Estimate contrast with known alpha

Cecile Ane and Qing (Sabrina) Yu

Feb 22,2017

Goal of the study

In this report, our goal is to check the distribution of phylogenetic contrasts with known alpha, known shift configuration and unknown shift values that are used in the bootstrap procedure in the `l1ou` package. In the last report, we used the true shift edge, true shift values, true alpha.

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.

Estimate shift configuration from original lizard data (change)

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). To simulate data according to a model that is biologically plausible, we estimated the shift configuration, shift values and covariance parameters from original lizard data.

This trait was analyzed to estimate the shifts in trait evolution using the function `fit_OU` from the `l1ou` package.

```
sessionInfo()$otherPkgs$l1ou$Version  #version 1.29
```

```
## [1] "1.29"
```

```
load("eModel_2_27.RData") # eModel.tree same as lizard.tree
data(lizard.tree, lizard.traits)
lizard <- adjust_data(lizard.tree, lizard.traits[,1])
```

```
## the new tree is normalized: each tip at distance 1 from the root.
## new Y: matrix of size 100 x 1
```

```
n_tips=length(lizard$tree$tip.label)  # 100: total number of tips
truealpha=eModel$alpha  # 0.606897967627925
y0=eModel$intercept
truetheta = y0 + l1ou:::convert_shifts2regions(eModel$tree,
                                eModel$shift.configuration, eModel$shift.values)
table(truetheta)
```

```
## truetheta
## -5.25186223573345 -4.56654712450638 -4.04766913950491 -3.45016309674712
##                 1                 3                 1                 1
## -3.23947780027265 -2.72293353502748 -2.36362306302032  -1.6605679645554
##                 1                 7                 1                25
## 0.248809571469948
##               158
```

```
# -5.25186223573345 -4.56654712450638 -4.04766913950491 -3.45016309674712 -3.23947780027265 -2.72293353
#                 1                 3                 1                 1                 1
```

```
#-2.36362306302032  -1.6605679645554 0.248809571469948
#                1                25               158
nShifts   = length(eModel$shift.configuration) # c(55,98,118,74,14,77,32,164)

sigma2=eModel$sigma2   # 0.0625186613743141
shift_config=eModel$shift.configuration
shiftnode=eModel$tree$edge[shift_config,1]# internal nodes with shift
```

Below, we use this model to simulate new data using the `rTraitCont` function from the `phylolm` package. `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. `covInverseSqrt` is the inverse square root of the phylogenetic covariance matrix, and `covSqrt` is the square-root of the phylogenetic covariance matrix. Finally, `contrast` 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.

simulation procedure

We use the same data for all scenarios: `set.seed` is used here to make it possible to re-simulate the exact same data, then these data are saved in a file (`Y_table`).

```
n_sim=100000
Y_table=matrix(nrow=n_sim, ncol=n_tips, data=NA)
set.seed(1293)
for (i in 1:n_sim) {
  Y  <- rTraitCont(eModel$tree, "OU", theta=truetheta,
                alpha=truealpha,
                sigma=sqrt(eModel$sigma2), root.value=y0)
  Y_table[i,]=Y
}
colnames(Y_table)=eModel$tree$tip.label
save(Y_table, file="Y_table_2_27.RData")
```

```
load("Y_table_2_27.RData")
len=dim(Y_table)[1]
n_sim=100000
```

Just to check our simulated values, we define our own function "ss" to calculate the variance of assuming a known population mean of 0. The name "ss" stands for "sum of squares", although this sum of squares is rescaled by the sample size.

```
ss=function(x){
      sum(x^2)/length(x)
    }
```

Now to check the mean and variance of the simulated Y values:

```
round(colMeans(Y_table-rep(eModel$mu[,1],each=n_sim),5)) # all 0: good
convar = apply(Y_table-rep(eModel$mu[,1],each=n_sim), 2, ss)
round(convar,3) # all about the same
mean(convar) # 0.03622244: same as expected variance below: good
sigma2/(2*truealpha) * (1-exp(-2*truealpha*1)) # 0.03620576
```

Contrast calculation procedure

We consider 4 scenarios for the calculation of contrasts, depending on what parameters are known (yes) or unknown (no) for the calculation of the inverse square-root of the covariance matrix.

parameter	1: full model known	2: alpha known	3: shift config known	4: tree known
$\beta 0$, $\beta 1$, ...	yes	no	no	no
α	yes	yes	no	no
shift config	yes	yes	yes	no

For the tree topology, we assume it known always. Refer to "filename" for the results of simulation 1, in which we know everything, and the contrasts behave as expected.

scenario 1: all parameters known and set to their true values

use Y table, call contrasts table "contrast_table1"

```
REf = sqrt_OU_covariance(lizard.tree, alpha=truealpha,
                         root.model = "OUfixedRoot", # eModel$l1ou.options$root.model,
                         check.order=F, check.ultrametric=F) # needed in l1ou v1.29

covInverseSqrt  = t(REf$sqrtInvSigma)
```

To check the correctness of this square-root covariance matrix calculation:

```
treeheight = branching.times(eModel$tree)[1] # 1.00000000002
Vbm = vcv(eModel$tree) # t_ij values. Tree height is 1 here, do d_ij = 2*(1-t_ij)
Vou = (1-exp(-2*truealpha*Vbm)) * exp(-truealpha*2*(treeheight-Vbm)) / (2*truealpha)
tmp = covInverseSqrt %*%  Vou %*% t(covInverseSqrt)
diag(tmp) # if normalize.tree.height=T was ommitted above, and l1ou v1.29:
# all 1.742289: ratio found later in convar/sigma2
all.equal(tmp, diag(n_tips), scale=1) # TRUE
```

Now that we can trust the inverse square-root of the covariance matrix, 158 we use it to calculate contrasts:

```
contrast_table=matrix(nrow=n_sim, ncol=n_tips, data=NA)
for (i in 1:len) {
  contrast_table[i,]=covInverseSqrt%*%(Y_table[i,] - eModel$mu)
}
save(contrast_table, file="contrast_table_2_27.RData")
```

scenario 2: beta unknown

Here, we assume that we know alpha, we know the shift configuration, but beta is unknown. 171 It took around 3 hours.

```
RE = sqrt_OU_covariance(lizard$tree, alpha=truealpha, # assumed known here
                         root.model =  eModel$l1ou.options$root.model,
                         check.order=F, check.ultrametric=F)
covInverseSqrt  <- t(RE$sqrtInvSigma)
```

```
contrast_table2=matrix(nrow=len, ncol=n_tips, data=NA)
sigma_table2=rep(NA,len)
mu_table2=matrix(nrow=len, ncol=n_tips, data=NA)
shift_values2=matrix(nrow=len, ncol=length(shift_config), data=NA)
for (i in 1:len) {
  model= fit_OU(lizard$tree, Y_table[i,], shift_config,
                alpha.upper=truealpha, alpha.lower=truealpha, alpha.starting.value=truealpha)
  # the line above is to fix alpha to the known value
 sigma_table2[i] = model$sigma2
 shift_values2[i,]=model$shift.values
 mu_table2[i,]=model$mu
  contrast_table2[i,]  <-  covInverseSqrt%*%(Y_table[i,] - model$mu)
}
save(sigma_table2,file="knownalpha_sigma2.RData")
 save(contrast_table2,file="knownalpha_contrast.RData")
 save(mu_table2,file="knownalpha_mu.RData")
 save(shift_values2,file="knownalpha_shiftvalues.RData")
```

scenario 3: alpha and beta unknown

Here we assume that the shift configuration is known, but not alpha and not beta. The correlation matrix among tips is unknown, then, so its inverse square-root needs to be re-calculated (based on the estimated alpha) for each simulated vector.

```
contrast_table3 = matrix(nrow=len, ncol=n_tips, data=NA)
alpha_table3  = rep(NA,len)
sigma_table3 = rep(NA,len)
mu_table3 = matrix(nrow=len, ncol=n_tips, data=NA)
shift_values3 = matrix(nrow=len, ncol=length(shift_config), data=NA)
# also save sigma2 estimated values, and 8 shift values? and/or 100 mu values?
for (i in 1:len) {
  model= fit_OU(lizard$tree, Y_table[i,], shift_config)
  alpha_table3[i] = model$alpha
  sigma_table3[i] = model$sigma2
  shift_values3[i,]=model$shift.values
  mu_table3[i,]=model$mu
  RE = sqrt_OU_covariance(lizard$tree, alpha=model$alpha, # alpha estimated here
                          root.model =  eModel$l1ou.options$root.model,
                         check.order=F, check.ultrametric=F)
  covInverseSqrt  <- t(RE$sqrtInvSigma)
  contrast_table3[i,] <-  covInverseSqrt%*%(Y_table[i,] - model$mu)
}
save(contrast_table3,file="unknownalpha_contrast.RData")
save(alpha_table3, file="unknownalpha_alpha.RData")
save(sigma_table3,file="unknownalpha_sigma2.RData")
save(mu_table3,file="unknownalpha_mu.RData")
save(shift_values3,file="unknownalpha_shiftvalues.RData")
```

