SX: Regression Distribution Results

Label switching and the impact on regression results

One of the quirks of hidden Markov models (HMMs) is that the labeling of rate classes is arbitrary. Not in the sense that labels are meaningless, but rate that it makes no difference whether something is labeled as A or B (a rate by any other name would smell as sweet).

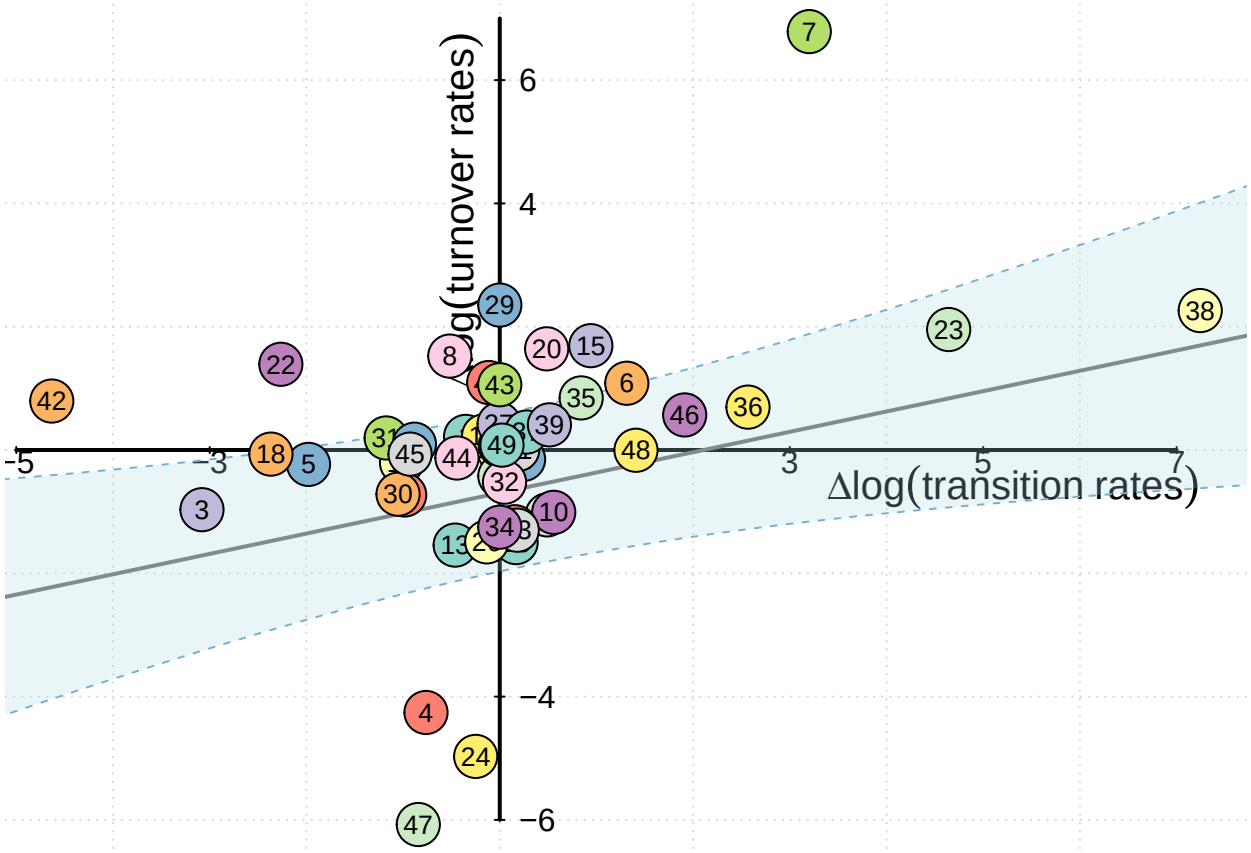
This presents a problem for optimization since this means that there are two parameterizations with exactly the same likelihood (swapping rate class A and rate class B). Within packages like corHMM or hisse we have tried to circumvent this issue by generally forcing one rate class to be faster than another. This is done internally to help with the likelihood search but should also provide more consistency in the output results. Nonetheless, this issue could still impact our analysis of parameter estimates.

