phytools deomonstration (Version 0.1-61)

Liam J. Revell

January 17, 2012

1 Introduction

This document provides a demonstration of some of the functionality of the phytools package in simulation, comparative biology, and plotting.

2 Simulation

The first function does fast Brownian motion simulation as well as simulation under related models. Brownian motion is a stochastic model in which evolutionary changes along the edges of the tree are drawn from a random normal distribution with mean zero and variance proportional to the product of the variance of the Brownian process and the length of the edge. The phytools function fastBM conducts a number of different simulations in this general vein.

```
> library(phytools); library(geiger)
> tree<-pbtree(n=20,scale=30) # simulate tree

> plotTree(tree,pts=F)
```

(Figure 1)

```
> x<-fastBM(tree,sig2=0.1,internal=T) # simulate data

> par(mfrow=c(2,1))
> phenogram(tree,x,ftype="off")
> title("Known ancestors")
> phenogram(tree,x[1:20],ftype="off")
> title("ML estimated ancestors")
```

(Figure 2)

Ok, that was pretty easy, let's try out some of the other capabilities of fastBM(). For instance, it can do simulations with bounds and with a trend:

```
> # first with bounds
> x<-fastBM(tree,sig2=0.1,bounds=c(-1,1),internal=T)
```

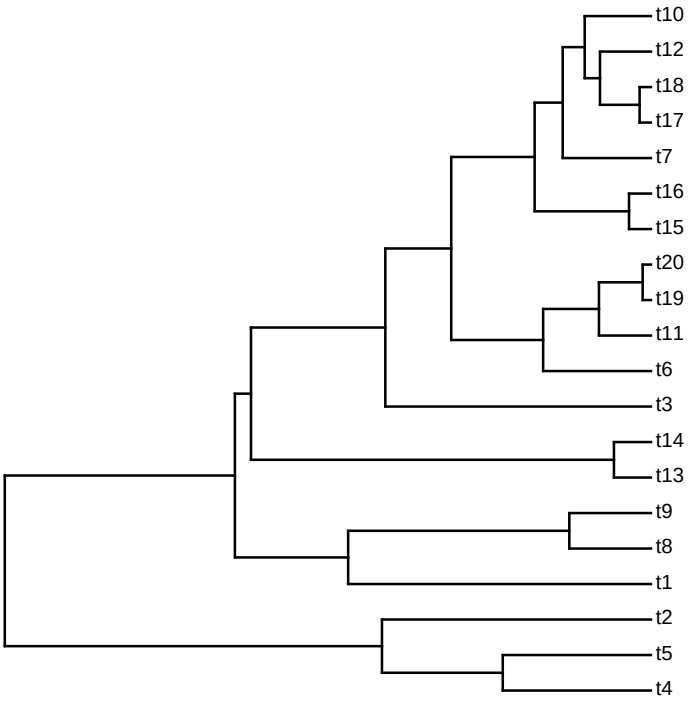

Figure 1: Simulated phylogeny.

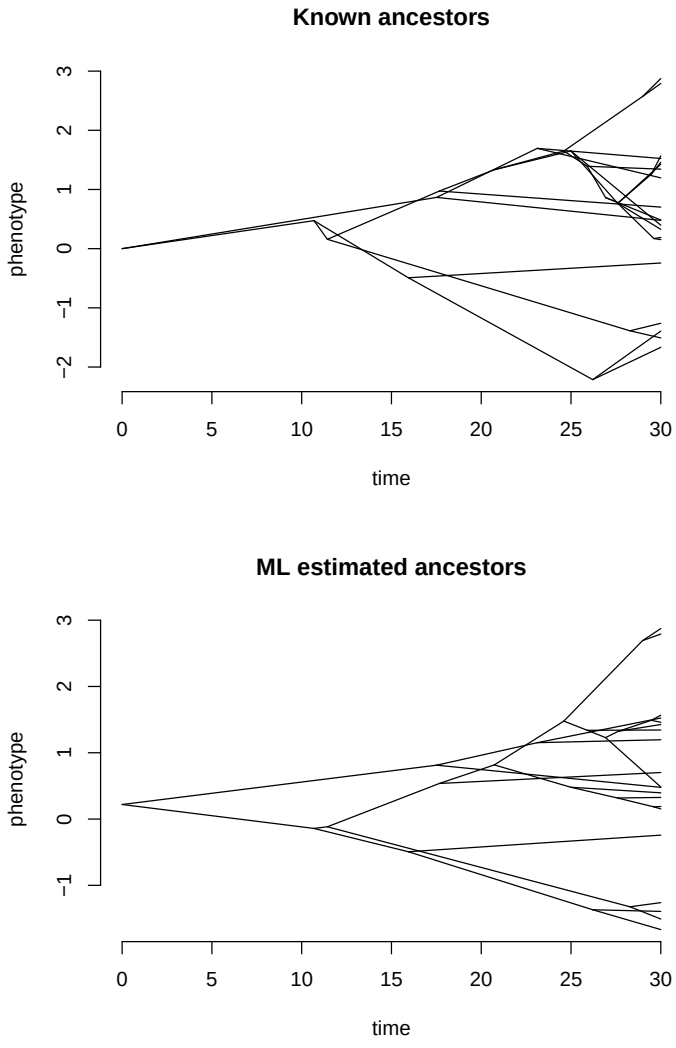

Figure 2: Phenogram plots.

```
> par(mfrow=c(2,1))
> phenogram(tree,x,ftype="off")
> title("Known ancestors")
> phenogram(tree,x[1:20],ftype="off")
> title("ML estimated ancestors")
```

(Figure 3)

```
> # now with a trend
> x<-fastBM(tree,sig2=0.1,mu=0.1,internal=T)

> par(mfrow=c(2,1))
> phenogram(tree,x,ftype="off")
> title("Known ancestors")
> phenogram(tree,x[1:20],ftype="off")
> title("ML estimated ancestors")
```

(Figure 4)

Phytools has lots of other simulation functions as well. For instance, it can be used to simulate the history of a discrete character evolving on the tree. Let's try it.

```
> tree<-pbtree(n=50,scale=1)

> plotTree(tree,fsize=0.7,pts=F)
```