scenario 4.1: unknown shift configuration

Also: unknown alpha and unknown beta. The estimation of the shift configuration takes a long time, so the number of simulations had to be reduced. In a preliminary run, the contrasts for 1262 simulations were calculated in 24 hours. So for the final run, we plan to complete a total of 2000 simulations ("n_sim" below). criterion: pBIC, which is the default of function `estimate_shift_configuration`.

```
n_sim=800 # lower because estimation is a lot slower when we have to search for the config
# use first n_sim rows of Ytable only
Y_table=Y_table[1:n_sim,]
contrast_table4=matrix(nrow=n_sim, ncol=n_tips, data=NA)
vectorOfShift <- vector(mode = "list", length = n_sim)
alpha_table4  = rep(NA,n_sim)
sigma_table4 = rep(NA,n_sim)
mu_table4 = matrix(nrow=n_sim, ncol=n_tips, data=NA)
shift_values4 = vector(mode = "list", length = n_sim)

for (i in 1:n_sim) {
  model= estimate_shift_configuration(lizard$tree, Y_table[i,])
  vectorOfShift[[i]] =model$shift.configuration
  alpha_table4[i] = model$alpha
  sigma_table4[i] = model$sigma2
  shift_values4[[i]]=model$shift.values
  mu_table4[i,]=model$mu
  RE = sqrt_OU_covariance(lizard$tree ,alpha=model$alpha,
                          root.model = "OUfixedRoot",
                          check.order=FALSE, check.ultrametric=FALSE)
  covInverseSqrt  <- t(RE$sqrtInvSigma)
  contrast_table4[i,] <-  covInverseSqrt%*%(Y_table[i,] - model$mu)
}
save(contrast_table4,file="unknownconfig_contrast_4.1.RData")
save(alpha_table4, file="unknownconfig_alpha_4.1.RData")
save(sigma_table4,file="unknownconfig_sigma2_4.1.RData")
save(mu_table4,file="unknownconfig_mu_4.1.RData")
save(shift_values4,file="unknownconfig_shiftvalues_4.1.RData")
save(vectorOfShift,file="unknownconfig_shifts_4.1.RData")
```

scenario 4.2: unknown shift configuration, unknown alpha and unknown beta

criterion: AICc

```
n_sim=800 # lower because estimation is a lot slower when we have to search for the config
# use first n_sim rows of Ytable only
Y_table=Y_table[1:n_sim,]
contrast_table5=matrix(nrow=n_sim, ncol=n_tips, data=NA)
vectorOfShift2 <- vector(mode = "list", length = n_sim)
alpha_table5  = rep(NA,n_sim)
sigma_table5 = rep(NA,n_sim)
mu_table5 = matrix(nrow=n_sim, ncol=n_tips, data=NA)
shift_values5 = vector(mode = "list", length = n_sim)

for (i in 1:n_sim) {
  model= estimate_shift_configuration(lizard$tree, Y_table[i,],criterion="AICc" )
  vectorOfShift2[[i]] =model$shift.configuration
```

```
  alpha_table5[i] = model$alpha
  sigma_table5[i] = model$sigma2
  shift_values5[[i]]=model$shift.values
  mu_table5[i,]=model$mu
  RE = sqrt_OU_covariance(lizard$tree ,alpha=model$alpha,
                          root.model = "OUfixedRoot",
                          check.order=FALSE, check.ultrametric=FALSE)
  covInverseSqrt  <- t(RE$sqrtInvSigma)
  contrast_table5[i,] <-  covInverseSqrt%*%(Y_table[i,] - model$mu)
}
save(contrast_table5,file="unknownconfig_contrast_4.2.RData")
save(alpha_table5, file="unknownconfig_alpha_4.2.RData")
save(sigma_table5,file="unknownconfig_sigma2_4.2.RData")
save(mu_table5,file="unknownconfig_mu_4.2.RData")
save(shift_values5,file="unknownconfig_shiftvalues_4.2.RData")
save(vectorOfShift2,file="unknownconfig_shifts_4.2.RData")
```

Results: visualizations of contrast distributions

```
#nodes that corresponding to small variance contrasts
intnode=lizard.tree$edge[,1]
all.equal(intnode, rep(199:101, each=2))
```

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

```
intnode=unique(intnode) ##remove repetitive numbers
```

The sequence of internal node is from 199:101, which corresponding to contrasts from 1: 99. The 100th contrast is the contrast of the root node. Here is a simple proof: Now to check the mean and variance of the simulated Y values:

```
# Now, we want to know which contrast does node 197 corresponding to. After we add 5 to all traits valu
truealpha=eModel$alpha
REf = sqrt_OU_covariance(lizard.tree, alpha=truealpha,
                         root.model =  eModel$l1ou.options$root.model,
                         check.order=F, check.ultrametric=F)

Dtf  = t(REf$sqrtInvSigma)
contrast_original=Dtf%*%(eModel$Y - eModel$mu)
eModel$Y[99]=eModel$Y[99]+5
eModel$Y[98]=eModel$Y[98]-5
eModel$Y[97]=eModel$Y[97]-5
eModel$Y[96]=eModel$Y[96]-5
contrast_change=Dtf%*%(eModel$Y - eModel$mu)
options(scipen=999)
cbind(contrast_original,contrast_change)
```

```
##                          [,1]          [,2]
##   [1,] -0.10947518720180664  -0.109475187
##   [2,] -0.20391621170386781  -0.203916212
##   [3,] -0.16836339207557977 -12.802737545
##   [4,] -0.17517647154780189  -0.175176472
##   [5,]  0.12732390795686993   0.127323908
```

```
##   [6,]  0.41444565695355912   0.414445657
##   [7,]  0.25622293765488147   0.256222938
##   [8,]  0.79739857446070006   2.871655463
##   [9,] -0.10215266190625116  -0.102152662
##  [10,] -0.07654323764936991  -0.076543238
##  [11,]  0.22266361486011427   0.222663615
##  [12,]  0.29620190282784437   0.296201903
##  [13,] -0.20014166745501877   0.469654355
##  [14,]  0.24630449148053085   0.246304491
##  [15,]  0.30678236248580520   0.906779820
##  [16,] -0.05228136002252097  -0.404287138
##  [17,] -0.01156318144703972  -0.011563181
##  [18,] -0.10881056113728954  -0.108810561
##  [19,] -0.82369060763912372  -0.823690608
##  [20,]  0.08883698279294662   0.088836983
##  [21,]  0.24087678680282099   0.240876787
##  [22,]  0.42436466955939178   0.424364670
##  [23,]  0.10123126465911000   0.101231265
##  [24,]  0.13214554643056095   0.132145546
##  [25,] -0.54857898601102362  -0.548578986
##  [26,] -0.21719368358609842  -0.217193684
##  [27,]  0.44261460470085257   0.442614605
##  [28,]  0.12302260183133965   0.123022602
##  [29,]  0.12774703852808544   0.127747039
##  [30,]  0.10991519921449291   0.109915199
##  [31,]  0.15269325398011388   0.152693254
##  [32,]  0.01559767597371309   0.015597676
##  [33,]  0.05878371655583753   0.058783717
##  [34,]  0.08219759885015171   0.082197599
##  [35,] -0.74069790336751351  -0.740697903
##  [36,] -0.33125218701662390  -0.331252187
##  [37,]  0.06906407801163141   0.069064078
##  [38,]  0.22486210735706463   0.224862107
##  [39,]  0.24455094796942473   0.244550948
##  [40,] -0.40893176298358963  -0.408931763
##  [41,] -0.39290292938698346  -0.392902929
##  [42,] -0.08295851926307238  -0.082958519
##  [43,]  0.06808987686994178   0.068089877
##  [44,]  0.05608603180076910   0.056086032
##  [45,] -0.14340772870844740  -0.143407729
##  [46,] -0.79950172014086185  -0.799501720
##  [47,] -0.12071242724422748  -0.120712427
##  [48,] -0.00377694651711848  -0.003776947
##  [49,]  0.01982999253016450   0.019829993
##  [50,] -0.02731047221956251  -0.027310472
##  [51,] -0.09786195803743299  -0.097861958
##  [52,]  0.11716437857299533   0.117164379
##  [53,]  0.03701845299343412   0.037018453
##  [54,]  0.26251621532315095   0.262516215
##  [55,] -0.00414884272676316  -0.004148843
##  [56,] -0.17781139288956221  -0.177811393
##  [57,]  0.07074554554806685   0.070745546
##  [58,]  0.09158298448682092   0.091582984
##  [59,] -0.00926447502481605  -0.009264475
```

```
##  [60,]  0.32801206751722367   0.328012068
##  [61,] -0.07538346472426145  -0.075383465
##  [62,]  0.01205077976859600   0.012050780
##  [63,] -0.14040997953331524  -0.140409980
##  [64,] -0.29119427606946491  -0.291194276
##  [65,]  0.02791556668065551   0.027915567
##  [66,]  0.21553645531217711   0.215536455
##  [67,] -0.19819860043871512  -0.198198600
##  [68,]  0.11504496220451804   0.115044962
##  [69,]  0.17175157534445609   0.171751575
##  [70,] -0.16834246964768601  -0.168342470
##  [71,]  0.10221808681558664   0.102218087
##  [72,]  0.36538847403961938   0.365388474
##  [73,]  0.29408134777332556   0.294081348
##  [74,]  0.04586877664551146   0.045868777
##  [75,]  0.05392494430993808   0.053924944
##  [76,]  0.38314657194311141   0.383146572
##  [77,] -0.26727759597524625  -0.267277596
##  [78,]  0.23701378566859410   0.237013786
##  [79,] -0.20418134660822992  -0.204181347
##  [80,]  0.05872596182633311   0.058725962
##  [81,] -0.00458795124548232  -0.004587951
##  [82,] -0.12782695529349680  -0.127826955
##  [83,] -0.37006249194232949  -0.370062492
##  [84,] -0.11476953145425853  -0.114769531
##  [85,] -0.05706542383790767  -0.057065424
##  [86,] -0.01897784543956124  -0.018977845
##  [87,]  0.04632514344698074   0.046325143
##  [88,]  0.08554927061323926   0.085549271
##  [89,]  0.23085318770631477   0.230853188
##  [90,] -0.16791845737715250  -0.167918457
##  [91,] -0.01151117341378606  -0.011511173
##  [92,]  0.07723932701150366   0.077239327
##  [93,]  0.09052346301305478   0.090523463
##  [94,] -0.02053727613353006  -0.020537276
##  [95,]  0.22770115095317778   0.227701151
##  [96,] -0.16649238471049044  -0.166492385
##  [97,]  0.13075389341821070   0.130753893
##  [98,] -0.25361707270714468  -0.253617073
##  [99,]  0.14801248520906202   0.704575135
## [100,] -0.00000000003619286  -0.325602613
```

Here, we can see the third contrast changes the most significantly, so the node 197 corresponding to contrast 3.

