching with Geography (CoPM_{geo})

Expand the CoPM_{geo} model to incorporate geography, (same S, interaction only between overlapping taxa within a tree)

```
modelCoPM_geo <- createModelCoevolution(coevo$marsupial.tree,
                                        coevo$goanna.tree,
                                        geo.object = coevo$joint.geo.object,
                                        keyword="CoPM_geo")
```

```
## [1] "The CoPM 'GEO' model. This is an extension of the CoPM model, which is
a joint estimation of the PM model for two trees. It estimates single
interaction (S) and rate (sigma) values for both trees, but S is estimated
solely from intra-clade interactions (no interaction between trees). It
correctly accounts for interaction only among geographic overlapping lineages,
and corrects the interaction estimate for the number of overlapping lineages."
```

```
show(modelCoPM_geo)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "CoPM_geo"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S1" "sigma"
## *** Description : The CoPM 'GEO' model. This is an extension of the CoPM
model, which is a joint estimation of the PM model for two trees. It estimates
single interaction (S) and rate (sigma) values for both trees, but S is
estimated solely from intra-clade interactions (no interaction between trees).
It correctly accounts for interaction only among geographic overlapping
lineages, and corrects the interaction estimate for the number of overlapping
lineages.
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssCoPM_geo <- search.surface(modelCoPM_geo,
                             n.iter=8,
                             traits=joint.traits,
                             n.proc=4,
```

```
                                  results="best") # this took ~11 minutes (13 another time)
```

Joint Phenotypic Matching (JointPM)

Build the JointPM model, which estimates separate interaction parameters (S1, S2) for the two trees. Interaction only occurs within a tree.

```
modelJointPM <- createModelCoevolution(coevo$marsupial.tree,
                                       coevo$goanna.tree,
                                       keyword="JointPM")
```

```
## [1] "The JointPM model. This is a joint estimation of the PM model for two
trees. It differs from the CoPM model by estimating separate interaction values
for each clade (tree1 = S1; tree2 = S2). All lineages in a tree are assumed to
interact with ALL other lineages in that tree."
```

```
show(modelJointPM)
```

```
## ****************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "JointPM"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S1" "sigma" "S2"
## *** Description : The JointPM model. This is a joint estimation of the PM
model for two trees. It differs from the CoPM model by estimating separate
interaction values for each clade (tree1 = S1; tree2 = S2). All lineages in a
tree are assumed to interact with ALL other lineages in that tree.
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ****************************************************************
```

```
ssJointPM <- search.surface(modelJointPM,
                            n.iter=8,
                            traits=joint.traits,
                            n.proc=4,
                            no.S=2,
                            results="best") # this took ~
```

Joint Phenotypic Matching with Geography ($\text{JointPM}_{\text{geo}}$)

Build the $\text{JointPM}_{\text{geo}}$ model which incorporates geography and estimates separate interaction parameters (S1, S2) for the two trees. Interaction only occurs within a tree.

```
modelJointPM_geo <- createModelCoevolution(coevo$marsupial.tree,
                                       coevo$goanna.tree,
                                       geo.object = coevo$joint.geo.object,
                                       keyword="JointPM_geo")
```

```
## [1] "The JointPM 'GEO' model. This is a joint estimation of the PM model for
two trees. It differs from the CoPM_geo model by estimating separate
interaction values for each clade (tree1 = S1; tree2 = S2). Like the CoPM_geo
(unlike JointPM) it correctly estimates the interaction parameters (S1,S2) for
only geographic overlapping taxa (it also corrects for the number of taxa
overlapping)."
```

```
show(modelJointPM_geo)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "JointPM_geo"
## *** Parameters of the model : [1] "m0" "v0" "d1" "d2" "S1" "sigma" "S2"
## *** Description : The JointPM 'GEO' model. This is a joint estimation of the
PM model for two trees. It differs from the CoPM_geo model by estimating
separate interaction values for each clade (tree1 = S1; tree2 = S2). Like the
CoPM_geo (unlike JointPM) it correctly estimates the interaction parameters
(S1,S2) for only geographic overlapping taxa (it also corrects for the number
of taxa overlapping).
## *** Periods : the model is cut into 92 parts.
## For more details on the model, call : print(PhenotypicModel)
## ********************************************************************
```

```
ssJointPM_geo <- search.surface(modelJointPM_geo,
                                n.iter=8,
                                traits=joint.traits,
                                n.proc=4,
                                no.S=2,
                                results="best") # this took ~
```

Phenotypic Matching without the OU process ($\mathbf{PMOU_{less}}$)