(Figure 5)

```
> Q<-matrix(c(-2,1,1,1,-2,1,1,1,-2),3,3)
> mtree<-sim.history(tree,Q,anc="1")
> cols<-c("blue","green","red")
> names(cols)<-c(1,2,3)

> plotSimmap(mtree,cols,pts=F,lwd=3)
```

(Figure 6)

```
> nchanges<-sum(sapply(mtree$maps,length))-nrow(tree$edge)
> nchanges

[1] 34
```

We can also simulate an irreversible trait using this function:

```
> Q<-matrix(c(-1,1,1e-10,-1e-10),2,2)
> rownames(Q)<-colnames(Q)<-c("0","1")
> itree<-sim.history(tree,Q,"0")
> cols<-c("blue","red"); names(cols)<-c(0,1)
```

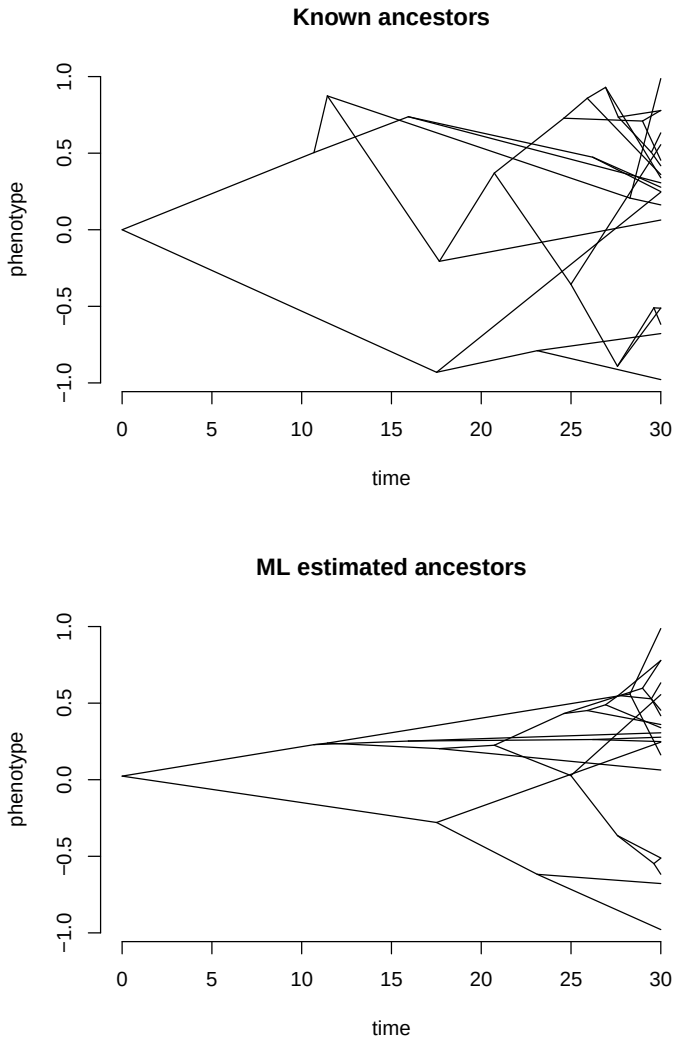

Figure 3: Phenogram plots. BM with bounds.

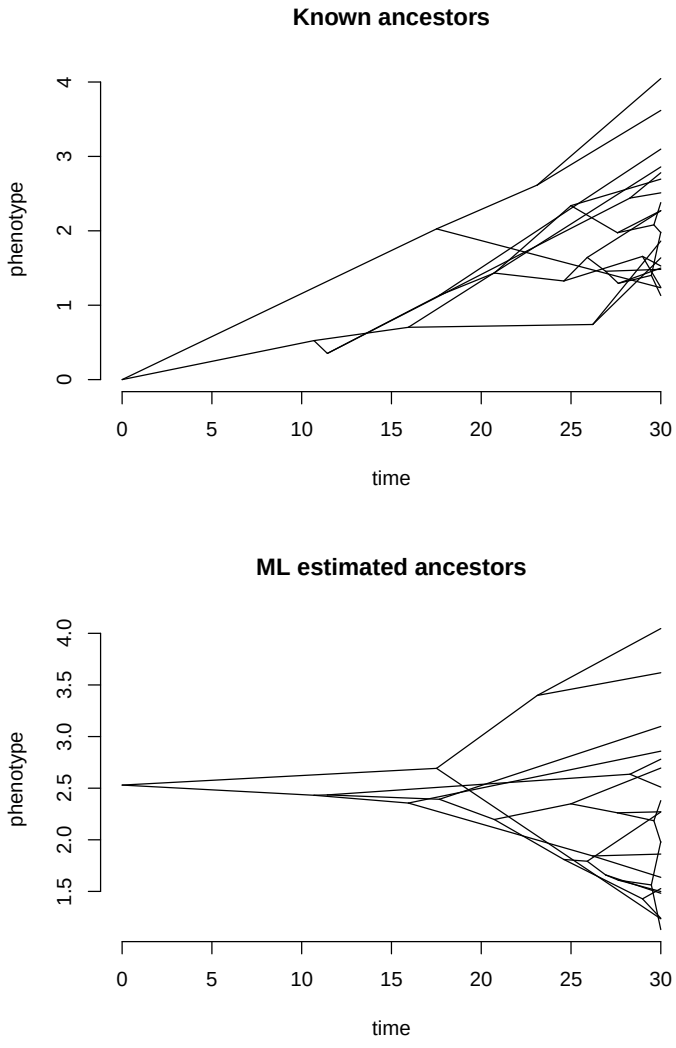

Figure 4: Phenogram plots. BM with a trend.