Results from Scenario 1: all parameters known and set to their true values

```
load("allknown_contrast.RData")
head(colMeans(contrast_table1))
```

```
## [1]  0.0002936259  0.0005437081 -0.0011580854 -0.0014622974  0.0013248416
## [6] -0.0001489877
```

```
max(abs(colMeans(contrast_table1)))# 0.0018224
```

```
## [1] 0.001822459
```

```
round(colMeans(contrast_table1),5)
```

```
##   [1]  0.00029  0.00054 -0.00116 -0.00146  0.00132 -0.00015 -0.00011
##   [8]  0.00032  0.00109 -0.00039 -0.00069 -0.00036 -0.00044  0.00150
##  [15]  0.00060  0.00015 -0.00048  0.00109 -0.00112  0.00032 -0.00021
##  [22]  0.00124  0.00055  0.00019 -0.00014  0.00166 -0.00067 -0.00171
##  [29] -0.00113  0.00035  0.00039 -0.00005  0.00079 -0.00073 -0.00058
##  [36]  0.00024  0.00016  0.00078  0.00136 -0.00034 -0.00106 -0.00081
##  [43]  0.00090  0.00036 -0.00182 -0.00073 -0.00113  0.00001  0.00079
##  [50] -0.00036  0.00092 -0.00088 -0.00041 -0.00033  0.00038 -0.00038
##  [57] -0.00080 -0.00008  0.00013 -0.00045  0.00070 -0.00048  0.00063
##  [64] -0.00121 -0.00094  0.00022 -0.00029 -0.00129 -0.00058  0.00018
##  [71]  0.00028 -0.00132 -0.00082  0.00006 -0.00080 -0.00120  0.00040
##  [78]  0.00006  0.00045  0.00045 -0.00039  0.00005  0.00058  0.00101
##  [85]  0.00033  0.00005  0.00086  0.00040 -0.00028  0.00032 -0.00049
##  [92]  0.00142 -0.00022 -0.00119  0.00075  0.00029 -0.00035 -0.00123
##  [99] -0.00040 -0.00026
```

```
convar=apply(contrast_table1,2,ss)
round(convar,3)
```

```
##   [1] 0.062 0.063 0.062 0.062 0.062 0.062 0.063 0.062 0.062 0.062 0.062
##  [12] 0.062 0.063 0.062 0.063 0.063 0.063 0.062 0.062 0.063 0.062 0.063
##  [23] 0.063 0.062 0.062 0.063 0.062 0.062 0.062 0.063 0.062 0.062 0.063
##  [34] 0.063 0.063 0.063 0.063 0.063 0.063 0.063 0.063 0.062 0.062 0.062
##  [45] 0.062 0.062 0.062 0.062 0.063 0.062 0.063 0.063 0.062 0.062 0.063
##  [56] 0.062 0.062 0.062 0.062 0.063 0.063 0.063 0.062 0.063 0.063 0.063
##  [67] 0.062 0.063 0.062 0.063 0.063 0.063 0.063 0.062 0.062 0.062 0.063
##  [78] 0.062 0.063 0.063 0.062 0.063 0.063 0.063 0.063 0.063 0.062 0.062
##  [89] 0.063 0.062 0.062 0.063 0.063 0.062 0.062 0.063 0.062 0.063 0.062
## [100] 0.062
```

```
mean(convar)# 0.06249321
```

```
## [1] 0.06249321
```

```
sigma2# 0.06251866
```

```
## [1] 0.06251866
```

```
0.1089751/0.06251866 # 1.743081. l1ou v1.29 without normalize.tree.height: mean(convar) was 0.1089751
```

```
## [1] 1.743081
```

Mean of contrasts are close to zero and the variance of contrasts is close to `sigma2`.

Results from Scenario 2: ## scenario 2: beta unknown

but known alpha and known shift configuration

```
load("knownalpha_contrast.RData")
load("knownalpha_sigma2.RData")
load("knownalpha_shiftvalues.RData")
load("knownalpha_mu.RData")
```

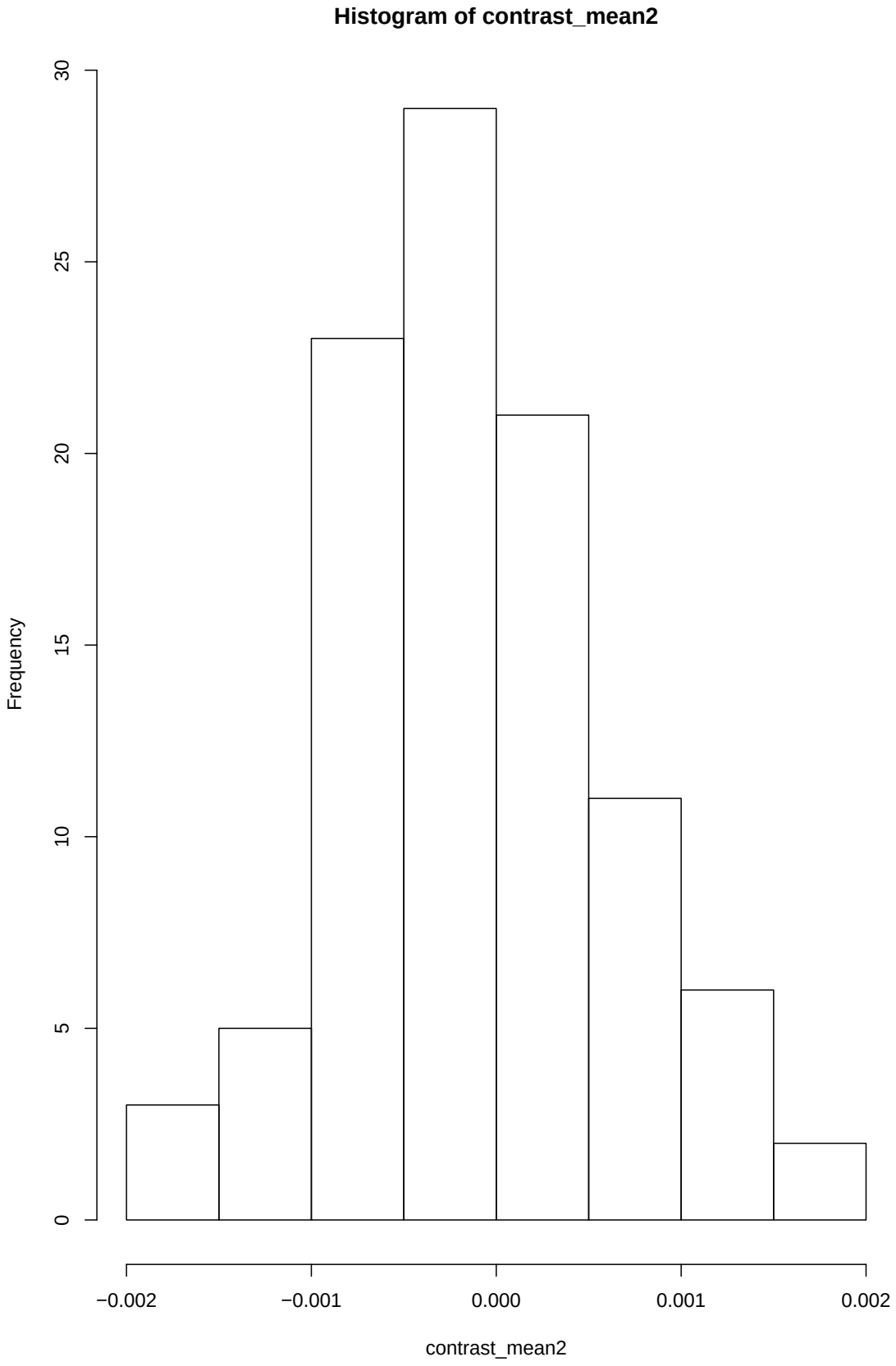
```r
# contrasts mean
contrast_mean2=colMeans(contrast_table2)
head(contrast_mean2)
```

```
## [1] -0.0008017153 -0.0010609915 -0.0008243967 -0.0009007101 -0.0001081647
## [6]  0.0006750217
```

