Distribution of phylogenetic contrasts under OU model

Qing (Sabrina) Yu and Cecile Ane

August 22, 2016

Goal of the study

In this report, our goal is to check the distribution of phylogenetic contrasts that are used in the bootstrap procedure in the `l1ou` package. We found that for some nodes, the contrasts' mean was far away from zero. We found the same variance for contrasts on all nodes, but with an unexpected value, so we investigated a few potential reasons for this. We did not identify a specific reason at this point, but the conclusion is that we may need to fix the contrast calculation for the bootstrap in the `l1ou` package.

We used code from the bootstrap procedure to simulate contrast values. Then we calculated the mean and standard deviation of the contrast values at each node. The conclusions above held for all nodes, whether nodes had a shift on a child edge or not.

Simulation procedure

We used the lizard tree from the `phylolm` package and associated trait data on these lizard species (just the first trait, which is the first PC axis from a PCA). This trait was analyzed to estimate the shifts in trait evolution using the function `estimate_shift_configuration` from the `l1ou` package. The analysis takes about 2 minutes, so the results were saved in an R data file:

```
data(lizard.tree, lizard.traits)
lizard <- adjust_data(lizard.tree, lizard.traits[,1])
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y)
save(eModel,  file = "eModel.RData")
```

The estimated shift configuration can be loaded quickly now:

```
load("eModel.RData")
```

Next, variables are set up to prepare the simulation.

```
truealpha=eModel$alpha
y0=eModel$intercept
truetheta = y0 + l1ou:::convert_shifts2regions(eModel$tree,
                                eModel$shift.configuration, eModel$shift.values)
nShifts   = length(eModel$shift.configuration) # Total number of shifts
n_tips=length(eModel$tree$tip.label) # Total number of tips
sigma2=eModel$sigma2
shiftnode= eModel$tree$edge[eModel$shift.configuration,1]-n_tips # internal nodes with shift
othernode=(1:(n_tips-1))[-shiftnode] # Other nodes: those without a true shift
```

The tree has 100 tips and 8 nodes with a shift at one of its child edges, numbered $72, 51, 41, 63, 93, 61, 84, 18$ (`shiftnode`). At these shifts, the optimal value changed by: -2.91, -2.97, -5.5, -2.39, -2.61, -1.91, -3.7, -3.49 (shift values). The other parameters were estimated to be $\alpha = 0.606898$ (`truealpha`) and $\sigma^2 = 0.0625187$ and an "intercept" `y0` of 0.2488096. Together with the changes, this intercept gives the following optimal

values `truetheta`, one for each edge: $\theta =$ 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, -2.36, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, -3.45, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, -1.66, -1.66, -1.66, -1.66, -4.57, -4.57, -4.57, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -1.66, -4.05, -1.66, -1.66, -1.66, 0.25, 0.25, 0.25, -2.72, -2.72, -2.72, -2.72, -2.72, -2.72, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, -2.72, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, -5.25, 0.25, -3.24, 0.25.

We used this model to simulate new data using the `rTraitCont` function from the `phylolm` package. Below, `RE` is the result of the function `sqrt_OU_covariance`, which calculates the square-root of the phylogenetic covariance matrix with a recursive algorithm, which traverses the tree once. `C.IH` is the inverse square root of the phylogenetic covariance matrix, and `C.H` is the square-root of the phylogenetic covariance matrix. Finally, `YY` contains the contrasts at all nodes. These matrices and contrasts were obtained using the true value of α, the same value used to simulate the data. This is an ideal situation when α is known without error.

```
n_sim=100000
boot_table=matrix(nrow=n_sim, ncol=n_tips, data=NA)
RE = sqrt_OU_covariance(eModel$tree, alpha=truealpha,
                        root.model = eModel$l1ou.options$root.model,
                        check.order=F, check.ultrametric=F)
C.IH  = t(RE$sqrtInvSigma)
C.H   = RE$sqrtSigma
for (i in 1:n_sim) {
  Y  <- rTraitCont(eModel$tree, "OU", theta=truetheta,
                 alpha=truealpha,
                 sigma=sqrt(eModel$sigma2), root.value=y0)
  YY    = C.IH%*%(Y - eModel$mu)
  boot_table[i,]=YY
}
save(boot_table, file="boot_table.RData")
```

The simulation of 10^5 data sets took 1 minute and the results were saved in the `boot_table.RData` file.