Create and fit the simplified PM model $\text{PMOU}_{\text{less}}$ (no drift) for goannas

```
modelPMOU <- createModel(coevo$goanna.tree,
                         keyword = "PM_OUless")
```

```
## [1] "PM_OUless: fit the simplified version of the Phenotypic Matching model
(should be nearly identical to the MC model)"
```

```
show(modelPMOU)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "PM_OUless"
## *** Parameters of the model : [1] "m0" "v0" "S" "sigma"
## *** Description : Simplified Phenotype Matching model.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving then non-independtly
according to the Phenotype Matching expression, without the OU term.
## *** Periods : the model is cut into 31 parts.
## For more details on the model, call : print(PhenotypicModel)
## ********************************************************************
```

```
ssPMOU_g <- search.surface(modelPMOU,
                           n.iter=8,
                           traits=gtraits,
                           n.proc=8,
                           results="best")
```

Create and fit the simplified PM model $\text{PMOU}_{\text{less}}$ (no drift) for marsupials

```
modelPMOU <- createModel(coevo$marsupial.tree,
                         keyword = "PM_OUless")
```

```
## [1] "PM_OUless: fit the simplified version of the Phenotypic Matching model
(should be nearly identical to the MC model)"
```

```
show(modelPMOU)
```

```
## ********************************************************************
## *** Object of Class PhenotypicModel ***
## *** Name of the model : [1] "PM_OUless"
## *** Parameters of the model : [1] "m0" "v0" "S" "sigma"
## *** Description : Simplified Phenotype Matching model.
## Starts with two lineages having the same value X_0 ~ Normal(m0,v0).
## One trait in each lineage, all lineages evolving then non-independtly
according to the Phenotype Matching expression, without the OU term.
## *** Periods : the model is cut into 62 parts.
## For more details on the model, call : print(PhenotypicModel)
## ********************************************************************
```

```
ssPMOU_m <- search.surface(modelPMOU,
                           n.iter=8,
                           traits=mtraits,
                           n.proc=4,
                           results="best")
ssPMOU <- joint.fit(ssPMOU_g, ssPMOU_m)
```

Shared/Independent Brownian Motion (BM_{shared}, BM_{ind}) and Ornstein-Uhlenbeck (OU_{shared}, OU_{ind})

Join both trees together into a single multiPhylo object, and the traits into a list of trait matrices

```
gtraits <- subset(joint.traits,
                  names(joint.traits) %in% coevo$goanna.tree$tip.label)
mtraits <- subset(joint.traits,
                  names(joint.traits) %in% coevo$marsupial.tree$tip.label)

rbt.traits <- NULL;
rbt.traits[[1]] <- gtraits;
rbt.traits[[2]] <- mtraits

head(rbt.traits[[1]])
head(rbt.traits[[2]])
```

Fit alternative models (BM, OU) of trait evolution using *ratebytree* in *phytools*, the likelihood of two trees under:
Brownian Motion (with shared and independent rate parameters)

```
ssrbt.BM <- ratebytree(joint.tree,
                       rbt.traits,
                       model="BM")
```

Ornstein Uhlenbeck (with shared and independent rate/alpha parameters)

```
ssrbt.OU <- ratebytree(joint.tree,
                       rbt.traits,
```

```
                        model="OU")
```

Summarizing Model Fit

Now we want to summarize the model fits

```
allmodels <- multiphy.AIC("ss",
             joint.tree,
             models=c("GMM", "GMM_all", "CoEvo",
                      "CoEvo_Split", "CoPM_geo",
                      "CoEvo_all", "CoPM",
                      "JointPM", "JointPM_geo",
                      "PMOU", "rbt.BM", "rbt.OU"))
```

```
##                   logLik no.Param      AIC     AICc delta.AICc       AICcWt
## CoPM_geo      -23.90383        6 59.80767 60.78441  0.0000000 0.2414555383
## CoPM          -24.00661        6 60.01322 60.98997  0.2055527 0.2178722752
## PMOU          -22.21675        8 60.43350 62.14779  1.3633731 0.1221193473
## JointPM_geo   -23.73944        7 61.47888 62.79653  2.0121169 0.0882900034
## JointPM       -23.81588        7 61.63177 62.94942  2.1650026 0.0817923780
## CoEvo_Split   -23.83223        7 61.66447 62.98212  2.1977010 0.0804660098
## GMM_all       -25.28098        6 62.56195 63.53870  2.7542820 0.0609189545
## CoEvo_all     -25.49440        6 62.98880 63.96554  3.1811274 0.0492112203
## GMM           -26.43762        6 64.87524 65.85199  5.0675740 0.0191614101
## rbt.BM_shared -29.90901        3 65.81802 66.08768  5.3032690 0.0170312661
## CoEvo         -27.22316        6 66.44632 67.42307  6.6386539 0.0087351789
## rbt.BM_ind    -29.77132        4 67.54264 67.99719  7.2127713 0.0065554691
## rbt.OU_shared -29.90844        4 67.81688 68.27142  7.4870071 0.0057155000
## rbt.OU_ind    -29.78290        6 71.56580 72.54254 11.7581263 0.0006754489
```

```
##             logLik no.Param      AIC     AICc delta.AICc    AICcWt
## CoPM_geo -23.90383        6 59.80767 60.78441          0 0.2414555
```

```
##            m0            v0            d1            d2            S1
##  5.452294e+00  5.872096e-09  2.108749e-02 -8.051211e-02 -4.408615e-02
##         sigma
##  7.525589e-02
```

```
## [1] "The CoPM 'GEO' model. This is an extension of the CoPM model, which is
a joint estimation of the PM model for two trees. It estimates single
interaction (S) and rate (sigma) values for both trees, but S is estimated
solely from intra-clade interactions (no interaction between trees). It
correctly accounts for interaction only among geographic overlapping lineages,
and corrects the interaction estimate for the number of overlapping lineages."
```