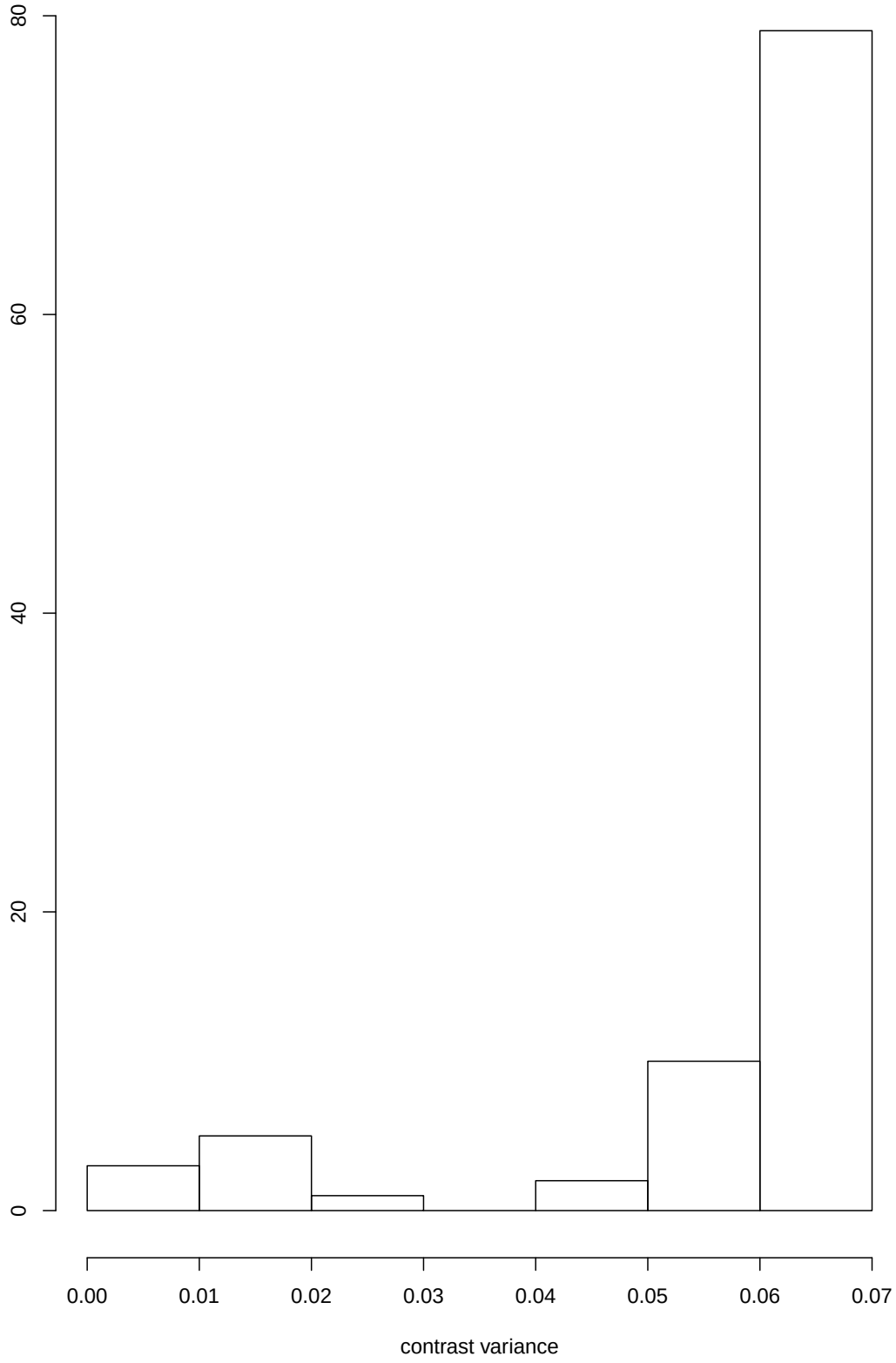
```r
max(abs(contrast_mean2))
```

```
## [1] 0.001775369
```

```r
hist(contrast_mean2)
```

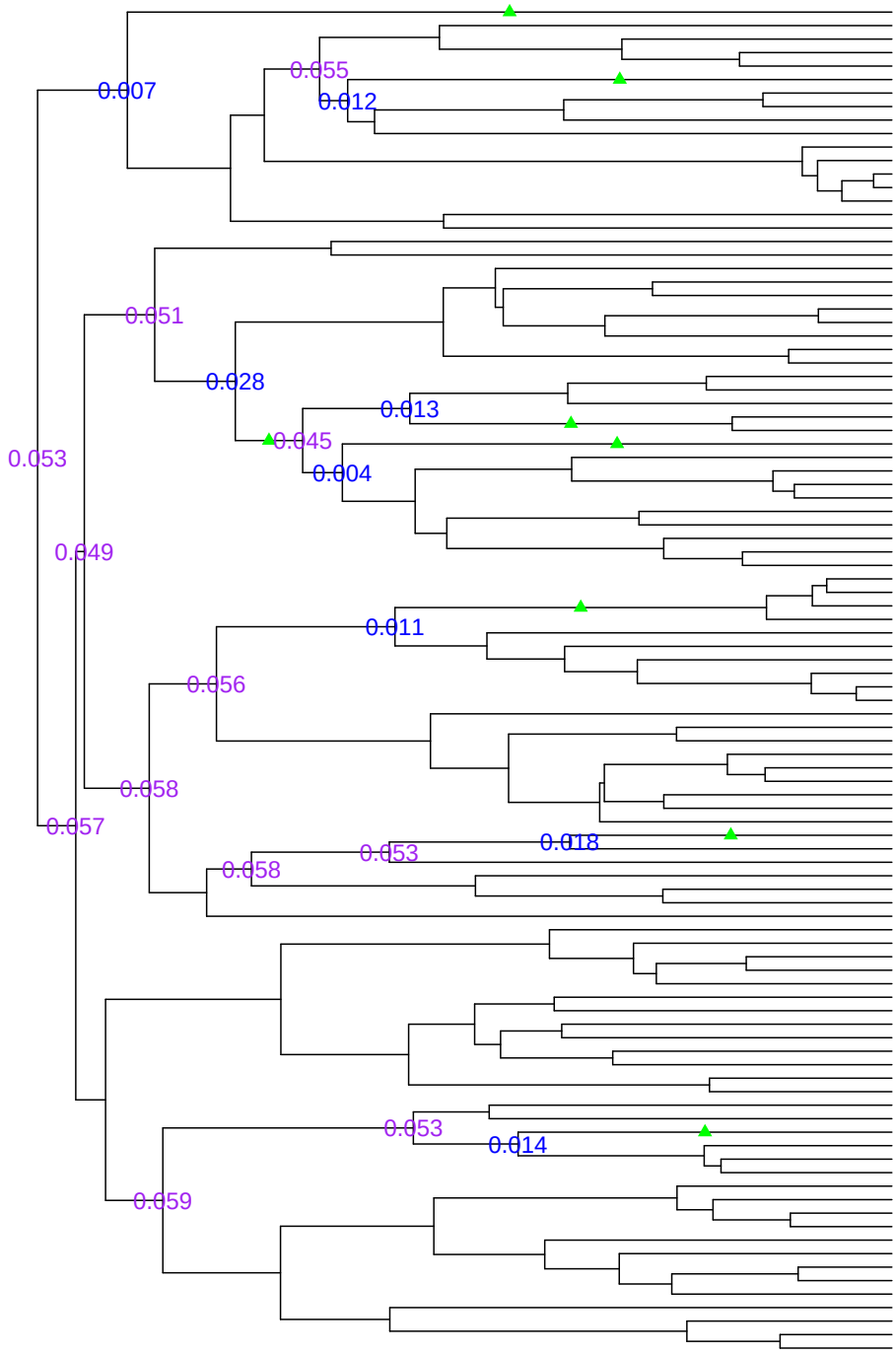

Histogram of contrast_mean2
Frequency
0
5
10
15
20
25
30
−0.002
−0.001
0.000
0.001
0.002
contrast_mean2

```r
# contrasts variance
convar2=apply(contrast_table2,2,ss)
contrast2=colMeans(contrast_table2)
hist(convar2,main="variance of contrasts in Scenario 2",xlab="contrast variance",ylab="")
```


variance of contrasts in Scenario 2
0
20
40
60
80
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
contrast variance

```r
#Try to find the contrast with variance relatively small, median and large.
ind_small2=which(convar2<0.04)
ind_median2=which(convar2<0.06&convar2>=0.04)
median_node2=intnode[ind_median2]
small_node2=intnode[ind_small2]
plot(eModel$tree,show.tip.label = F,main="",font=8)
nodelabels(round(convar2[ind_small2],3),small_node2,col="blue",frame = "none")
edgelabels(edge=shift_config,col="green",frame = "none",pch=17)
nodelabels(round(convar2[ind_median2],3),median_node2,col="purple",frame = "none")
mtext("blue: contrast variance < 0.04, purple: contrast variance > 0.04 and < 0.06,
      green dots: shifts",
      line=1,side=3,las=0)
```

blue: contrast variance < 0.04, purple: contrast variance > 0.04 and < 0.06,
green dots: shifts



```
round(convar2,3)
```

```
##   [1] 0.062 0.062 0.062 0.063 0.063 0.063 0.012 0.055 0.063 0.063 0.062
##  [12] 0.063 0.061 0.063 0.061 0.007 0.063 0.062 0.062 0.062 0.062 0.063
##  [23] 0.062 0.062 0.063 0.063 0.062 0.013 0.063 0.063 0.063 0.063 0.062
##  [34] 0.063 0.062 0.063 0.004 0.045 0.028 0.051 0.063 0.062 0.063 0.063
##  [45] 0.062 0.063 0.063 0.062 0.011 0.062 0.062 0.062 0.062 0.062 0.062
##  [56] 0.063 0.063 0.056 0.018 0.053 0.063 0.063 0.058 0.061 0.058 0.049
##  [67] 0.063 0.062 0.063 0.063 0.063 0.062 0.062 0.062 0.063 0.063 0.063
##  [78] 0.063 0.063 0.063 0.063 0.014 0.053 0.063 0.063 0.062 0.063 0.063
##  [89] 0.063 0.063 0.062 0.063 0.063 0.063 0.062 0.059 0.062 0.057 0.053
## [100] 0.000
```

```
mean(convar2)
```

```
## [1] 0.0569135
```

```
mean(convar2[-ind_small2])
```

```
## [1] 0.06136439
```