The impact on our regression estimates would be due to the fact that whether we are subtracting B from A is not consistent across our datasets. This means that you could comfortably flip, for any given clade, whether you are subtracting A from B or B from A. In this vignette, I will run a randomization test to determine whether the default results are trustworthy

Let's review our initial results.

```
##
## Call:
## phylolm(formula = d_turns ~ d_trans, data = lm_dat, phy = phy_bb,
##     boot = 1000)
##
##    AIC logLik
## 197.12 -95.56
##
## Raw residuals:
##     Min      1Q  Median      3Q     Max
## -5.1012  0.2260  0.7647  1.2625  6.4253
##
## Mean tip height: 135.9122
## Parameter estimate(s) using ML:
## sigma2: 0.0296571
##
## Coefficients:
##             Estimate    StdErr   t.value lowerbootCI upperbootCI p.value
## (Intercept) -0.695293  0.609029 -1.141642   -1.842716      0.4672 0.25939
## d_trans      0.330045  0.135031  2.444214    0.068137      0.5975 0.01832 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-squared: 0.1128    Adjusted R-squared: 0.0939
##
## sigma2: 0.0296571
##       bootstrap mean: 0.02847662 (on raw scale)
##                       0.02786147 (on log scale, then back transformed)
##       bootstrap 95% CI: (0.01781035,0.04110257)
##
```

```
## Parametric bootstrap results based on 1000 fitted replicates
```



The results of our initial regression are shown above. We find a significant non-zero slope. However, as discussed above, for each clade, rate class a and rate class b are interchangeable. If we shuffle the rate classes around randomly, we will get a different result.

Let's look at the raw data as a starting point. The data shown below are the transition rates for all possible transitions in our model with the order as follows: $q_{00:01(A)}, q_{01:00(A)}, q_{01:11(A)}, q_{11:01(A)}, q_{00:01(B)}, q_{01:00(B)}, q_{01:11(B)}, q_{11:01(B)}$