Results: visualizations of contrast distributions

Contrasts on shift nodes

We first focus on the distribution, mean and standard deviations of contrasts associated with shift nodes, that is, nodes for which one child edge has a shift.

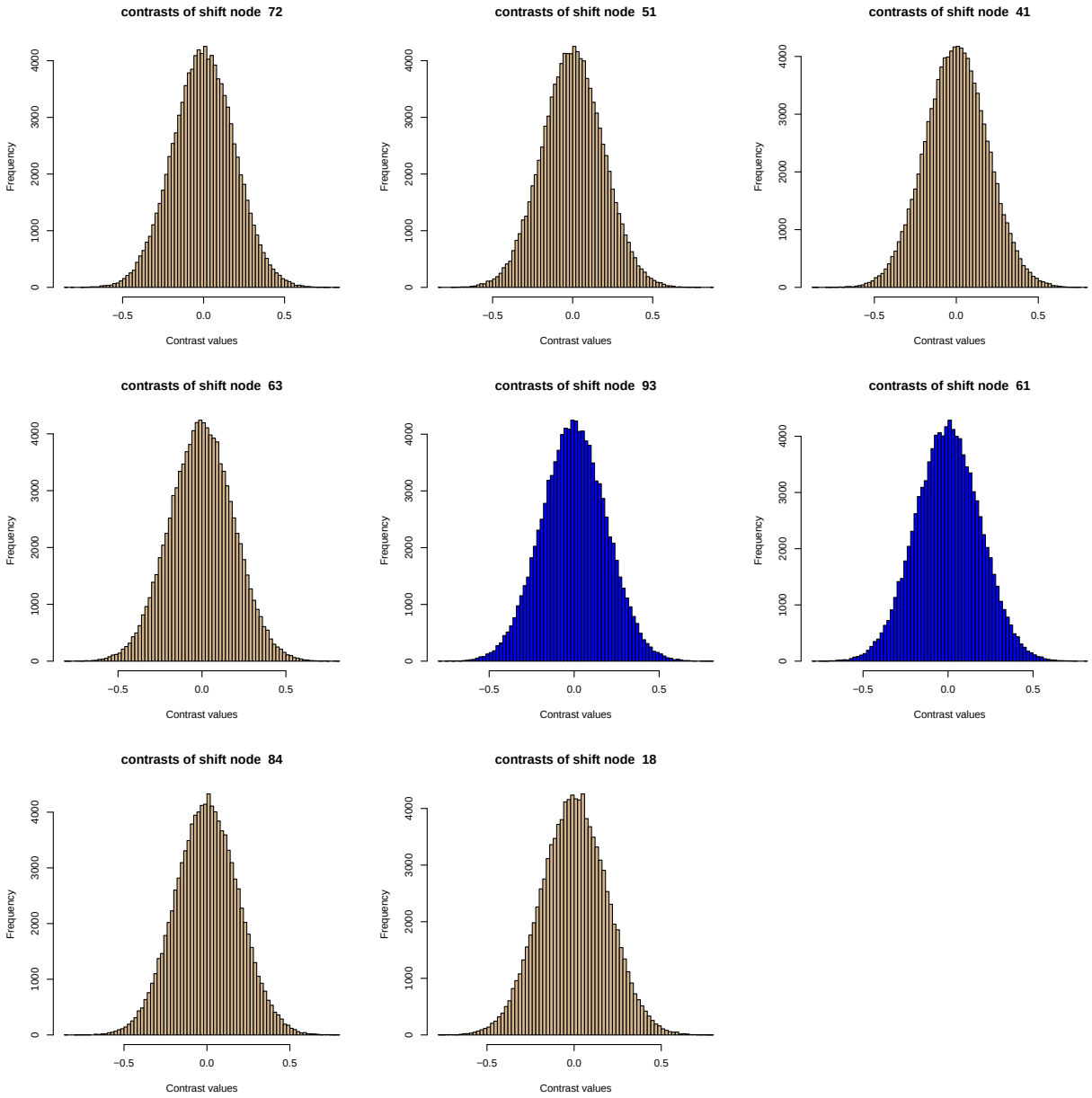
```
meanlist= colMeans(boot_table)[shiftnode] # mean contrasts at shift nodes
round(meanlist,4)
```

```
## [1]  5e-04 -2e-04 -3e-04 -1e-03 -9e-04 -5e-04  8e-04  5e-04
```

```
sdlist=apply(boot_table,2,sd)[shiftnode] # sd contrasts at shift nodes
round(sdlist, 4)
```

```
## [1] 0.1904 0.1906 0.1901 0.1907 0.1903 0.1898 0.1907 0.1900
```

```
layout(matrix(1:9,ncol=3,nrow=3, byrow=TRUE))
for (j in shiftnode) {
  mycol="tan"
  if (j %in% c(93,61)){ mycol="blue"}
  hist(boot_table[,j],col=mycol,xlab="Contrast values", main=
         paste("contrasts of shift node " , j),breaks=100)
}
```