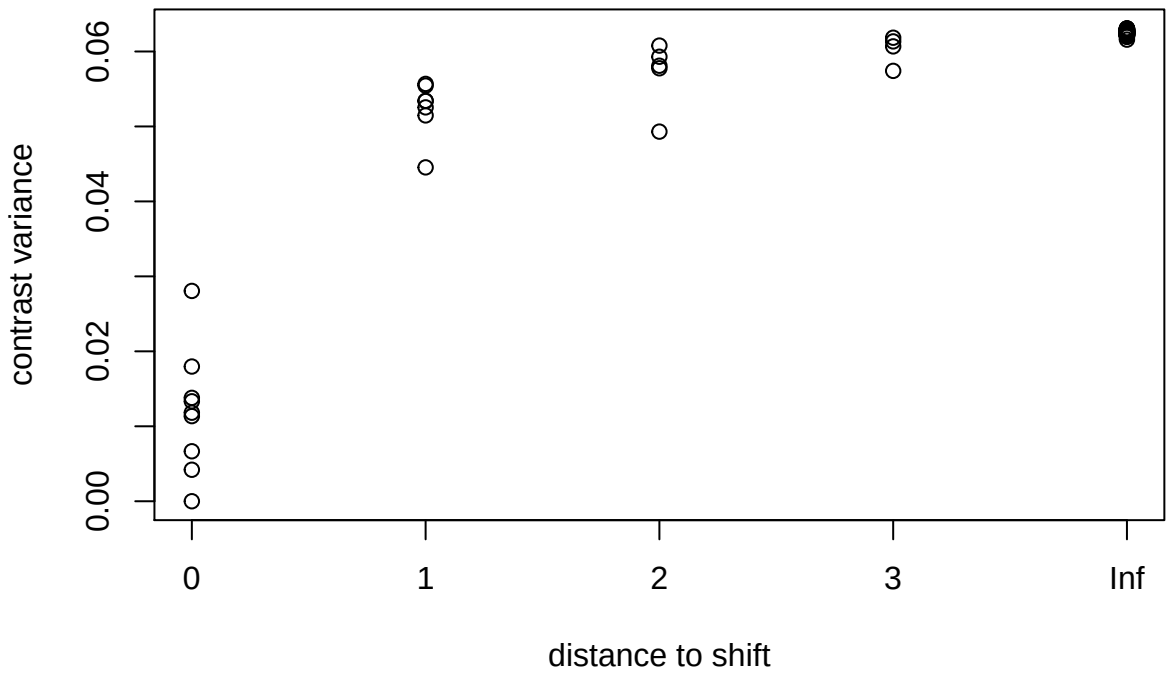
```
sigma2
```

```
## [1] 0.06251866
```

Some contrasts with extremely small variance draw our attention. When we make a plot, we can see those contrasts with small variances appear where the shift exists. Those shifts make the variances of contrasts underestimated. And the plot below can clearly show as nodes become further away from shifts, their variances of contrast increase. The nodes closer to the root have higher variance of contrasts. And it is clear that the last contrast has variance equal to 0 since we have a fixed root.

```
###making plots for contrasts
dist2shift=rep(Inf,199)
dist2shift[shiftnode]=0
shiftparent=shiftnode
for(k in 1:10){
  old=c()
  for(node in shiftparent){
    ind=which(node==eModel$tree$edge[,2])
    old=c(old,ind)
  }
  shiftparent=eModel$tree$edge[old,1]
  for(node in shiftparent){
    if(dist2shift[node]>k) dist2shift[node]=k
  }
}

dist2shift=dist2shift[-c(1:99)]
dist2shift=rev(dist2shift)
dist2shift[100]=0 ##The last contrast is the root edge
dist2shift[dist2shift==Inf] = 4
plot(convar2~dist2shift, xaxt="n",xlab="distance to shift", ylab="contrast variance")
axis(1, at=0:4, labels=as.character(c(0:3,Inf)))
```



Results from Scenario 3: alpha and beta unknown

but known shift configuration

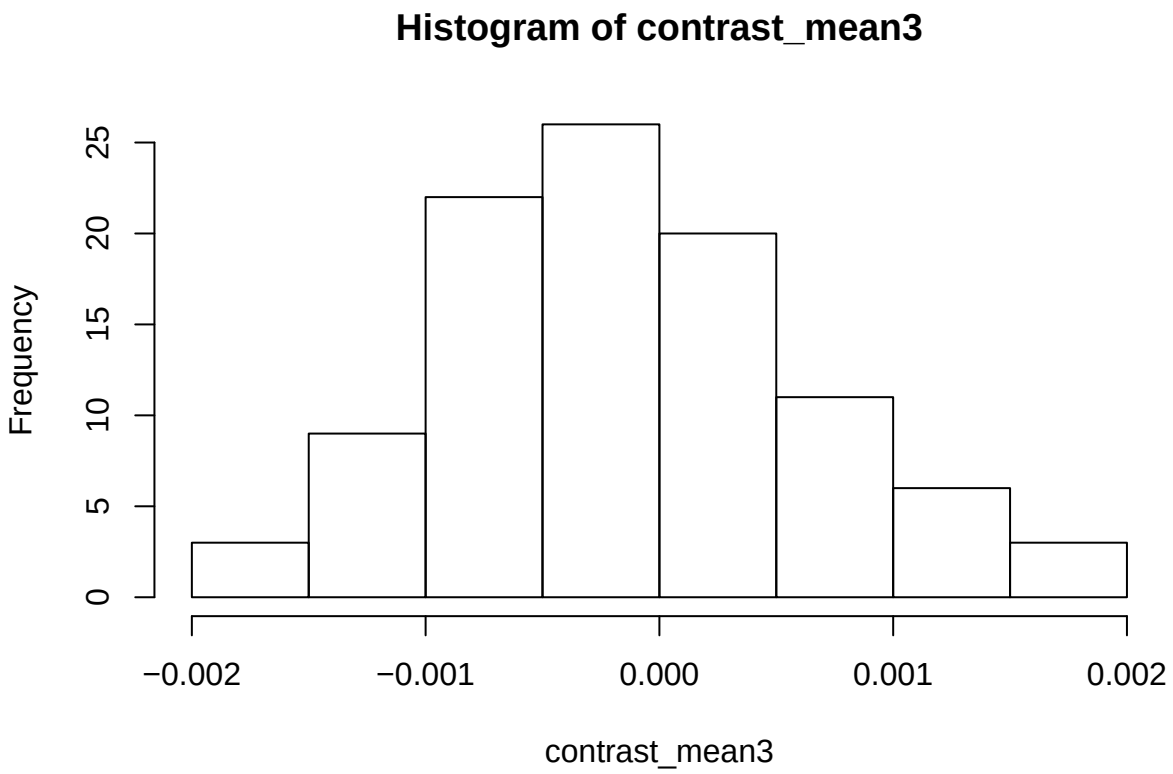
```
load("unknownalpha_contrast.RData")
load("unknownalpha_sigma2.RData")
load("unknownalpha_shiftvalues.RData")
load("unknownalpha_mu.RData")
load("unknownalpha_alpha.RData")
# calculate contrasts mean
contrast_mean3=colMeans(contrast_table3)
head(contrast_mean3)
```

```
## [1] -0.0008208711 -0.0010534489 -0.0008314135 -0.0009439863 -0.0001392928
## [6]  0.0007259835
```

```
max(abs(contrast_mean3))
```

```
## [1] 0.001924415
```