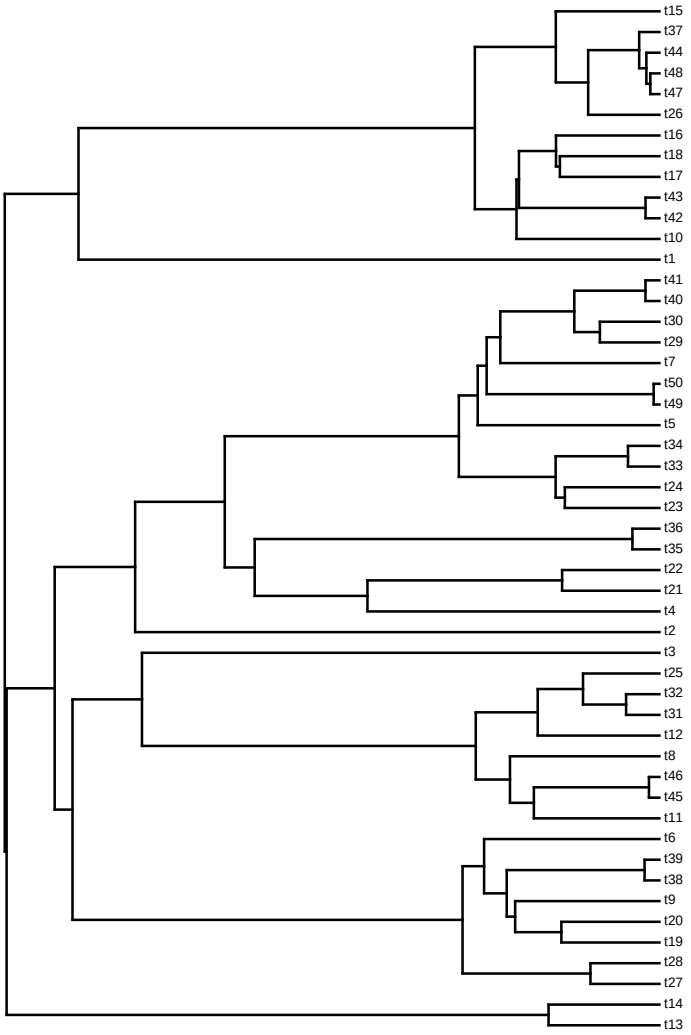

Figure 5: Simulated tree.

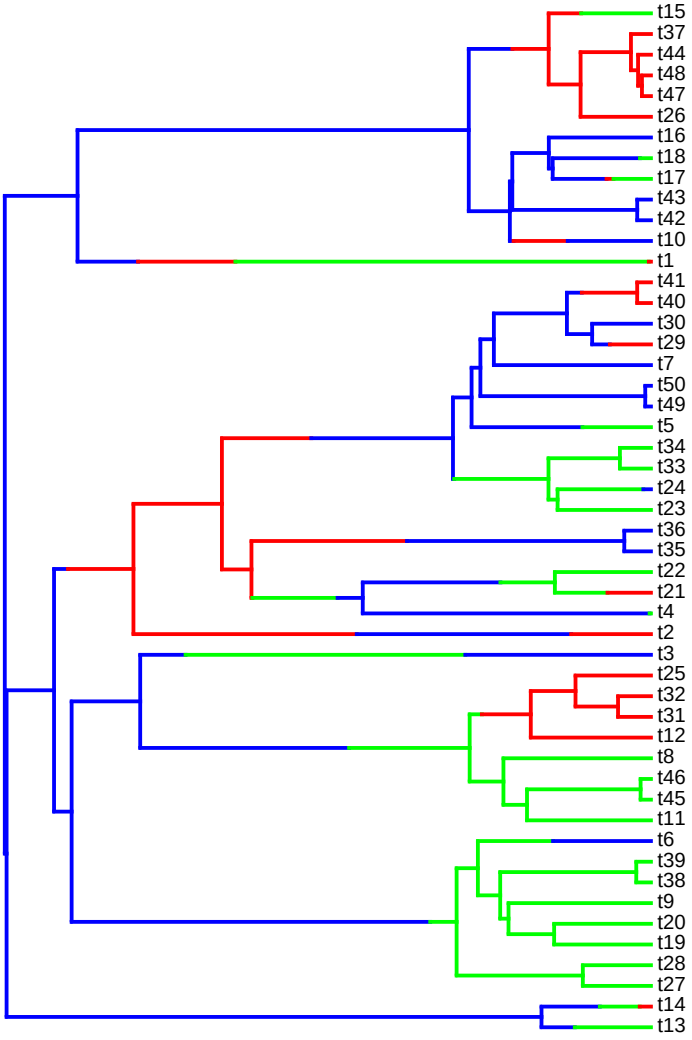

Figure 6: Simulated discrete character history.

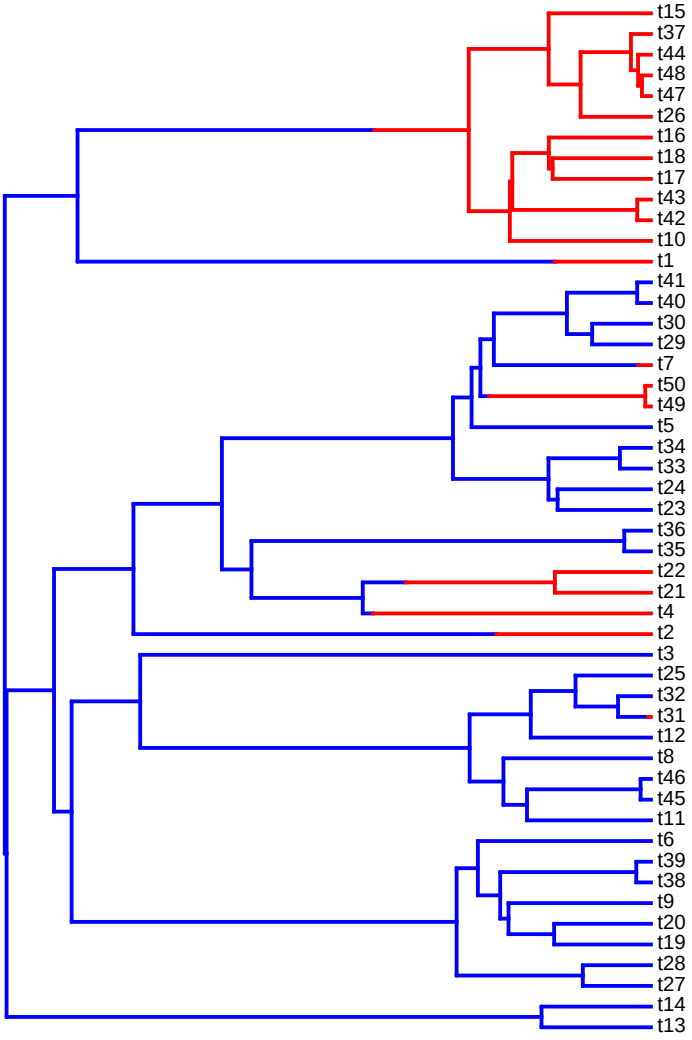

Figure 7: Simulated irreversible discrete character history.

```
> plotSimmap(itree,cols,pts=F,lwd=3)
```