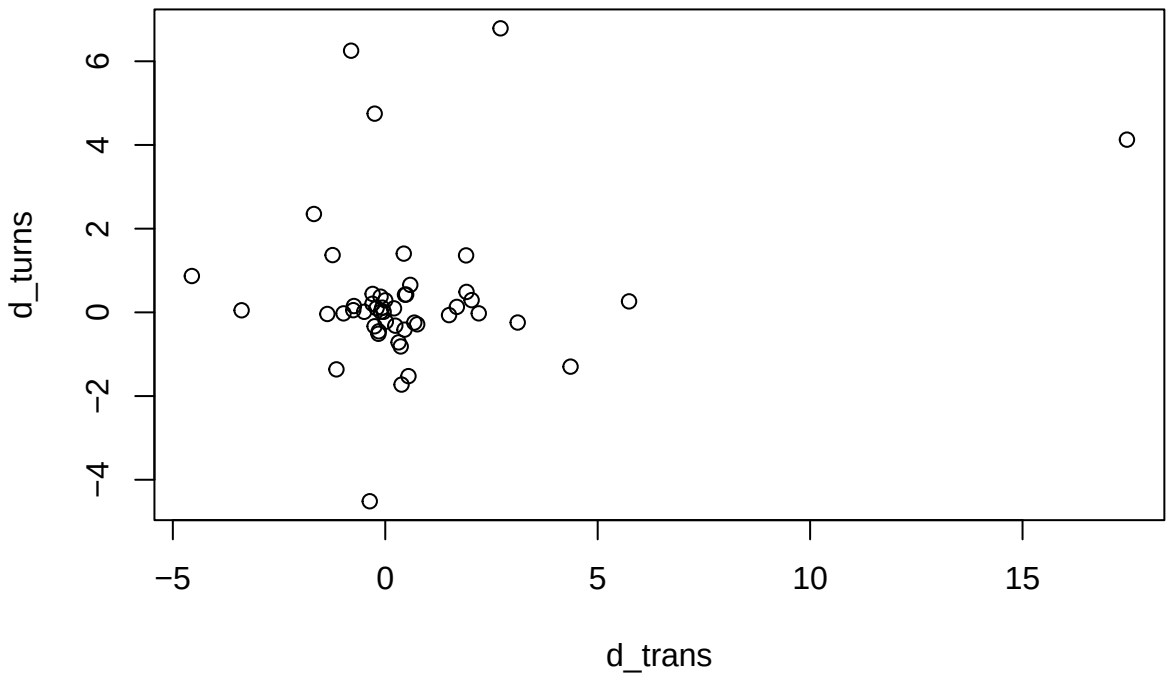
```
demo_df <- data.frame( a = head(rate_class_a_rate),
                       b = head(rate_class_b_rate))
print(demo_df)
```

```
##          a.1         a.2        a.3        a.4        b.1       b.2        b.3
## 1  1.3326693  0.91372757  0.2759942  0.1598574 0.82861701 0.5864088  0.2800526
## 2  4.9219393  5.42387691 79.7965024 34.8578563 4.85165113 5.0091649 79.5819495
## 3  7.3769518 74.20552435 13.9641454 74.4910010 0.02403847 0.2138439  1.1785288
## 4  1.3308422 12.88786020  0.4080388  9.9758091 0.42336915 3.9711361  0.4093351
## 5 15.6736533  8.11807005 19.6690682  2.6754370 2.28548958 2.4752178  1.3565549
## 6  0.1514647  0.09133707  0.3293246  0.1734857 1.24066757 0.9679194  0.4678052
##           b.4
## 1  0.18960221
## 2 34.95402507
## 3  6.40132668
## 4  6.66990903
## 5  0.26680956
## 6  0.09777796
```

Random shuffling for a null distribution

To begin with let's try and examine a null expectation. In this case, if we were to randomly shuffle all transition rates and turnover rates (mixing rate class A and B) we would not expect to find an association. By shuffling rate classes any estimated differences between rate class A and rate class B should be removed.

```
shuffled_rates <- cbind(rate_class_a_rate, rate_class_b_rate)
shuffled_rates <- apply(shuffled_rates, 1,
                        function(x) x[sample(1:8,8)])
shuffled_rate_class_a_rate <- t(shuffled_rates)[,1:4]
shuffled_rate_class_b_rate <- t(shuffled_rates)[,5:8]


shuffled_turns <- cbind(rate_class_a_turn, rate_class_b_turn)
shuffled_turns <- apply(shuffled_turns, 1,
                        function(x) x[sample(1:6,6)])
shuffled_rate_class_a_turn <- t(shuffled_turns)[,1:3]
shuffled_rate_class_b_turn <- t(shuffled_turns)[,4:6]

d_turns <- d_trans <- c()
for(i in 1:49){
  d_trans <- (c(d_trans, log(rowMeans(shuffled_rate_class_b_rate)[i]) -
                  log(rowMeans(shuffled_rate_class_a_rate)[i])))
  d_turns <- (c(d_turns, log(rowMeans(shuffled_rate_class_b_turn)[i]) -
                  log(rowMeans(shuffled_rate_class_a_turn)[i])))
}

lm_dat_null <- data.frame(row.names = gsub(" .*", "", plot_data[,1]),
                      d_trans = d_trans,
                      d_turns = d_turns)
fit_null = phylolm(d_turns ~ d_trans, data=lm_dat_null, phy=phy_bb, boot = 0)
plot(lm_dat_null)
```




```
summary(fit_null)
```