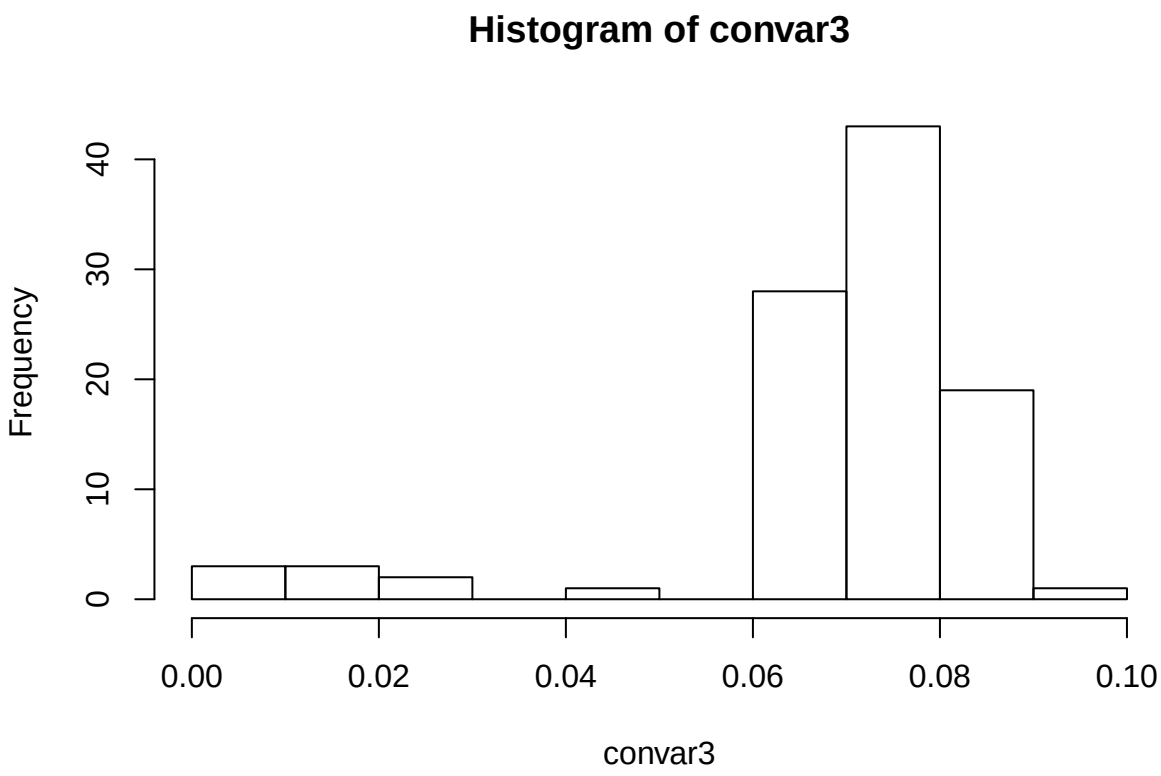
```
hist(contrast_mean3)
```




```
# Calculate contrasts variance
convar3=apply(contrast_table3,2,ss)
round(convar3,3)
```

```
##   [1] 0.069 0.074 0.080 0.068 0.076 0.081 0.017 0.076 0.064 0.065 0.066
##  [12] 0.067 0.083 0.078 0.084 0.009 0.081 0.071 0.065 0.074 0.079 0.077
##  [23] 0.067 0.081 0.070 0.076 0.069 0.020 0.067 0.068 0.076 0.073 0.068
##  [34] 0.072 0.082 0.084 0.005 0.062 0.045 0.074 0.066 0.066 0.069 0.064
##  [45] 0.066 0.073 0.076 0.078 0.019 0.071 0.067 0.069 0.071 0.075 0.074
##  [56] 0.080 0.080 0.080 0.027 0.067 0.072 0.079 0.078 0.082 0.083 0.068
##  [67] 0.069 0.072 0.073 0.076 0.075 0.074 0.072 0.078 0.080 0.070 0.083
##  [78] 0.089 0.077 0.070 0.070 0.020 0.070 0.067 0.070 0.071 0.067 0.072
##  [89] 0.074 0.077 0.083 0.067 0.071 0.081 0.087 0.085 0.092 0.084 0.080
## [100] 0.000
```

```
hist(convar3)
```




```
###Exactly the same as last one
intnode[convar3<0.06]
```

```
## [1] 193 184 172 163 161 151 141 118  NA
```

```
mean(convar3)
```

```
## [1] 0.06909039
```

```
sigma2
```

```
## [1] 0.06251866
```

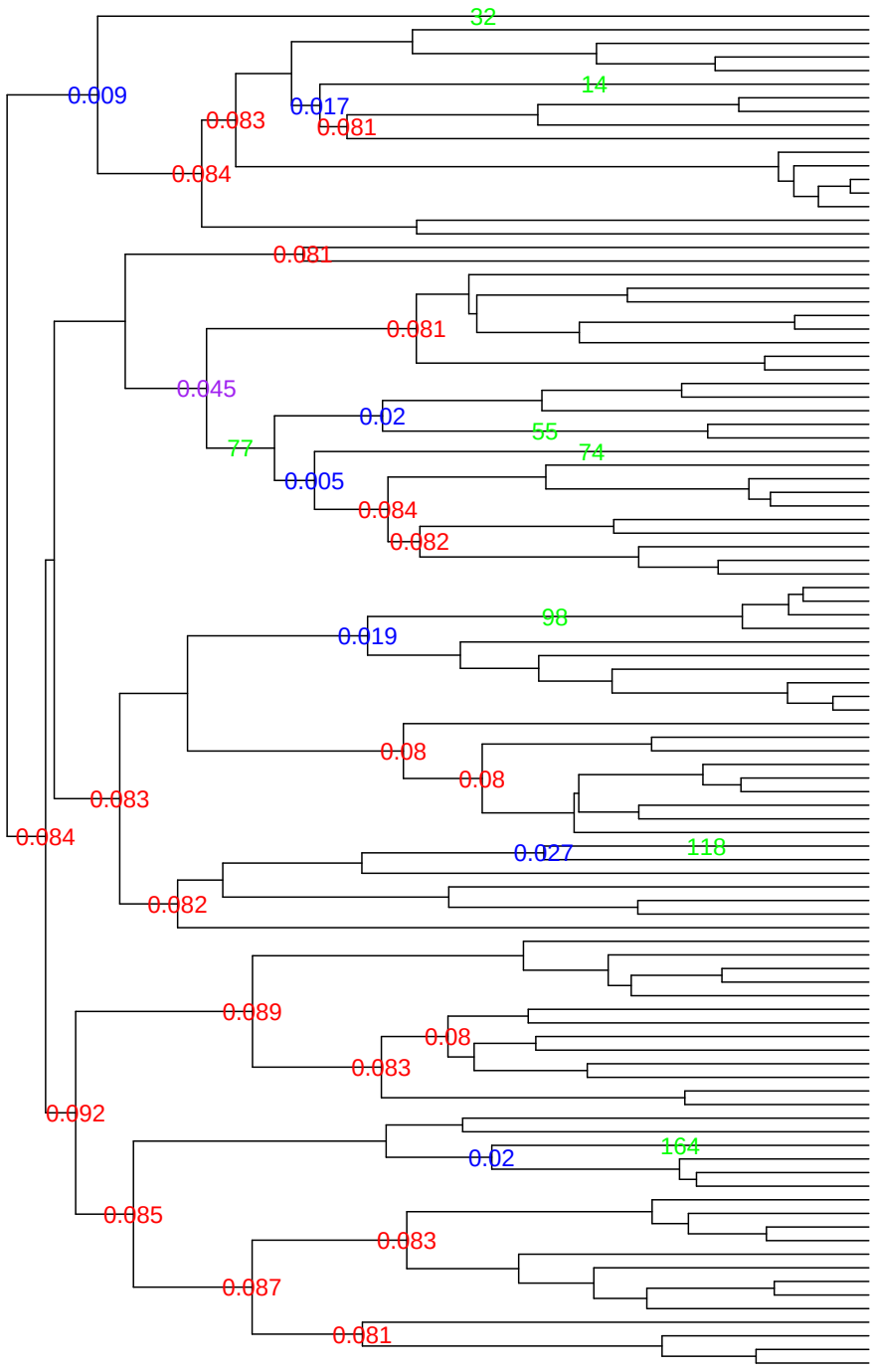
```
#Try to find the contrast with variance relatively small, median and large.
ind_small3=which(convar3<0.04)
ind_median3=which(convar3<0.06&convar3>=0.04)
ind_large3=which(convar3>=0.08)
median_node3=intnode[ind_median3]
small_node3=intnode[ind_small3]
large_node3=intnode[ind_large3]
large_node3
```

```
##  [1] 194 187 185 183 176 165 164 144 143 136 135 125 123 122 109 106 105
## [18] 104 103 102
```

```
mean(convar3[-c(ind_small3,ind_large3)])
```

```
## [1] 0.07117836
```

```
plot(eModel$tree,show.tip.label = F,main="small,median,large contrast nodes",font=8)
nodelabels(round(convar3[ind_small3],3),small_node3,col="blue",frame = "none")
edgelabels(shift_config,shift_config,col="green",frame = "none")
nodelabels(round(convar3[ind_median3],3),median_node3,col="purple",frame = "none")
nodelabels(round(convar3[ind_large3],3),large_node3,col="red",frame = "none")
```


small,median,large contrast nodes
32
14
0.009
0.017
0.083
0.081
0.084
0.081
0.081
0.045
0.02
55
77
74
0.005
0.084
0.082
98
0.019
0.08
0.08
0.083
0.084
0.027
118
0.082
0.089
0.08
0.083
0.092
0.02
164
0.085
0.083
0.087
0.081

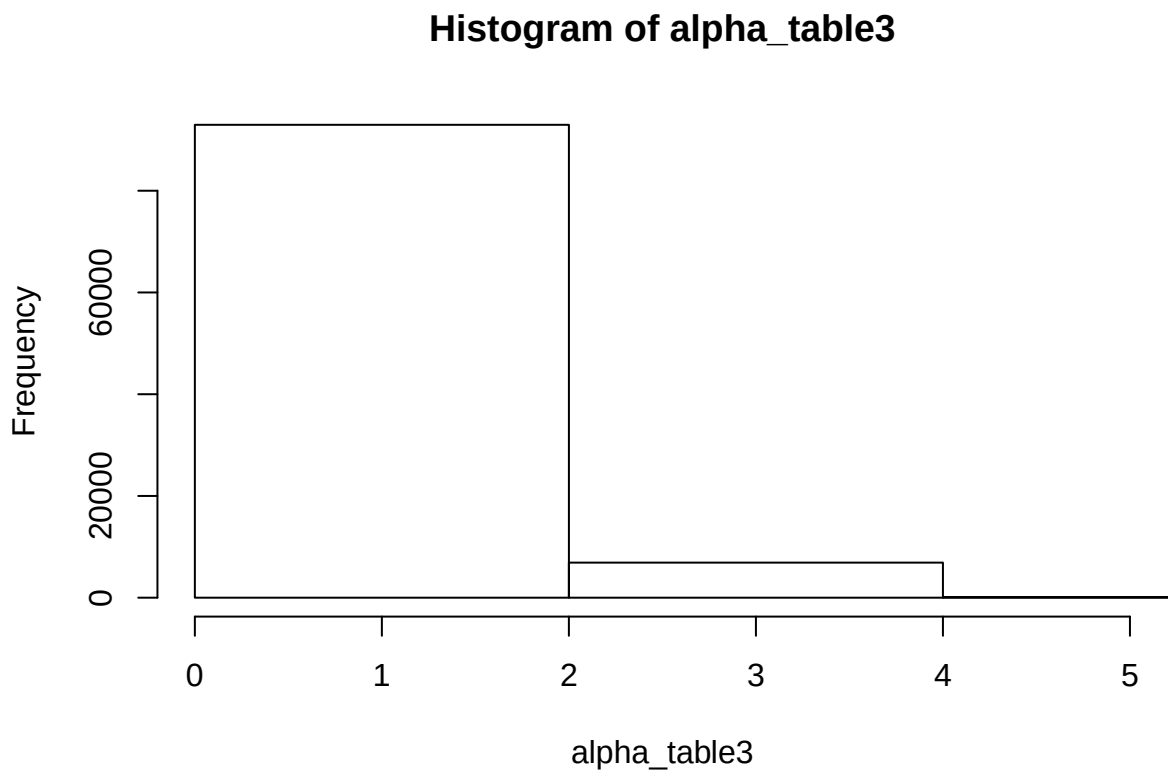
Since al-

pha is unknown, the variance of contrasts being estimated would be different with varied alpha values. So, I split contrast tables into two different sets of contrasts based on either alpha has been underestimated or overestimated.