(Figure 7)

With phytools you can also simulate a continuous trait that evolves with a rate conditioned on the state for a discrete character. Let's try that.

```
> sig2<-c(1,5,10); names(sig2)<-c(1,2,3)
> x<-sim.rates(mtree,sig2)

> par(mfrow=c(2,1))
> cols<-c("blue","green","red")
> names(cols)<-c(1,2,3)
> plotSimmap(mtree,colors=cols,pts=F,ftype="off",lwd=3)
> par(mar=c(1,1,1,1))
> phenogram(mtree,x,colors=cols,ftype="off")
> title("Simulated data on the tree")
```

(Figure 8)

3 Comparative Methods

The second major area of phytools is phylogenetic comparative biology. phytools includes several different likelihood and Bayesian phylogenetic comparative methods. First, let's take the data set and tree we generated in the last simulation. Recall that the data were generated under a model in which the rate of evolutionary changed as a function of the state of a discrete character's history on the tree. Well, we can fit this model to our data mapped history. This analysis is based on Brian O'Meara et al.'s (2006) Evolution paper.

```
> res<-brownie.lite(mtree,x)
> res

$sig2.single
[1] 4.290981

$a.single
[1] 0.3230227

$var.single
[1] 0.7365006

$logL1
[1] -71.97657

$k1
[1] 2
```

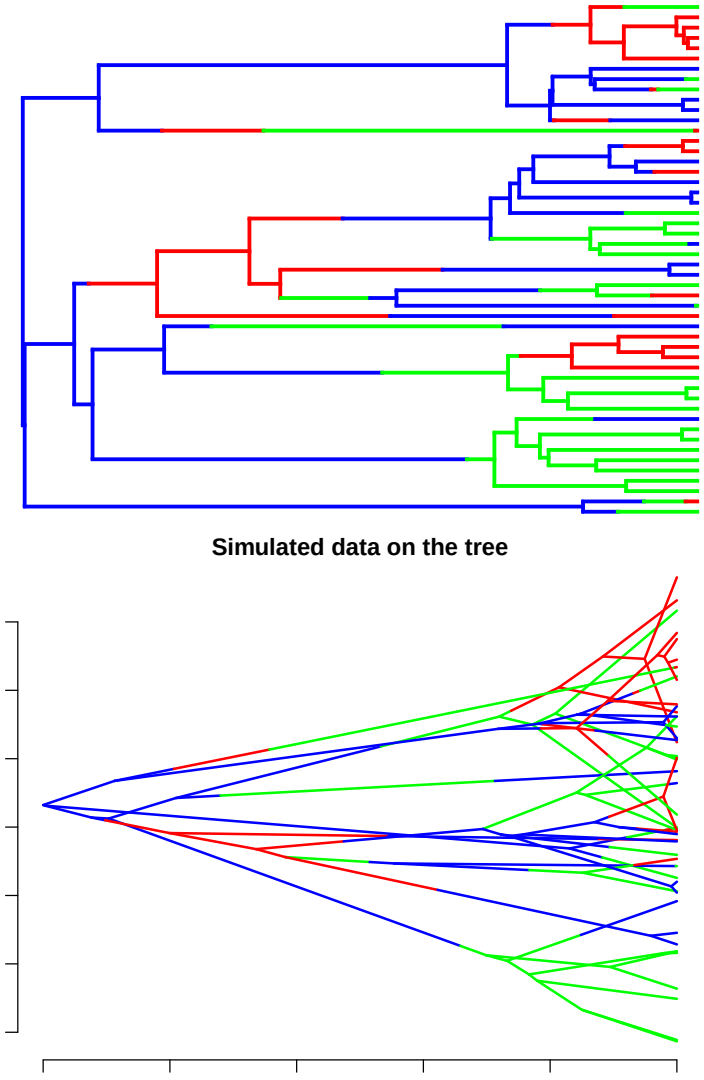

Figure 8: Simulated data where the rate of a continuous trait depends on the state of the discrete character mapped on the tree.

```
$sig2.multiple
       1        2        3
1.203670 3.486513 8.570042

$a.multiple
[1] 0.2775211

$vcv.multiple
            1          2           3
1  0.25592833 -0.1004861  0.06648948
2 -0.10048606  1.4995836 -0.18364060
3  0.06648948 -0.1836406  8.79919427

$logL.multiple
[1] -65.11904

$k2
[1] 4

$P.chisq
[1] 0.001051506

$convergence
[1] "Optimization has converged."
```

phytools also contains a REML version of this function which runs much faster and also has the property of being unbiased. Let's try it.

```
> res<-brownieREML(mtree,x)
> res

$sig2.single
[1] 4.378552

$logL1
[1] -71.3212

$sig2.multiple
       1        2        3
1.277818 3.500567 8.640819

$logL2
[1] -64.87327

$convergence
[1] TRUE
```

In this case we know the true history for our discretely valued trait. Typically this will not be true. Usually we have only the states at the tips of the tree. In the case of our mapped tree object, these values are stored in:

```
> y<-mtree$states
> y

 t1  t2  t3  t4  t5  t6  t7  t8  t9 t10 t11 t12 t13 t14 t15 t16 t17
"3" "3" "1" "2" "2" "1" "1" "2" "2" "1" "2" "3" "2" "3" "2" "1" "2"
t18 t19 t20 t21 t22 t23 t24 t25 t26 t27 t28 t29 t30 t31 t32 t33 t34
"2" "2" "2" "3" "2" "2" "1" "3" "3" "2" "2" "3" "1" "3" "3" "2" "2"
t35 t36 t37 t38 t39 t40 t41 t42 t43 t44 t45 t46 t47 t48 t49 t50
"1" "1" "3" "2" "2" "3" "3" "1" "1" "3" "2" "2" "3" "3" "1" "1"
```

One way to deal with this is by generating possible character histories according to their probability. We can do this using the phytools function make.simmap. Let's try it, this time generating 100 replicate stochastic character histories conditioned on our tip data.

```
> mtrees<-make.simmap(tree,y,nsim=100)
> overlap<-sapply(mtrees,map.overlap,tree2=mtree)

> hist(overlap)
```