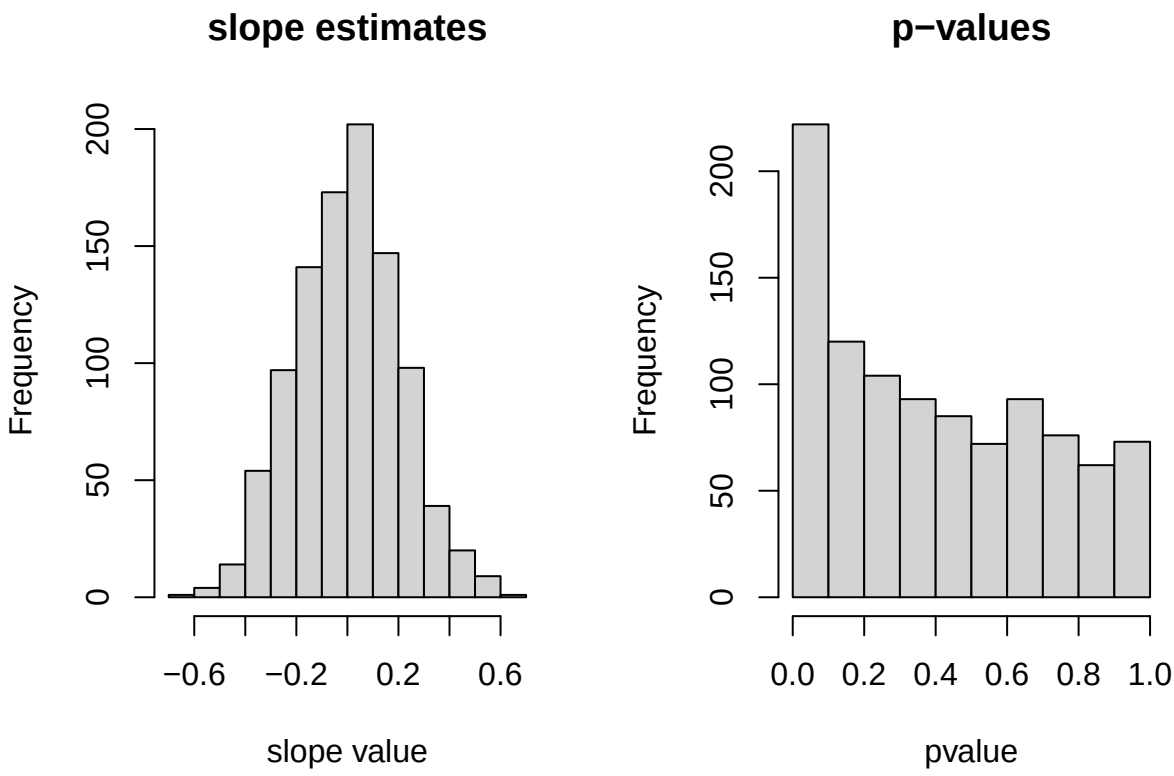
```
##
## Call:
## phylolm(formula = d_turns ~ d_trans, data = lm_dat_null, phy = phy_bb,
##     boot = 0)
##
##    AIC logLik
## 205.24 -99.62
##
## Raw residuals:
##     Min      1Q  Median      3Q     Max
## -4.4954 -0.4102  0.0480  0.3597  6.4116
##
## Mean tip height: 135.9122
## Parameter estimate(s) using ML:
## sigma2: 0.0350064
##
## Coefficients:
##             Estimate   StdErr t.value p.value
## (Intercept) 0.028926 0.666545  0.0434  0.9656
## d_trans     0.127531 0.094949  1.3432  0.1857
##
## R-squared: 0.03697   Adjusted R-squared: 0.01648
```