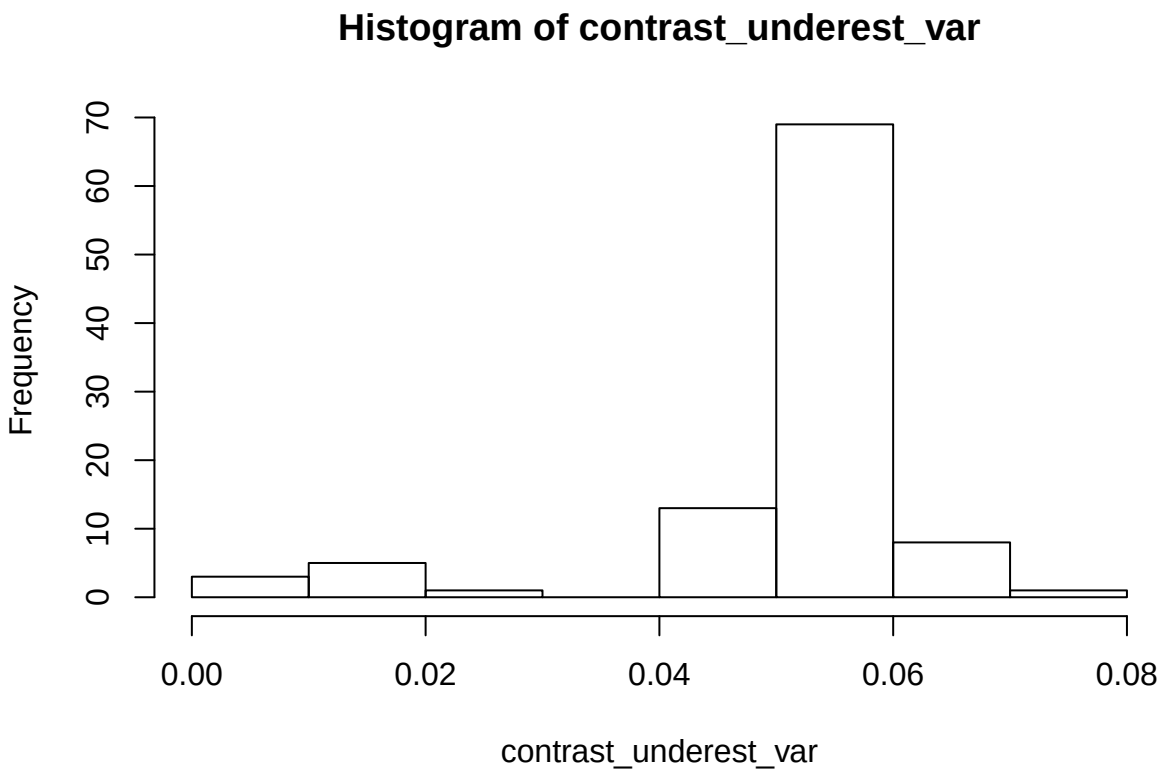
```
#explore alpha
n_sim=100000
mean(alpha_table3)
```

```
## [1] 1.065242
```

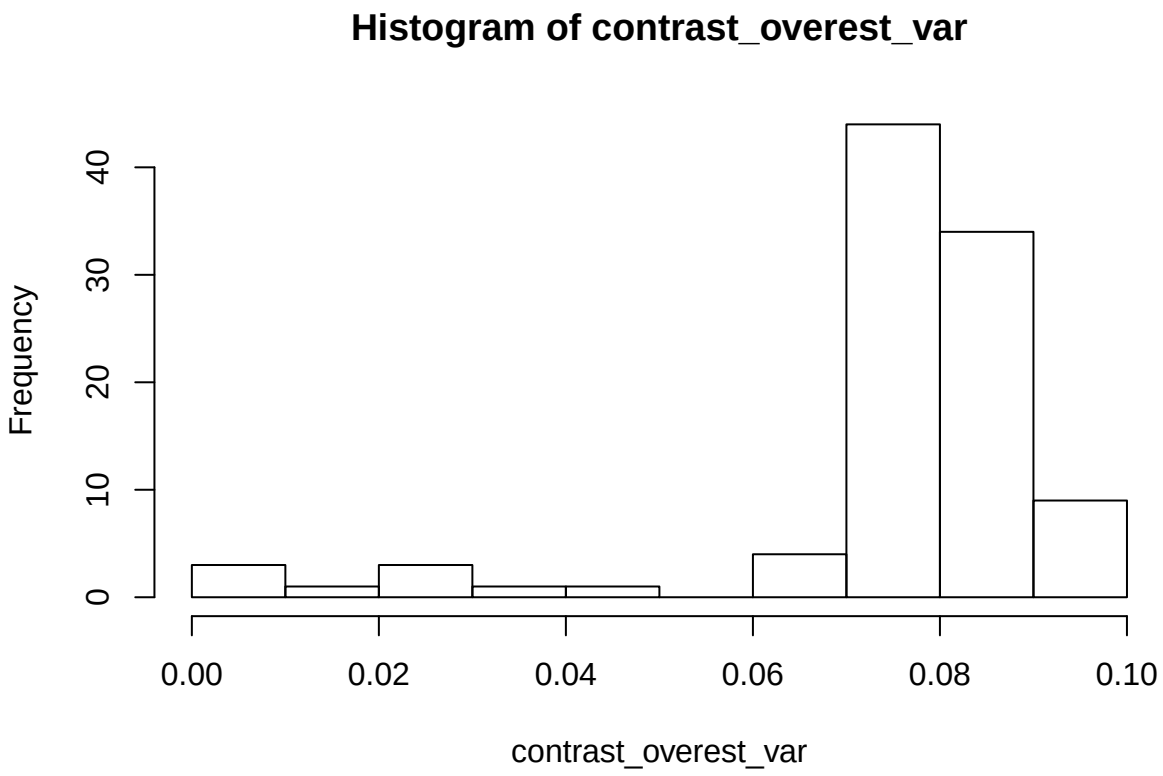
```
hist(alpha_table3,xlim=c(0,5))
```




```
alpha_underest=rep(2,n_sim)
alpha_underest[alpha_table3<eModel$alpha]=1
contrast_underest=subset(contrast_table3,alpha_underest==1)
contrast_overest=subset(contrast_table3,alpha_underest==2)
contrast_underest_var=apply(contrast_underest,2,var)
contrast_overest_var=apply(contrast_overest,2,var)
hist(contrast_underest_var)
```




```
hist(contrast_overest_var)
```