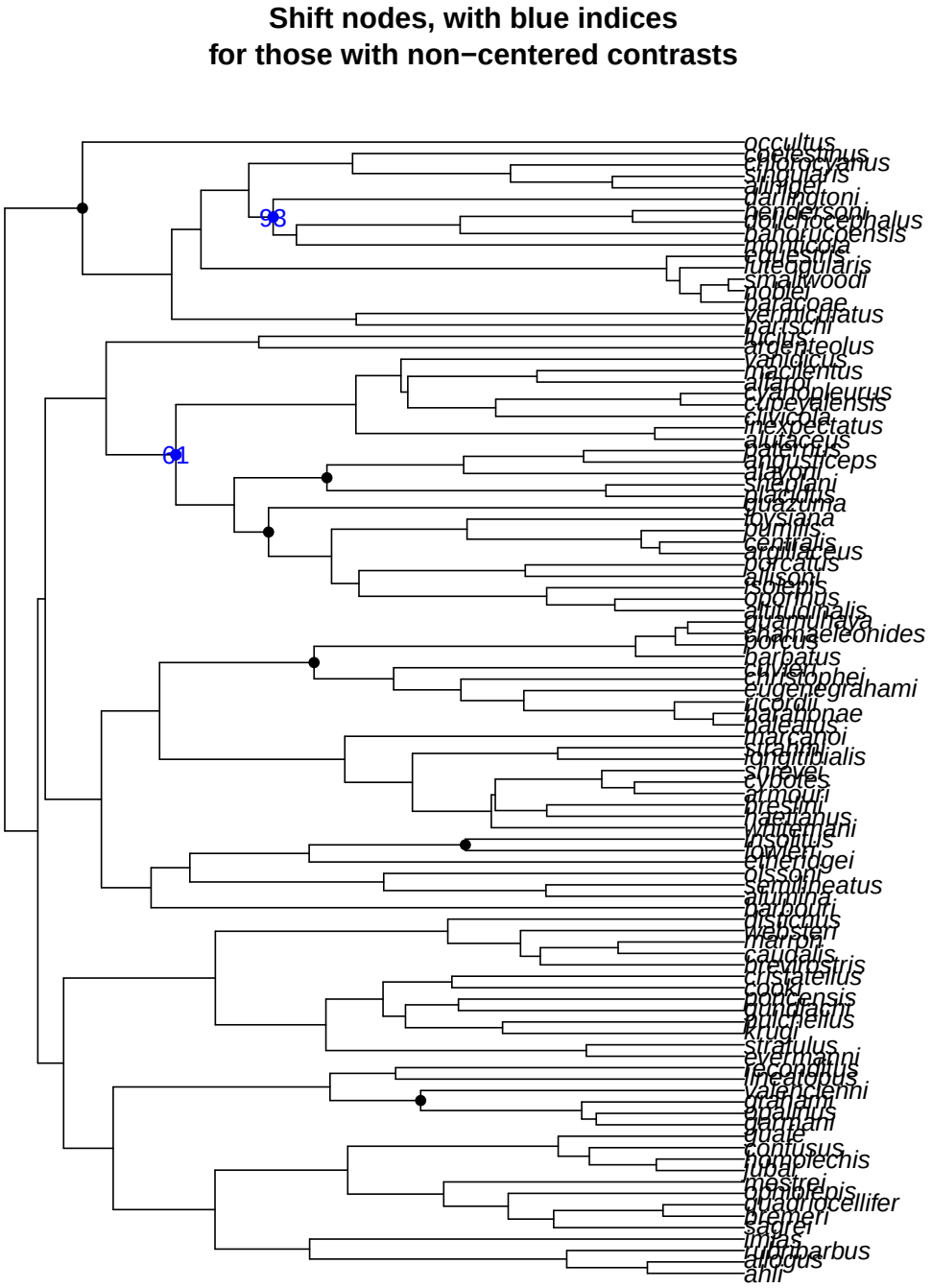
All the histograms are bell-shaped with a peak close to 0 except for the 5th and 6th shift nodes (in blue). The mean contrast at node 61 is 0, which is far away from 0 relative to its standard deviation (0.19). Accordingly, the proportion of contrast values below 0 is only 0.50108 (instead of 0.5). The mean contrast at node 93 is -0.001, which is also far away from 0, but less so. The proportion of negative contrasts, for all shift nodes, are:

```r
apply(boot_table,2,function(v){sum(v<0)/n_sim})[shiftnode]
```

```
## [1] 0.49796 0.50001 0.50012 0.50151 0.50229 0.50108 0.49851 0.49889
```