This suggests this shuffling procedure did succesfully remove the signal as the slope is not significantly different from 0. the Let's iterate this shuffling procedure 1000 times and create a null distribution of slope estimates.

```
all_fits_shuffled <- list()
for(j in 1:1000){
  shuffled_rates <- cbind(rate_class_a_rate, rate_class_b_rate)
  shuffled_rates <- apply(shuffled_rates, 1,
                          function(x) x[sample(1:8,8)])
  shuffled_rate_class_a_rate <- t(shuffled_rates)[,1:4]
  shuffled_rate_class_b_rate <- t(shuffled_rates)[,5:8]


  shuffled_turns <- cbind(rate_class_a_turn, rate_class_b_turn)
  shuffled_turns <- apply(shuffled_turns, 1,
                          function(x) x[sample(1:6,6)])
  shuffled_rate_class_a_turn <- t(shuffled_turns)[,1:3]
  shuffled_rate_class_b_turn <- t(shuffled_turns)[,4:6]

  d_turns <- d_trans <- c()
    for(i in 1:49){
      d_trans <- (c(d_trans, log(rowMeans(shuffled_rate_class_b_rate)[i]) -
                      log(rowMeans(shuffled_rate_class_a_rate)[i])))
      d_turns <- (c(d_turns, log(rowMeans(shuffled_rate_class_b_turn)[i]) -
                      log(rowMeans(shuffled_rate_class_a_turn)[i])))
    }
  lm_dat_shuffle <- data.frame(row.names = gsub(" .*", "", plot_data[,1]),
                       d_trans = d_trans,
                       d_turns = d_turns)
  fit_shuffle = phylolm(d_turns ~ d_trans, data=lm_dat_shuffle, phy=phy_bb, boot = 0)
  all_fits_shuffled[[j]] <- fit_shuffle
}
```

Distribution of the p-values and slope estimates.

```
shuffled_est <- lapply(all_fits_shuffled,
                       function(x) summary(x)$coefficients[2,c(1,4)])
shuffled_est_df <- do.call(rbind,shuffled_est)

par(mfrow=c(1,2))
hist(shuffled_est_df[,1], main = "slope estimates", xlab = "slope value")
hist(shuffled_est_df[,2], main = "p-values", xlab = "pvalue")
```