(Figure 9)

We can then use brownieREML to fit the model across all the stochastically mapped histories.

```
> Res<-sapply(mtrees,function(...) brownieREML(...)$sig2.multiple[as.character(1:3)],x=x)
> rowMeans(Res)

       1        2        3
1.391261 3.406900 9.482656
```

Phytools performs many other comparative analyses also. For instance, I recently added a function to do Bayesian ancestral character estimation. Let's try it out.

```
> x<-fastBM(tree,internal=TRUE)
> X<-anc.Bayes(tree,x[1:50],ngen=100000)

List of 7
 $ sig2   : num 1.02
 $ a      : num [1, 1] -0.145
 $ y      : num [1:48] -0.145 -0.145 -0.145 -0.145 -0.145 ...
 $ pr.mean: num [1:50] 1000 0 0 0 0 0 0 0 0 0 ...
 $ pr.var : num [1:50] 1e+06 1e+03 1e+03 1e+03 1e+03 1e+03 1e+03 1e+03 1e+03 1e+03 ...
 $ prop   : num [1:50] 0.0102 0.0102 0.0102 0.0102 0.0102 ...
 $ sample : num 100
```

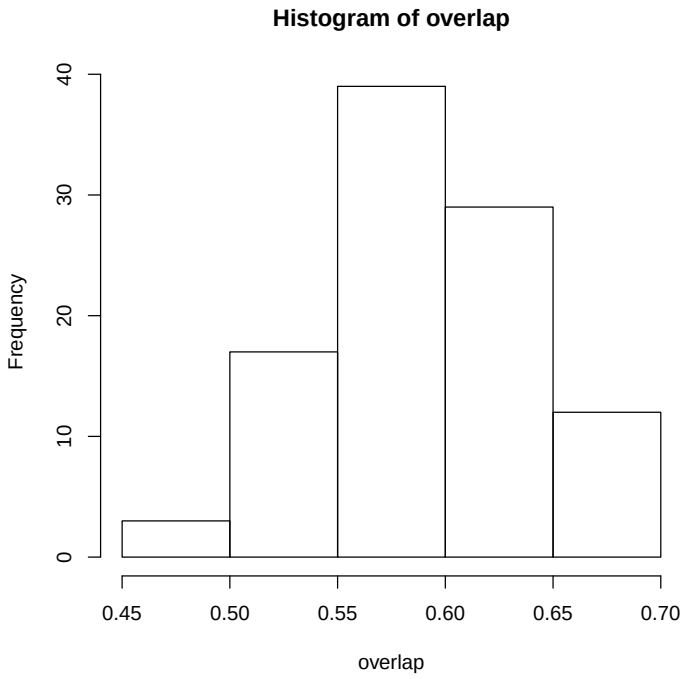

Figure 9: Overlap between generating and stochastically mapped character histories from 100 simulations.

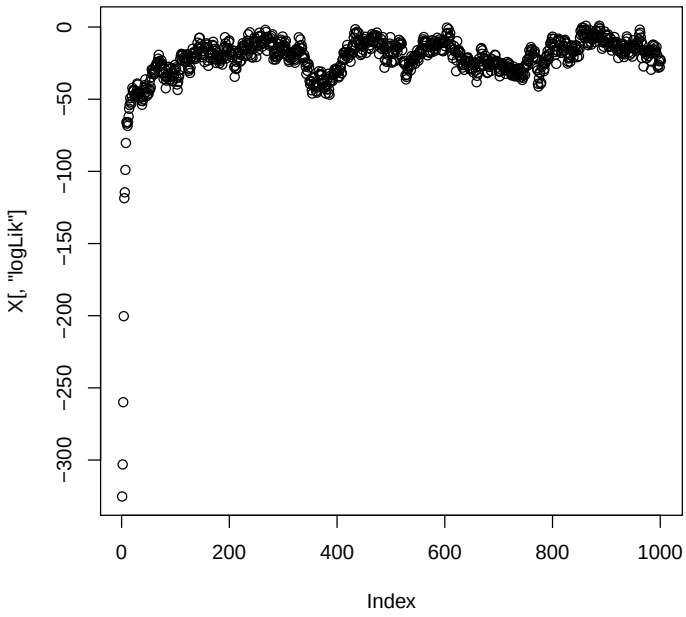

Figure 10: Log-likelihood trace from Bayesian ancestral character estimation.

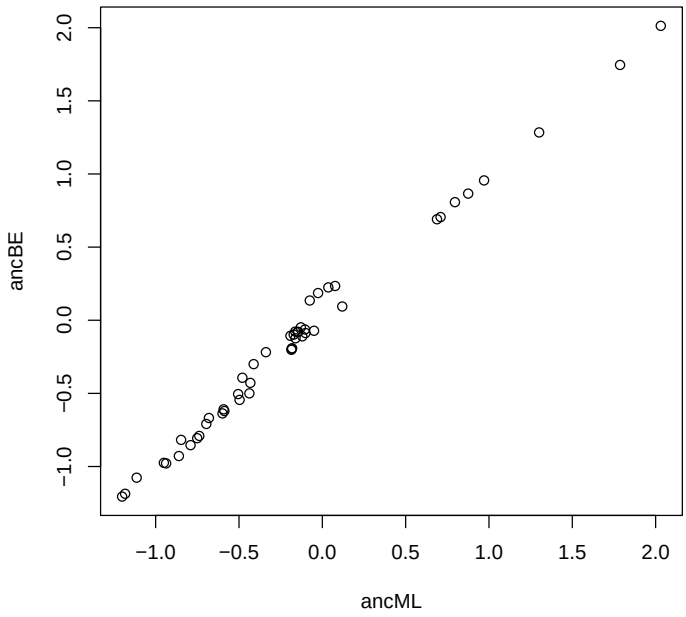

Figure 11: Bayesian ML estimated ancestral character states.

```
> plot(X[,"logLik"])
```

(Figure 10)

```
> ancBE<-colMeans(X[200:nrow(X),as.character(length(tree$tip)+1:tree$Nnode)])
> ancML<-anc.ML(tree,x,maxit=4000)$ace

> plot(ancML,ancBE)
```