Let's visualize the 2 nodes with non-zero mean contrasts on the tree:

```r
layout(matrix(1, 1))
plot(eModel$tree,main="Shift nodes, with blue indices\nfor those with non-centered contrasts")
nodelabels("",shiftnode+n_tips,pch=16,frame="none") # Add n_tips for the indices of shiftnodes
# because 1:n_ntips are external nodes
nodelabels(shiftnode[5],shiftnode[5]+n_tips,pch=16,col="blue",frame="none")
nodelabels(shiftnode[6],shiftnode[6]+n_tips,pch=16,col="blue",frame="none")
```



The two nodes with contrasts far away from 0 are close to the root. We do not understand the reason why their means are not zero.

Mean, SD and distribution of contrasts at non-shift nodes

```
options(digits=4)
round(colMeans(boot_table)[othernode], 3) # contrast mean at all non-shift nodes
```

```
##  [1]  0.000  0.000  0.000  0.001  0.000  0.000  0.000 -0.001  0.000  0.000
## [11] -0.001  0.000  0.000 -0.001  0.000  0.000  0.000 -0.001  0.001  0.000
## [21]  0.000  0.001  0.000  0.000  0.000  0.000 -0.001  0.001 -0.001  0.000
## [31]  0.000  0.001  0.000  0.000  0.001  0.001  0.000  0.001  0.000  0.000
## [41]  0.001  0.000 -0.001 -0.001  0.000  0.000  0.001  0.000  0.000  0.001
## [51]  0.000  0.000  0.000  0.001 -0.001 -0.001  0.000  0.000  0.001  0.000
## [61]  0.000  0.000 -0.001 -0.001 -0.001 -0.001 -0.001  0.001  0.000  0.000
## [71]  0.000  0.000  0.000 -0.001 -0.001  0.000  0.000  0.000  0.001  0.000
## [81]  0.000  0.001  0.000  0.001  0.000  0.000 -0.001  0.000  0.000  0.000
## [91] -0.001
```

```
# below: proportion of negative contrasts for non-shift nodes. should be 0.5
pro0other=apply(boot_table,2,function(v){sum(v<0)/n_sim})[othernode]
round(pro0other, 3)
```

```
##  [1] 0.502 0.501 0.500 0.499 0.499 0.501 0.502 0.502 0.500 0.500 0.504
## [12] 0.500 0.500 0.503 0.500 0.500 0.500 0.500 0.498 0.500 0.500 0.499
## [23] 0.498 0.500 0.498 0.502 0.501 0.498 0.501 0.499 0.501 0.500 0.499
## [34] 0.500 0.497 0.498 0.499 0.498 0.500 0.502 0.498 0.498 0.501 0.500
## [45] 0.500 0.501 0.497 0.499 0.500 0.496 0.500 0.500 0.501 0.497 0.503
## [56] 0.501 0.500 0.500 0.496 0.500 0.500 0.499 0.501 0.502 0.502 0.502
## [67] 0.501 0.498 0.500 0.499 0.499 0.501 0.501 0.502 0.501 0.500 0.500
## [78] 0.501 0.499 0.500 0.499 0.499 0.500 0.499 0.500 0.498 0.501 0.500
## [89] 0.500 0.500 0.502
```