A number of these models (GMM, GMM_all, CoPM, etc.) assume that all contemporaneous lineages are interacting with one another, but that's not realistic. The Australian continent is a big place, so let's drop those unrealistic models.

```
modelsubset <- multiphy.AIC("ss",
             joint.tree,
             models= c("CoEvo", "CoEvo_Split",
                       "CoEvo_all", "CoPM_geo",
                       "JointPM_geo", "rbt.BM", "rbt.OU"))
```

```
##                  logLik no.Param      AIC     AICc delta.AICc      AICcWt
## CoPM_geo      -23.90383        6 59.80767 60.78441   0.000000 0.484718461
## JointPM_geo   -23.73944        7 61.47888 62.79653   2.012117 0.177240890
## CoEvo_Split   -23.83223        7 61.66447 62.98212   2.197701 0.161534338
## CoEvo_all     -25.49440        6 62.98880 63.96554   3.181127 0.098790805
## rbt.BM_shared -29.90901        3 65.81802 66.08768   5.303269 0.034190017
## CoEvo         -27.22316        6 66.44632 67.42307   6.638654 0.017535744
## rbt.BM_ind    -29.77132        4 67.54264 67.99719   7.212771 0.013160008
## rbt.OU_shared -29.90844        4 67.81688 68.27142   7.487007 0.011473783
## rbt.OU_ind    -29.78290        6 71.56580 72.54254  11.758126 0.001355954
```

```
##             logLik no.Param      AIC     AICc delta.AICc    AICcWt
## CoPM_geo -23.90383        6 59.80767 60.78441          0 0.4847185
```

```
##            m0            v0            d1            d2            S1
##  5.452294e+00  5.872096e-09  2.108749e-02 -8.051211e-02 -4.408615e-02
##         sigma
##  7.525589e-02
```

```
## [1] "The CoPM 'GEO' model. This is an extension of the CoPM model, which is
a joint estimation of the PM model for two trees. It estimates single
interaction (S) and rate (sigma) values for both trees, but S is estimated
solely from intra-clade interactions (no interaction between trees). It
correctly accounts for interaction only among geographic overlapping lineages,
and corrects the interaction estimate for the number of overlapping lineages."
```

Compare all the models (then save em)

```
model.list <- c("GMM", "GMM_all",
                "CoEvo", "CoEvo_Split",
                "CoPM_geo", "CoEvo_all",
                "CoPM", "JointPM",
                "JointPM_geo")
all.res <- multiphy.AIC("ss",
                        joint.tree,
                        models=model.list)
fit.res <- lapply(model.list,
                  function(x) get(paste0("ss", x)));
names(fit.res) <- model.list

save.all.models <- paste0(save.path,
                          group.name,
                          "_Empirical_ModelObjects.RData")
save(fit.res, ssGMM,
     ssGMM_all, ssCoEvo,
     ssCoEvo_all, ssCoEvo_Split,
     ssCoPM, ssCoPM_geo,
     ssJointPM, ssJointPM_geo,
     ssPMOU, file=save.all.models)
```

Compare just the realistic models (then save em)

```
model.list <- c("CoEvo", "CoEvo_Split",
                "CoEvo_all", "CoPM_geo",
                "JointPM_geo", "rbt.BM",
                "rbt.OU")
```

```
all.res <- multiphy.AIC("ss",
                        joint.tree,
                        models=model.list)
fit.res <- lapply(model.list,
                  function(x) get(paste0("ss", x)));
names(fit.res) <- model.list
```