(Figure 11)

phytools can also do ancestral character estimation with a trend using likelihood. Note that this will only work if the tree is ultrametric.

```
> atree<-rtree(n=50)
> x<-fastBM(atree,mu=1.0,internal=TRUE)
> res<-anc.trend(atree,x[1:50])
> res

$ace
        51          52          53          54          55
```

```
-1.14265483 -1.14278779  0.11926584  0.68984684  0.86448201
         56          57          58          59          60
 1.17591127  1.55225238  1.43930541  3.16915943  2.52798399
         61          62          63          64          65
 1.63246197  2.23360464  2.79337753  3.38506986  4.12820629
         66          67          68          69          70
 2.89317675 -1.28396963  0.01839947  0.09567194  1.65622321
         71          72          73          74          75
 1.75160827  2.72180020  4.73958462 -1.62108449  0.07274976
         76          77          78          79          80
 2.01357839  2.77462041  3.11703909  3.98643572  4.62764077
         81          82          83          84          85
 4.85052052  5.10257546  6.87938114  7.64186028  7.90656412
         86          87          88          89          90
 8.06708958  5.08494941  5.90741942  2.34580356  2.73114851
         91          92          93          94          95
 3.33237051  3.39787654  3.83739941  3.91399711  4.10902602
         96          97          98          99
 1.62643681  1.93833540  1.94401642  2.45016863

$mu
[1] 1.419129

$sig2
[1] 0.5042665

$logL
[1] -53.26643

$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"

> plot(x[as.character(51:99)],res$ace[as.character(51:99)])
```

(Figure 12)

phytools can be used to compute phylogenetic signal using a couple of different methods. This is implemented in the function phylosig, which can also be used to test hypotheses about signal.

```
> x<-fastBM(tree)
> K<-phylosig(tree,x) # Blomberg et al.'s K
> K

[1] 0.5825225
```

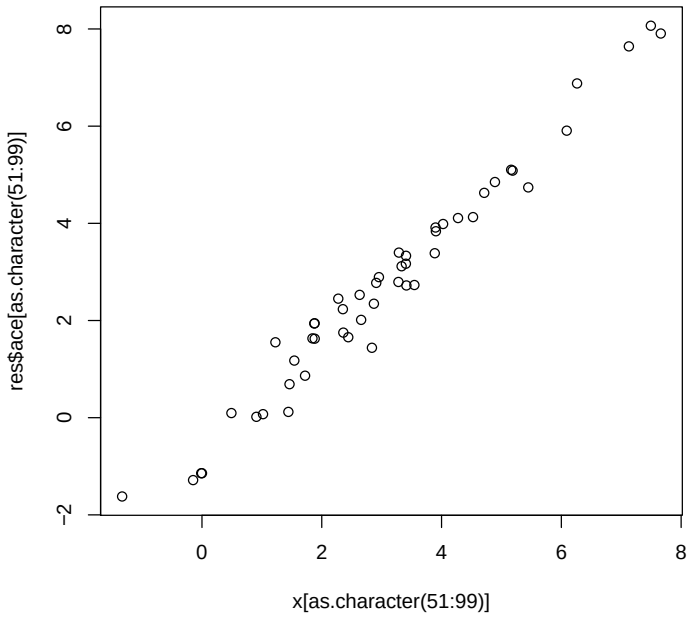

Figure 12: Simulated vs. estimated ancestral character states with a trend.

```
> X<-fastBM(tree,nsim=10)
> K<-apply(X,2,phylosig,tree=tree) # compute K for multiple traits
```

It's possible to incorporate sampling error too, following the method of Ives et al.

```
> se<-rexp(n=length(tree$tip),rate=4)
> names(se)<-tree$tip.label
> xe<-x+rnorm(n=length(tree$tip),sd=se)
> phylosig(tree,xe) # ignoring error

[1] 0.2868017

> phylosig(tree,xe,se=se)

$K
[1] 0.7264139

$sig2
[1] 0.6293001

$logL
[1] -41.56183
```

phylosig also does hypothesis testing, for instance, that there is no signal

```
> phylosig(tree,x,test=T)

$K
[1] 0.5825225

$P
[1] 0.001
```

Or, as I illustrated recently on my blog, we can test that phylogenetic signal is not equal to 1.0.

```
> K<-phylosig(tree,x)
> nullK<-apply(fastBM(tree,n=1000,sig2=mean(pic(x,tree)^2)),2,phylosig,tree=tree)
> P<-mean(abs(log(c(K,nullK)))>=abs(log(K)))
> P

[1] 0.1948052
```

We can also estimate lambda, from Pagel (1999).

```
> Lambda<-phylosig(tree,x,method="lambda")
> Lambda
```

```
$lambda
[1] 0.9963419

$logL
[1] -30.48883
```

```
> Lambda<-simplify2array(apply(X,2,phylosig,tree=tree,method="lambda",test=T))
> Lambda
```

```
       [,1]          [,2]       [,3]       [,4]         [,5]
lambda 1.000299     0.9974273 1.002342  0.9931034   1.008489
logL   -26.28481    -31.87319 -31.47542 -33.46167   -35.07487
logL0  -53.21521    -71.00668 -84.02907 -51.51849   -70.47619
P      2.151612e-13 0         0         1.86139e-09 0
       [,6]          [,7]          [,8]          [,9]
lambda 1.003889     1.008798     0.9985882    0.9910816
logL   -31.04126    -36.34281    -37.40873    -35.96723
logL0  -57.62758    -59.58095    -67.47428    -41.98976
P      3.055334e-13 9.273471e-12 8.881784e-15 0.0005192961
       [,10]
lambda 1.001657
logL   -37.11634
logL0  -57.9989
P      1.029228e-10
```