The contrast means seem to be way off for some nodes, but correct (≈ 0) for a bunch of nodes. To count how many nodes have an unusually low mean, we can look at the nodes with 70% or more negative contrasts (0 such nodes out of 91 non-shift nodes). There are also 0 nodes with an unusually high mean, indicated by 30% or fewer negative contrasts.

```
apply(boot_table,2,sd)[othernode]      # contrast SD at all non-shift nodes
```

```
##  [1] 0.1900 0.1904 0.1901 0.1909 0.1903 0.1907 0.1911 0.1897 0.1903 0.1907
## [11] 0.1904 0.1902 0.1901 0.1904 0.1903 0.1908 0.1903 0.1894 0.1906 0.1904
## [21] 0.1896 0.1906 0.1902 0.1895 0.1901 0.1905 0.1903 0.1902 0.1894 0.1903
## [31] 0.1907 0.1903 0.1902 0.1905 0.1908 0.1896 0.1903 0.1899 0.1904 0.1903
## [41] 0.1902 0.1895 0.1894 0.1896 0.1902 0.1910 0.1902 0.1913 0.1904 0.1904
## [51] 0.1901 0.1910 0.1906 0.1899 0.1900 0.1903 0.1903 0.1900 0.1902 0.1900
## [61] 0.1907 0.1902 0.1910 0.1905 0.1907 0.1901 0.1897 0.1905 0.1902 0.1904
## [71] 0.1902 0.1909 0.1904 0.1906 0.1901 0.1903 0.1899 0.1904 0.1893 0.1902
## [81] 0.1891 0.1902 0.1905 0.1895 0.1902 0.1903 0.1906 0.1900 0.1909 0.1908
## [91] 0.1908
```

```
convar = mean(apply(boot_table,2,var)) # constrast variance, assumed to be the same at all nodes
```

The distribution of contrasts seems normal for all nodes (based on histograms that are not shown).

The contrast standard deviation seems to be equal at all nodes (up to simulation stochasticity), but incorrect because we expected the contrast variance to be `sigma2 =` 0.0625. An estimate of their common variance is quite different: `convar =` 0.0362, with square-root (SD) of 0.1903.