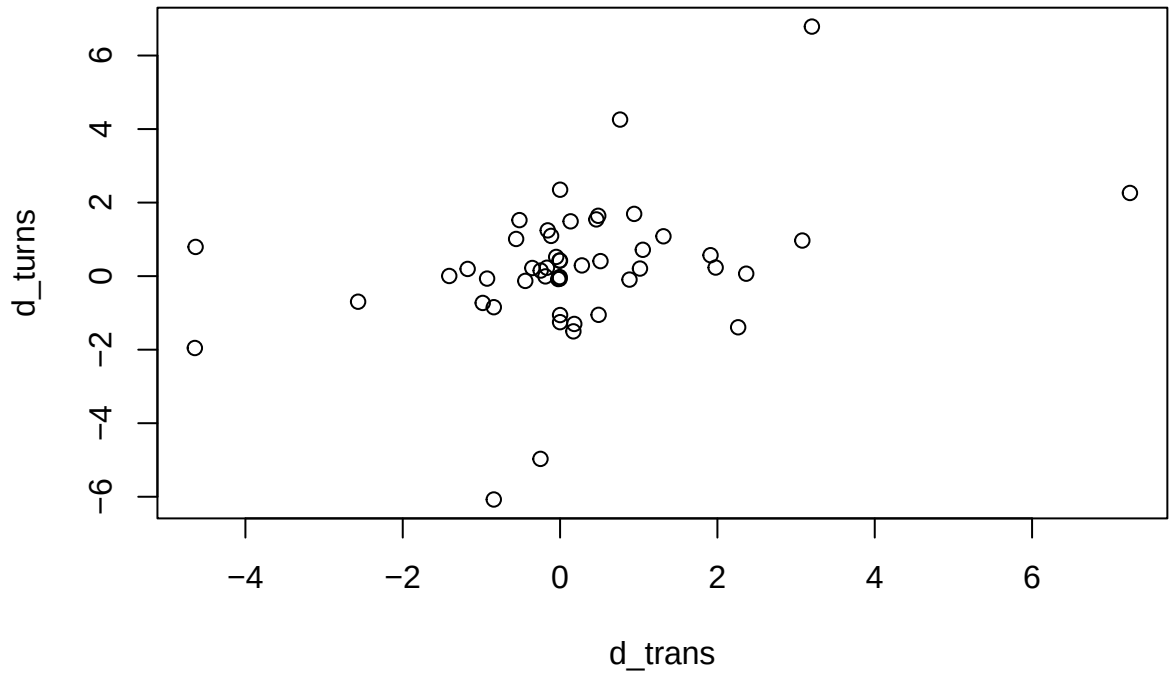


Random shuffling for robustness against label switching

With a null expectation established let's shuffle rate classes in a way that should preserve the signal by randomly shuffling labels. The shuffling will randomly select between 1 and 49 clades and then randomly choose which of those clades will swap the labels on the rate class A and B.

```
no_to_swap <- sample(1:49, 1)
to_swap <- sample(1:49, no_to_swap, FALSE)
d_turns <- d_trans <- c()
for(i in 1:49){
  if(i %in% to_swap){
    d_trans <- (c(d_trans, log(rowMeans(rate_class_a_rate)[i]) -
                  log(rowMeans(rate_class_b_rate)[i])))
    d_turns <- (c(d_turns, log(rowMeans(rate_class_a_turn)[i]) -
                  log(rowMeans(rate_class_b_turn)[i])))
  }else{
    d_trans <- (c(d_trans, log(rowMeans(rate_class_b_rate)[i]) -
                  log(rowMeans(rate_class_a_rate)[i])))
    d_turns <- (c(d_turns, log(rowMeans(rate_class_b_turn)[i]) -
                  log(rowMeans(rate_class_a_turn)[i])))
  }
}

lm_dat_shuffle <- data.frame(row.names = gsub(" .*", "", plot_data[,1]),
                     d_trans = d_trans,
                     d_turns = d_turns)
```

```
fit_shuffle = phylolm(d_turns ~ d_trans, data=lm_dat_shuffle, phy=phy_bb, boot = 0)
plot(lm_dat_shuffle)
```



If we look at the data we can see that the data has been shuffled.

```
summary(fit_shuffle)
```

```
##
## Call:
## phylolm(formula = d_turns ~ d_trans, data = lm_dat_shuffle, phy = phy_bb,
##     boot = 0)
##
##    AIC logLik
## 201.83 -97.92
##
## Raw residuals:
##     Min      1Q  Median      3Q     Max
## -6.0608 -0.5640 -0.0083  0.4930  5.6679
##
## Mean tip height: 135.9122
## Parameter estimate(s) using ML:
## sigma2: 0.03265196
##
## Coefficients:
##             Estimate  StdErr t.value p.value
## (Intercept)  0.22220 0.63955  0.3474 0.72981
## d_trans      0.28003 0.13275  2.1095 0.04026 *
```