phytools has other kinds of model fitting as well. For instance, we can fit a model in which the rates and correlation between two characters is different in different parts of a phylogeny. So, for instance, let's simulate data in which the rate of evolution variances according to a discrete character state - but the correlation is the same.

```
> Q<-matrix(c(-1,1,1,-1),2,2)
> mtree<-sim.history(pbtree(n=100,scale=1),Q)
> X<-cbind(sim.rates(mtree,c(1,10)),sim.rates(mtree,c(2,5)))
> res<-evolvcv.lite(mtree,X)
> res
```

```
$model1
$model1$description
[1] "common rates, common correlation"

$model1$R
           [,1]       [,2]
[1,]  3.2935426 -0.4537523
[2,] -0.4537523  2.1434383

$model1$logLik
```

```
[1] -247.0379

$model1$convergence
[1] 0

$model1$k
[1] 4

$model1$AIC
[1] 502.0758


$model2
$model2$description
[1] "different rates, common correlation"

$model2$R1
            [,1]        [,2]
[1,]  0.98883220 -0.07068668
[2,] -0.07068668  1.72627782

$model2$R2
           [,1]       [,2]
[1,] 12.3977654 -0.3625732
[2,] -0.3625732  3.6224805

$model2$logLik
[1] -218.5456

$model2$convergence
[1] 0

$model2$k
[1] 6

$model2$AIC
[1] 449.0912


$model3
$model3$description
[1] "common rates, different correlation"

$model3$R1
          [,1]      [,2]
[1,] 3.2743511 0.2710569
```

```
[2,] 0.2710569 2.0990953

$model3$R2
           [,1]       [,2]
[1,]  3.2743511 -0.7281047
[2,] -0.7281047  2.0990953

$model3$logLik
[1] -246.1355

$model3$convergence
[1] 0

$model3$k
[1] 5

$model3$AIC
[1] 502.2711


$model4
$model4$description
[1] "no common structure"

$model4$R1
           [,1]       [,2]
[1,] 0.98926078 0.06009565
[2,] 0.06009565 1.73395461

$model4$R2
          [,1]      [,2]
[1,] 12.370659 -2.336816
[2,] -2.336816  3.609812

$model4$logLik
[1] -217.2797

$model4$convergence
[1] 0

$model4$k
[1] 7

$model4$AIC
[1] 448.5595
```

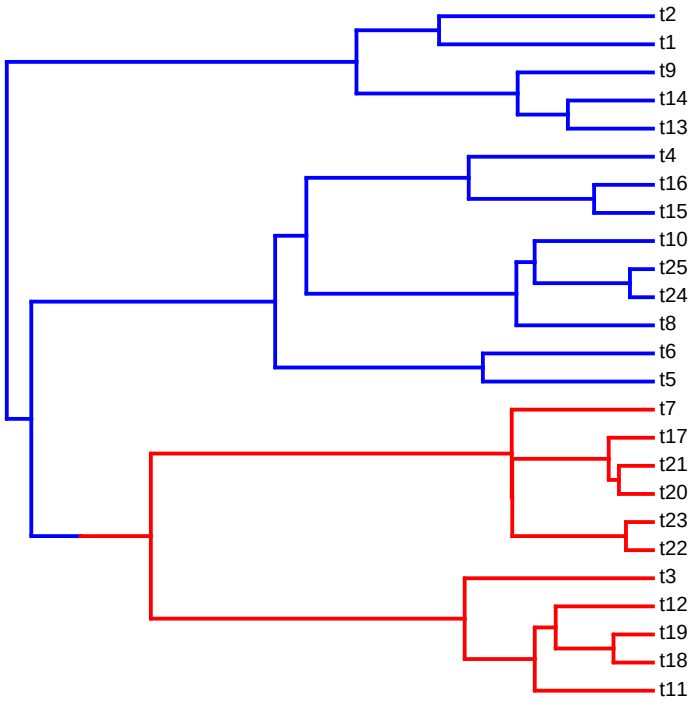

Figure 13: Tree simulated with one change in a discrete character at least 10 descendants in each state.

phytools also has a Bayesian MCMC method for locating the position of a shift in the evolutionary rate over time. Just to illustrate this, let's use another simulated example.

```
> tree<-pbtree(n=25,scale=1)
> mtree<-sim.history(tree,Q/sum(tree$edge.length),anc=1); i<-1
> while(((sum(sapply(mtree$maps,length))-nrow(tree$edge))!=1)||(min(c(sum(mtree$states=="1"),
+   mtree<-sim.history(tree,Q/sum(tree$edge.length),anc=1)

> cols<-c("blue","red"); names(cols)<-c(1,2)
> plotSimmap(mtree,cols,pts=F,lwd=3)
```

(Figure 13)

```
> x<-sim.rates(mtree,sig2=c(1,10))
> X<-evol.rate.mcmc(tree,x,ngen=100000,control=list(print=20000))

List of 11
 $ sig1      : num 7.95
```

```
 $ sig2      : num 7.95
 $ a         : num -0.704
 $ sd1       : num 1.59
 $ sd2       : num 1.59
 $ sda       : num 0.141
 $ kloc      : num 0.2
 $ sdlnr     : num 1
 $ rand.shift: num 0.05
 $ print     : num 20000
 $ sample    : num 100
      state         sig1         sig2            a         node
  0.00000000   7.95494915   7.95494915  -0.70417162  35.00000000
          bp   likelihood
  0.03199629 -47.61158588
      state         sig1         sig2            a         node
2.000000e+04 6.206502e+00 6.621798e+00 4.852515e+00 3.000000e+01
          bp   likelihood
4.100589e-02 7.736090e+01
      state         sig1         sig2            a         node
4.000000e+04 6.206502e+00 6.621798e+00 1.091092e+00 3.000000e+01
          bp   likelihood
6.315069e-03 7.895385e+01
       state          sig1          sig2             a          node
 6.000000e+04  6.206506e+00  1.153662e+01 -2.110307e+02  3.000000e+01
           bp    likelihood
 1.103638e-02  6.678415e+07
       state          sig1          sig2             a          node
 8.000000e+04  6.206506e+00  1.153662e+01 -4.881553e+02  3.000000e+01
           bp    likelihood
 1.103638e-02  3.392938e+08
       state          sig1          sig2             a          node
 1.000000e+05  6.206506e+00  1.153662e+01 -7.726893e+02  3.000000e+01
           bp    likelihood
 1.103638e-02  8.376212e+08
```

```
> ave.shift<-minSplit(tree,X$mcmc[201:1001,c("node","bp")])
> res<-posterior.evolrate(tree,ave.shift,X$mcmc[201:1001,],X$tips[201:1001])
> colMeans(res)
```

```
       state          sig1          sig2             a          node
 6.000000e+04  1.080938e+01  3.080777e+00 -2.637709e+02  3.000000e+01
           bp    likelihood
 1.503418e-02  1.914997e+08
```