Phylogenetic position of nodes with non-centered contrasts

Here we are showing the placement of nodes with non-zero contrast means, both with or without shifts. We quantified "non zero" mean by looking at the the proportion of negative contrasts (expected to be 0.5), and by looking at whether this proportion was below 0.3 (high mean relative to SD) or above 0.7 (low mean relative to SD). The thresholds 0.3 and 0.7 did not have much influence because the proportion of negative contrasts was either pretty close to 0.5 or pretty far:

```
prop.below0 = apply(boot_table,2,function(v){sum(v<0)/n_sim}) # proportion of negative contrasts
round(prop.below0,2)
```

```
##   [1] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
##  [18] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
##  [35] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
##  [52] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
##  [69] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
##  [86] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
```

When this proportion was not 0.50 (up to 2 decimal points), it was most often above 0.70 or below 0.30, except for 3 exception at 0.68, 0.35 and 0.48.

```
weird0.3 = which(prop.below0 < .3) # indices of nodes with a high mean
weird0.3
```

```
## integer(0)
```

```
weird0.7= which( prop.below0 > .7) # indices of nodes with a low mean
weird0.7
```

```
## integer(0)
```

```
#plot(eModel$tree, main="Nodes with a proportion of negative contrasts < 0.3 or > 0.7")
#nodelabels("",weird0.3+n_tips,pch=16,col="blue",frame = "none")
#nodelabels("",weird0.7+n_tips,pch=16,col="green",frame = "none")
#mtext("Blue: nodes with 30% or fewer negative contrasts.
#Green: nodes with 70% or more negative contrasts.",side=1,pch=16,line=2)
```

Some of these nodes with non-zero contrast means seem to cluster together (near the top of the graph), but this is a weak trend only.

Why is the contrast mean incorrect?

This might be the result of a bug. We can think of potential problems at these steps:

1. bug or incorrect use of the simulation function
2. bug in the calculation of the (inverse) square root covariance matrix
3. bug in the calculation of the `mu` vector, mean at the tips.

We did some calculations below to see if item 1 might be a problem.

```
options(digits=7)
# checking simulation with very large alpha, comparing Y to mu
range(abs(
  rTraitCont(eModel$tree, "OU", theta=truetheta,
             alpha=100000, sigma=sqrt(eModel$sigma2),
             root.value = y0)
  - eModel$mu))
```

```
## [1] 5.569626e-06 4.375359e+00
```

```
al = 10^10
range(abs(
  rTraitCont(eModel$tree, "OU", theta=truetheta,
             alpha=al, sigma=sqrt(eModel$sigma2),
             root.value = y0)
  - eModel$mu))
```

```
## [1] 9.189056e-09 4.375463e+00
```

When we simulate data using a huge α, the mean at a given tip i should be equal to the optimal value at the external edge to tip i: The bigger the α, the stronger the driving force towards the optimal value. When we change α from 100,000 to 10^{10}, the difference between Y and `mu` is getting a little bit smaller, but the difference is not super small. This difference can be explained by the trait variance, which is close to γ when α is very large, with standard deviation $\sqrt{\gamma} = \sigma/\sqrt{2\alpha} = 1.7680308 \times 10^{-6}$ when $\alpha = 10^{10}$.

Actually, `eModel$mu` was calculated when fitting the model to the lizard data and using $\alpha = 0.606898$. So this `mu` is not the optimal value, and a simulation using an infinite α *should not* give traits equal to `mu` but equal to `theta` at the external edges.

```
extedges=which(eModel$tree$edge[,2] <= 100) # indices of external edges
x=cbind(extedges,eModel$tree$edge[,2][extedges]) # combine indices: external edges with child nodes
edge_ind=x[order(x[,2]),][,1] # re-order external edges based on tips
thetatips=truetheta[edge_ind]  # theta values for external edges
max(abs(eModel$mu-thetatips))  # max absolute difference between mu and tip theta's
```

```
## [1] 4.375465
```

In the output above, we see that the optimal values at the tips are almost equal to the means `mu` expected with $\alpha = 0.606898$. This is very surprising because this α is quite small for a tree height of 1. The maximum difference between `thetatips` and `eModel$mu` is much smaller than expected: 4.3754648.