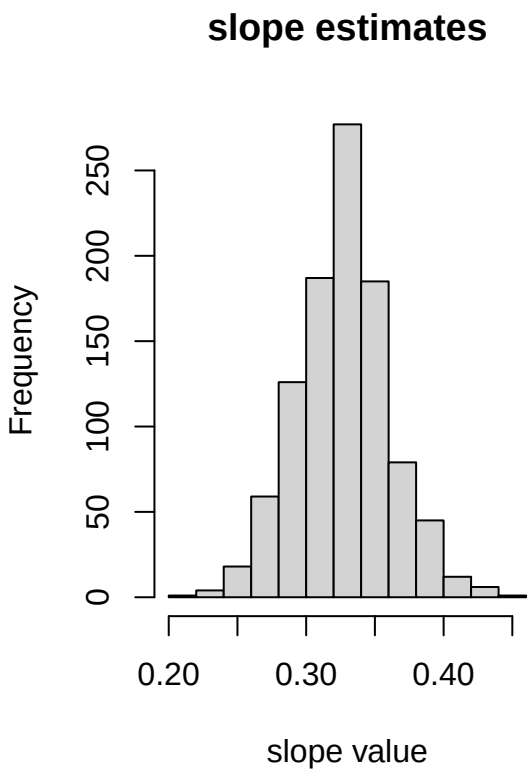
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-squared: 0.08649   Adjusted R-squared: 0.06706
```

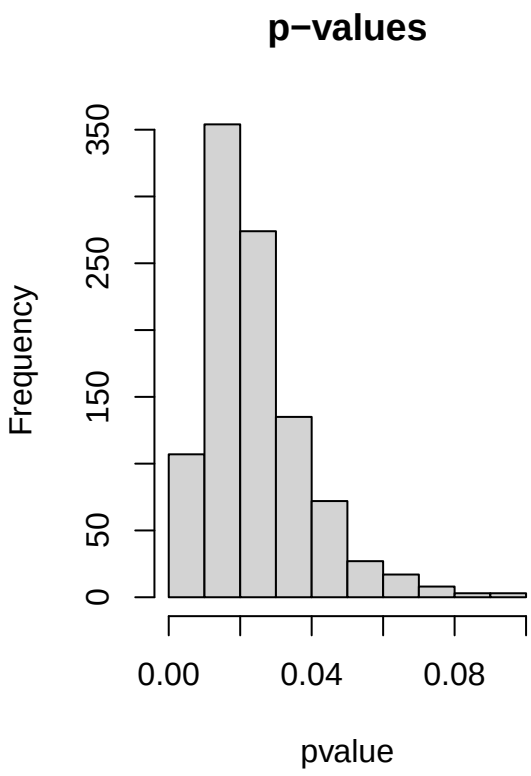
However, we are still able to recover a significantly positive slope. Lets repeat this procedure 1000 times.

```
all_fits <- list()
for(j in 1:1000){
  no_to_swap <- sample(1:49, 1)
  to_swap <- sample(1:49, no_to_swap, FALSE)
  d_turns <- d_trans <- c()
  for(i in 1:49){
    if(i %in% to_swap){
      d_trans <- (c(d_trans, log(rowMeans(rate_class_a_rate)[i]) -
                      log(rowMeans(rate_class_b_rate)[i])))
      d_turns <- (c(d_turns, log(rowMeans(rate_class_a_turn)[i]) -
                      log(rowMeans(rate_class_b_turn)[i])))
    }else{
      d_trans <- (c(d_trans, log(rowMeans(rate_class_b_rate)[i]) -
                      log(rowMeans(rate_class_a_rate)[i])))
      d_turns <- (c(d_turns, log(rowMeans(rate_class_b_turn)[i]) -
                      log(rowMeans(rate_class_a_turn)[i])))
    }
  }

  lm_dat_tmp <- data.frame(row.names = gsub(" .*", "", plot_data[,1]),
                       d_trans = d_trans,
                       d_turns = d_turns)
  fit_tmp = phylolm(d_turns ~ d_trans, data=lm_dat_tmp, phy=phy_bb, boot = 0)
  all_fits[[j]] <- fit_tmp
}
```

Let's plot the resulting distributions.

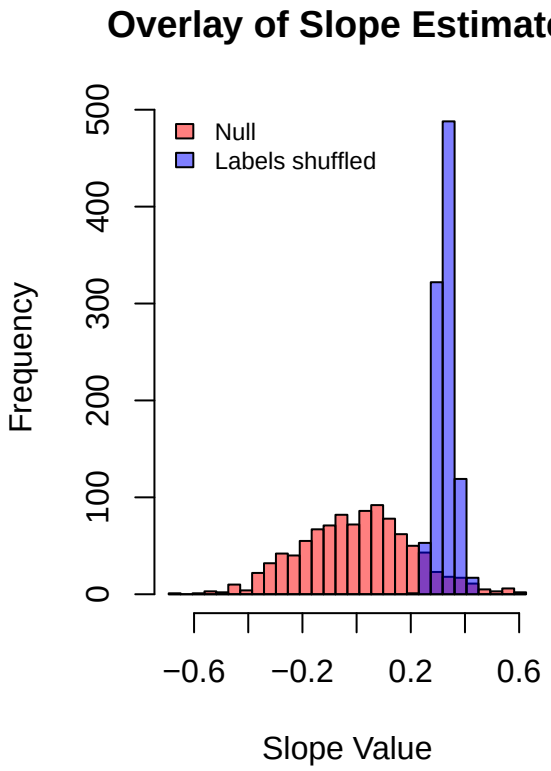
```
est <- lapply(all_fits,
              function(x) summary(x)$coefficients[2,c(1,4)])
est_df <- do.call(rbind,est)