The last comparative method I'm going to talk about is a new Bayesian approach for incorporating intraspecific variability into model fitting for comparative biology. The neat thing about this approach is that it samples the

evolutionary model parameters as well as the species means and variances from their join posterior probability distribution. Just to illustrate this, I'm going to first simulate some data under the lambda model, and then I'll run the Bayesian MCMC.

```
> tree<-pbtree(n=50,scale=1)
> xbar<-fastBM(lambdaTree(tree,0.7),sig2=2)
> x<-sampleFrom(xbar,xvar=1.0,randn=c(1,5))
> X<-fitBayes(tree,x,ngen=100000,model="lambda")

List of 9
 $ sig2   : num 16
 $ lambda : num 1
 $ a      : num -0.117
 $ xbar   : Named num [1:50] 0.5335 0.2127 0.1171 -0.0694 3.4719 ...
  ..- attr(*, "names")= chr [1:50] "t1" "t2" "t3" "t4" ...
 $ intV   : num 1.04
 $ pr.mean: num [1:54] 1000 0.501 0 0 0 ...
 $ pr.var : num [1:54] 1e+06 1e+00 1e+03 1e+03 1e+03 ...
 $ prop   : num [1:54] 0.16 0.02 0.16 0.16 0.16 ...
 $ sample : num 100

> x.bayes<-colMeans(X[201:nrow(X),])[tree$tip.label]
> plot(xbar,x.bayes)
> temp<-aggregate(x,list(species=as.factor(names(x))),mean)
> x.arith<-temp[,2]; names(x.arith)<-temp[,1]; x.arith<-x.arith[tree$tip.label]
> sse.bayes<-sum((x.bayes-xbar)^2)
> sse.bayes

[1] 12.75183

> sse.arith<-sum((x.arith-xbar)^2)
> sse.arith

[1] 22.28838
```

The two SSE values are the sum squared deviation between the Bayesian species means and the true values, vs. the arithmetic species means and their true values.

4 Plotting

The final demonstration is of some of the plotting functionally of phytools is in plotting. We have already seen plotTree, plotSimmap, and phenogram; phytools also can do lineage through time plots, including with extinct lineages.

```
> tree<-birthdeath.tree(b=1,d=0.3,taxa.stop=100)
> z<-ltt(tree)
```

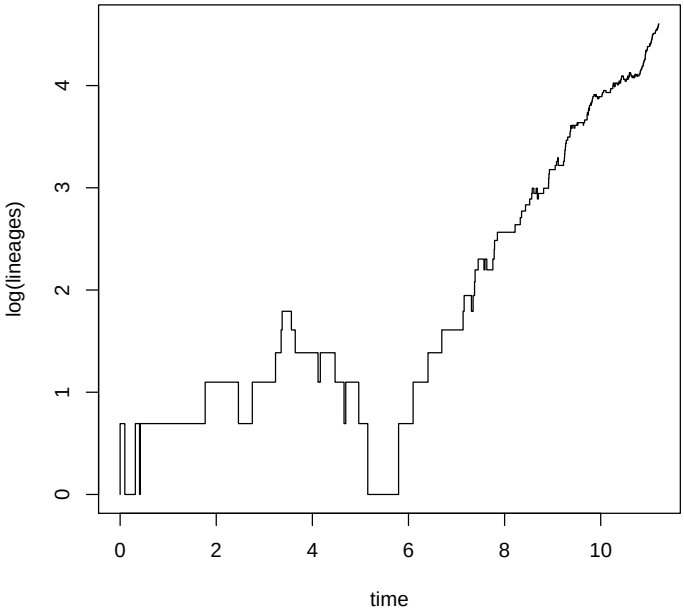

Figure 14: Lineage through time plot with extinct lineages.

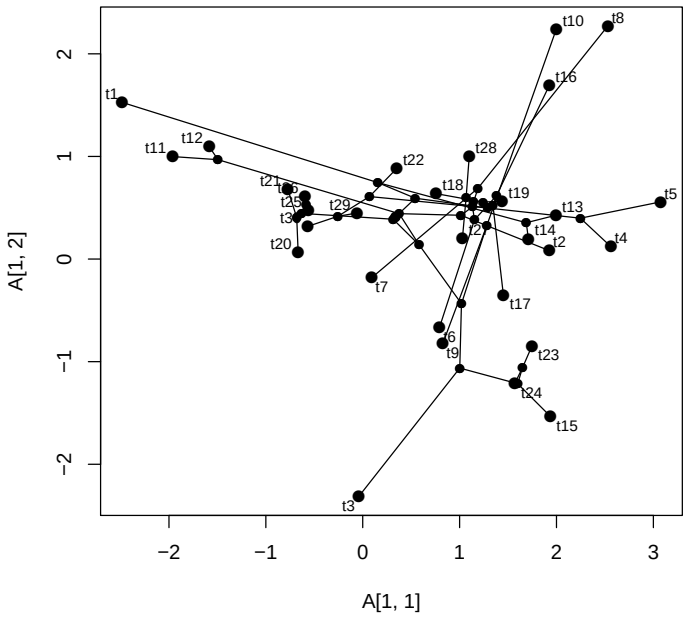

Figure 15: Projection of the phylogeny into two-dimensional morphospace.

(Figure 14)

We can also do a phylomorphospace plot following Sidlauskas (2008).

```
> tree<-pbtree(n=30)
> X<-fastBM(tree,nsim=2)
> phylomorphospace(tree,X)
```

(Figure 15)

5 Conclusion

OK, that's the end!

6 Session Information

The version number of and packages loaded for generating the vignette were:

- R version 2.14.1 (2011-12-22), `i386-pc-mingw32`

- Locale: `LC_COLLATE=C`, `LC_CTYPE=English_United States.1252`, `LC_MONETARY=English_United States.1252`, `LC_NUMERIC=C`, `LC_TIME=English_United States.1252`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: MASS 7.3-16, Matrix 1.0-2, ape 2.8, geiger 1.3-1, igraph 0.5.5-4, lattice 0.20-0, mnormt 1.4-5, msm 1.1, mvtnorm 0.9-9991, ouch 2.8-1, phangorn 1.5-1, phytools 0.1-61, subplex 1.1-3
- Loaded via a namespace (and not attached): calibrate 1.7, gee 4.13-17, grid 2.14.1, nlme 3.1-102, numDeriv 2010.11-1, splines 2.14.1, survival 2.36-10, tools 2.14.1