We also compared the trait vector generated by `rTrait` in the `phylolm` package with the trait vector generated by `rTraitCont` from the `l1ou` package, using a single simulation that had a very large α but no shifts, because `rTrait` cannot simulate shifts.

```
mu = rep(y0,n_tips)
range(abs(
  rTrait(n=1, phy=eModel$tree, model="OU",
         list(optimal.value=y0, sigma2=eModel$sigma2,
              alpha=al, ancestral.state=y0))
  - mu))
```

```
## [1] 5.511357e-08 4.368440e-06
```

```
range(abs(
  rTraitCont(eModel$tree, "OU", theta=y0, sigma=sqrt(eModel$sigma2),
             alpha=al, root.value = y0)
  - mu))
```

```
## [1] 2.050289e-08 5.679689e-06
```

The differences between the simulated traits and the expected value are small for both simulation tools, of the order of the trait's standard deviation. So it looks like the simulation function `rTraitCont` is giving the correct mean when there are no shifts in the optimal value, at least when α is large.

We did not investigate the other potential items 2 and 3, but our conclusion is that item 1 might be the issue: not through the simulation function `rTraitCont`, but through a wrong calculation of the trait means at the tips, `mu`. This vector was part of the output of `estimate_shift_configuration`, which we had in `eModel$mu`. The bug might be in the function that estimates a shift configuration, then, when it calculates the tip means...

Why is the variance incorrect?

Could it be a wrong choice in the model for the value at the root of the tree, when calculating the square-root of the covariance matrix, or a bug due to using a tree where edges are not in postorder? No, based on the output below.

```
# checking if we have the correct transformation
RE = sqrt_OU_covariance(eModel$tree, alpha=truealpha,
                        root.model = eModel$l1ou.options$root.model)
newtree = reorder(eModel$tree, "post")
all(newtree$edge == eModel$tree$edge)
```

```
## [1] TRUE
```

```
all(newtree$tip.label == eModel$tree$tip.label)
```

```
## [1] TRUE
```

```
RE1 = sqrt_OU_covariance(eModel$tree, alpha=truealpha, root.model="OUfixedRoot")
RE2 = sqrt_OU_covariance(eModel$tree, alpha=truealpha, root.model="OUrandomRoot")
all(RE1$sqrtInvSigma==RE2$sqrtInvSigma)
```

```
## [1] FALSE
```

The variances of all contrasts are all roughly equal, around 0.0362017. Why this particular value instead of the expected 0.0625187?

$\gamma = \sigma^2/(2\alpha)$ is the stationary variance: the variance of the OU process $(Y_t)_t$ when t goes to ∞. After a fixed time of evolution T, $\gamma(1 - e^{-2\alpha T})$ is the variance of Y_t if the root value Y_0 is fixed with variance 0.

Is the contrast variance equal to $\sigma^2(1 - e^{-2\alpha T})$ where T is the height of the tree? We have $T = 1$ here (code not shown).

```
(1-exp(-2*truealpha*1)) #  factor that differs between OU fixed root vs. random root
```

```
## [1] 0.7029325
```

```
convar / eModel$sigma2
```

```
## [1] 0.5790547
```

No. These ratios are different. Is the contrast variance, as currently calculated by the `l1ou` bootstrap function `sqrt_OU_covariance` equal to the stationary variance γ? No, based on the output below.

```
eModel$sigma2/ (2*truealpha) # this is gamma
```

```
## [1] 0.05150673
```

```
convar
```

```
## [1] 0.03620172
```

Is the contrast variance equal to $\gamma(1 - e^{-2\alpha T})$?

```
eModel$sigma2 * (1-exp(-2*truealpha*1)) / (2*truealpha)
```

```
## [1] 0.03620576
```

```
convar
```

```
## [1] 0.03620172
```

Yes, almost. The difference might be due to simulation randomness only. This gives us an idea to fix the contrast calculation for the bootstrap in `l1ou` package. This variance mismatch might not be a problem in the bootstrap function, if the square-root of the phylogenetic covariance `C.H`, which is used to get tip traits from node contrasts, is the inverse of the matrix `C.IH`, which is used to get node contrasts from tip traits.

But this variance mismatch might be a problem for other functions that use `sqrt_OU_covariance`. The scaling of the output matrices should at least be documented in the manual page for this function, even if this scaling is not changed in the code.