par(mfrow=c(1,2))
hist(est_df[,1], main = "slope estimates", xlab = "slope value")
hist(est_df[,2], main = "p-values", xlab = "pvalue")
```

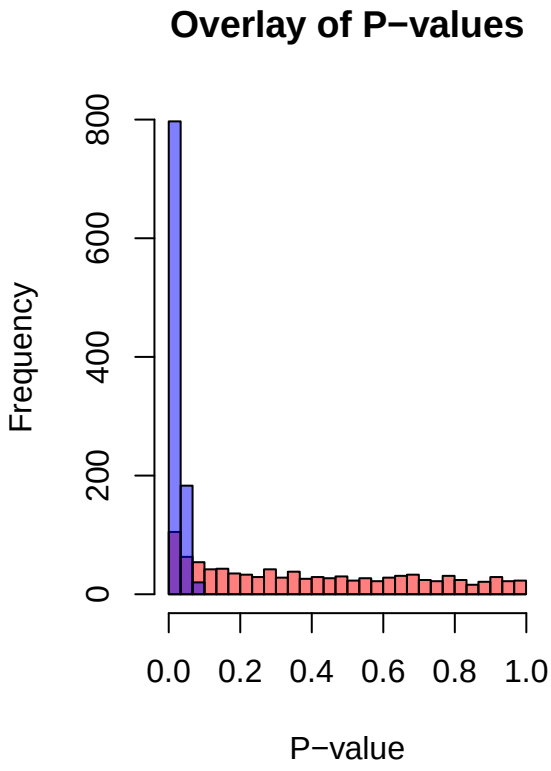




Comparing distributions

```r
# Calculate the combined range for the slope estimates
min_slope = min(c(shuffled_est_df[,1], est_df[,1]))
max_slope = max(c(shuffled_est_df[,1], est_df[,1]))

# Calculate the combined range for the p-values
min_pvalue = min(c(shuffled_est_df[,2], est_df[,2]))
max_pvalue = max(c(shuffled_est_df[,2], est_df[,2]))

# Define uniform break points for both datasets
breaks_slope = seq(min_slope, max_slope, length.out=31)  # 30 bins, adjust number of breaks as needed
breaks_pvalue = seq(min_pvalue, max_pvalue, length.out=31)  # 30 bins

par(mfrow=c(1,2))
# Plot histograms for slope estimates
hist(shuffled_est_df[,1], breaks=breaks_slope, main="Overlay of Slope Estimates", xlab="Slope Value", c
hist(est_df[,1], breaks=breaks_slope, add=TRUE, col=rgb(0, 0, 1, 0.5))
legend("topleft", legend=c("Null", "Labels shuffled"), fill=c(rgb(1, 0, 0, 0.5), rgb(0, 0, 1, 0.5)),bty

# Plot histograms for p-values
hist(shuffled_est_df[,2], breaks=breaks_pvalue, main="Overlay of P-values", xlab="P-value", col=rgb(1, 
hist(est_df[,2], breaks=breaks_pvalue, add=TRUE, col=rgb(0, 0, 1, 0.5))
```





This suggests that our results are robust to the label switching problem with average estimates shown below.

```
## [1] "mean null"
```

```
##    Estimate     p.value
## 0.001658102 0.402779611
```

```
## [1] "mean labels switched"
```

```
##   Estimate    p.value
## 0.32698784 0.02448534
```