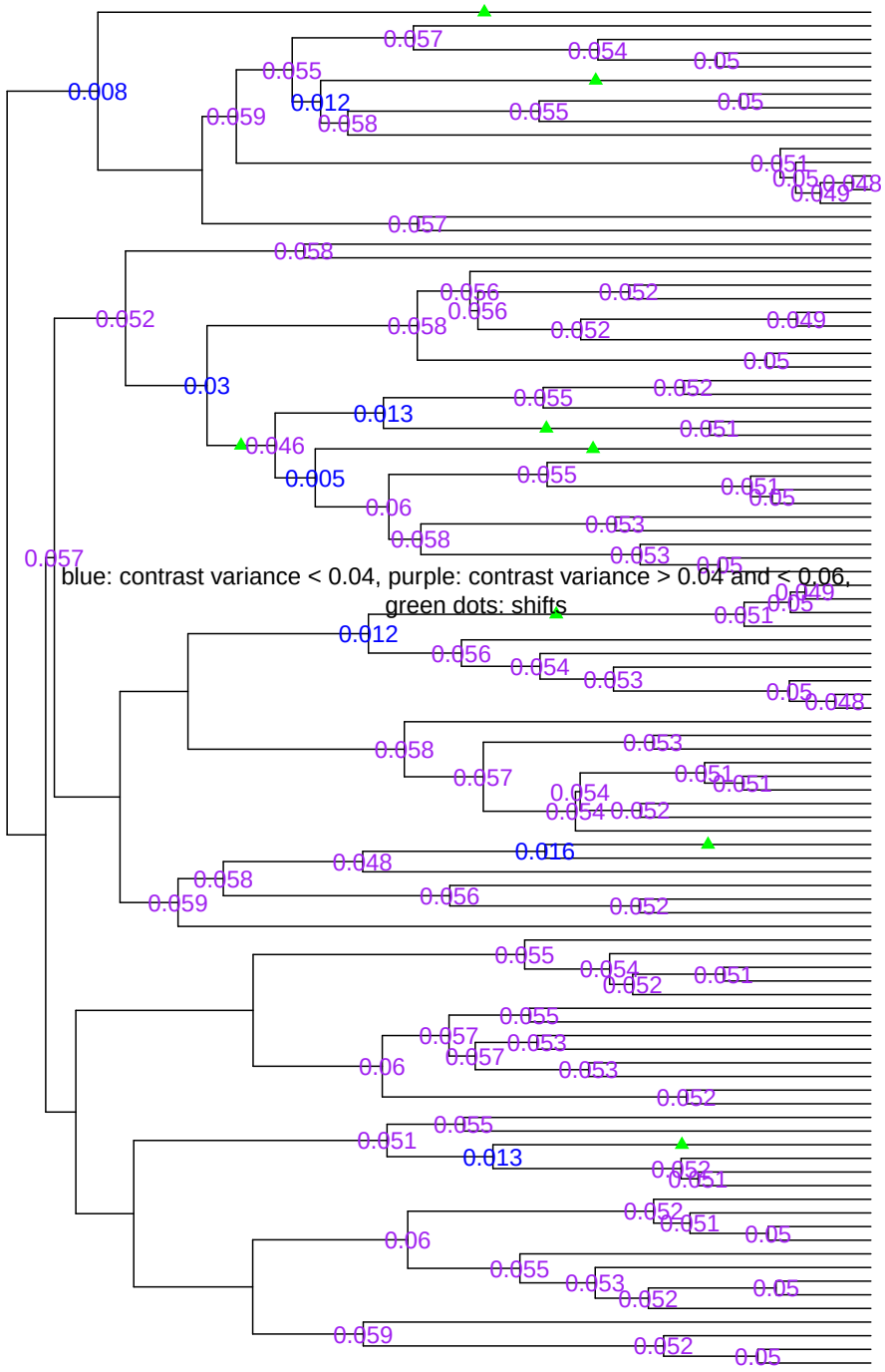



```
#Try to find the contrast with variance relatively small and large with underestimated alpha
ind_small_under=which(contrast_underest_var<0.04)
ind_median_under=which(contrast_underest_var<0.06&contrast_underest_var>=0.04)
ind_large_under=which(contrast_underest_var>=0.08)
median_node3_under=intnode[ind_median_under]
small_node3_under=intnode[ind_small_under]
large_node3_under=intnode[ind_large_under]
large_node3_under
```

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

Try to find the contrast‘s variance with underestimated alpha

```
plot(eModel$tree,show.tip.label = F,main="estimated alpha < true alpha",font=8)
nodelabels(round(contrast_underest_var[ind_small_under],3),small_node3_under,col="blue",frame = "none")
edgelabels(edge=shift_config,col="green",frame = "none",pch=17)
nodelabels(round(contrast_underest_var[ind_median_under],3),median_node3_under,col="purple",frame = "no
##nodelabels(round(contrast_underest_var[ind_large_under],3),large_node3_under,col="red",frame = "none"
mtext("blue: contrast variance < 0.04, purple: contrast variance > 0.04 and < 0.06,
      green dots: shifts",
      line=-23,side=3,las=0)
```


estimated alpha < true alpha
blue: contrast variance < 0.04, purple: contrast variance > 0.04 and < 0.06,
green dots: shifts

Many

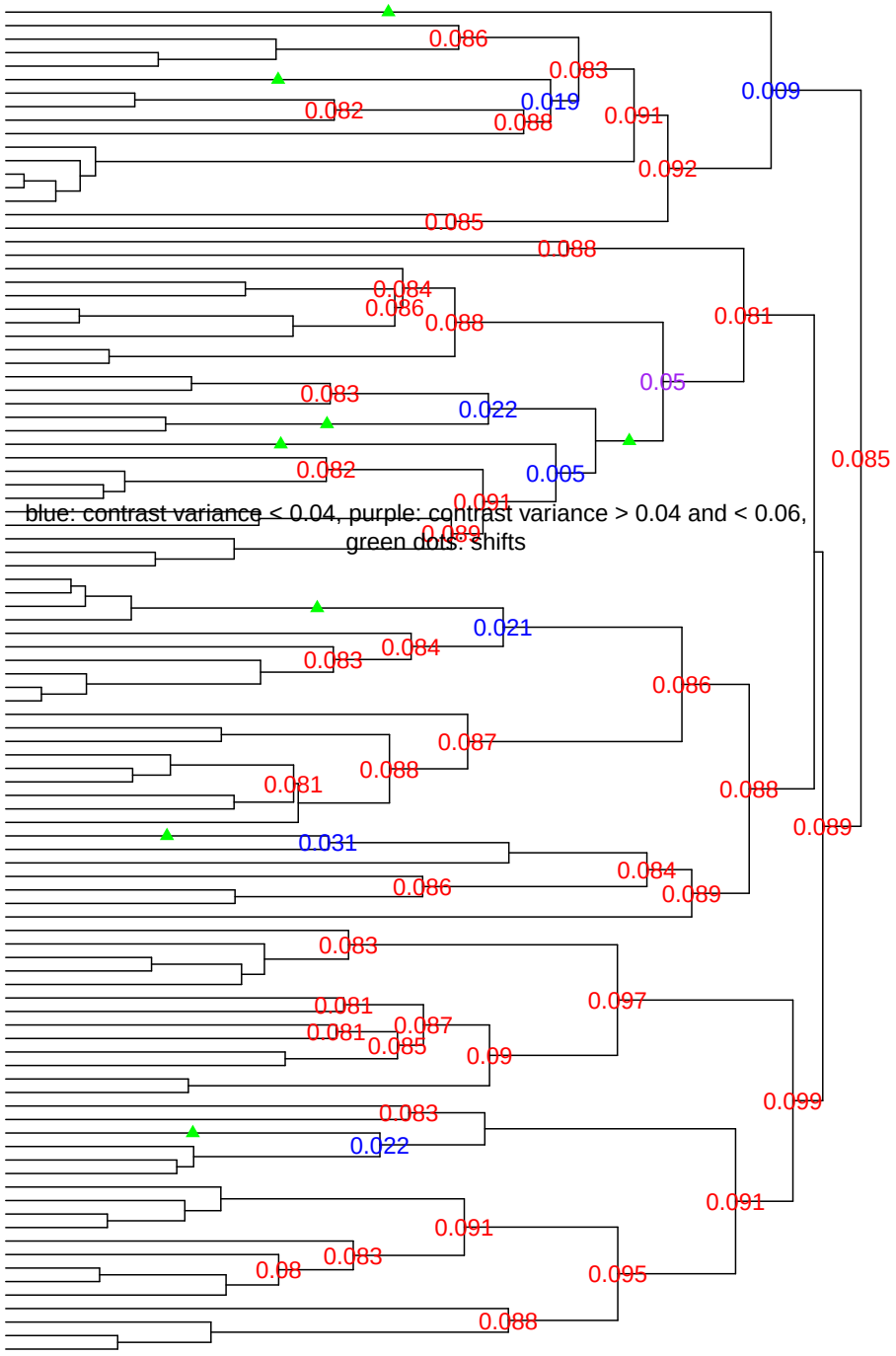
contrasts with at least one child node as tip of the tree are underestimated. As mentioned above, contrasts of nodes with a shift edge are underestimated. One of the node labled in red has relatively large variance of contrast. This situation cannot be fully explained by its location closer to the root. (need some investigations)

Try to find the contrast's variance with overestimated alpha

```
ind_small_over=which(contrast_overest_var<0.04)
ind_median_over=which(contrast_overest_var<0.06&contrast_overest_var>=0.04)
ind_large_over=which(contrast_overest_var>=0.08)
median_node3_over=intnode[ind_median_over]
small_node3_over=intnode[ind_small_over]
large_node3_over=intnode[ind_large_over]
large_node3_over
```

```
##  [1] 197 195 194 192 187 186 185 183 179 178 176 174 169 165 164 160 153
## [18] 152 146 144 143 142 138 137 136 135 130 129 128 126 125 123 122 121
## [35] 111 110 109 106 105 104 103 102 101
```

```
plot(eModel$tree,show.tip.label = F,main="estimated alpha > true alpha",font=8,direction = "leftwards")
nodelabels(round(contrast_overest_var[ind_small_over],3),small_node3_over,col="blue",frame = "none")
edgelabels(edge=shift_config,col="green",frame = "none",pch=17)
nodelabels(round(contrast_overest_var[ind_median_over],3),median_node3_over,col="purple",frame = "none")
nodelabels(round(contrast_overest_var[ind_large_over],3),large_node3_over,col="red",frame = "none")
mtext("blue: contrast variance < 0.04, purple: contrast variance > 0.04 and < 0.06,
      green dots: shifts",
      line=-21,side=3,las=0)
```


estimated alpha > true alpha
0.086
0.083
0.019
0.009
0.082
0.088
0.091
0.092
0.085
0.088
0.084
0.086
0.088
0.081
0.05
0.083
0.022
0.085
0.082
0.005
0.091
blue: contrast variance < 0.04, purple: contrast variance > 0.04 and < 0.06,
green dots: shifts
0.089
0.021
0.084
0.083
0.086
0.087
0.088
0.081
0.088
0.089
0.031
0.084
0.086
0.089
0.083
0.097
0.081
0.087
0.081
0.085
0.09
0.099
0.083
0.022
0.091
0.091
0.083
0.08
0.095
0.088

When

alpha has been overestimated, there are much more nodes with large variance of contrasts. And most of those nodes are closer to root.

Results from unknown shift configuration

and also unknown alpha and unknown beta. Spent 24hrs, generated 1262 eModels

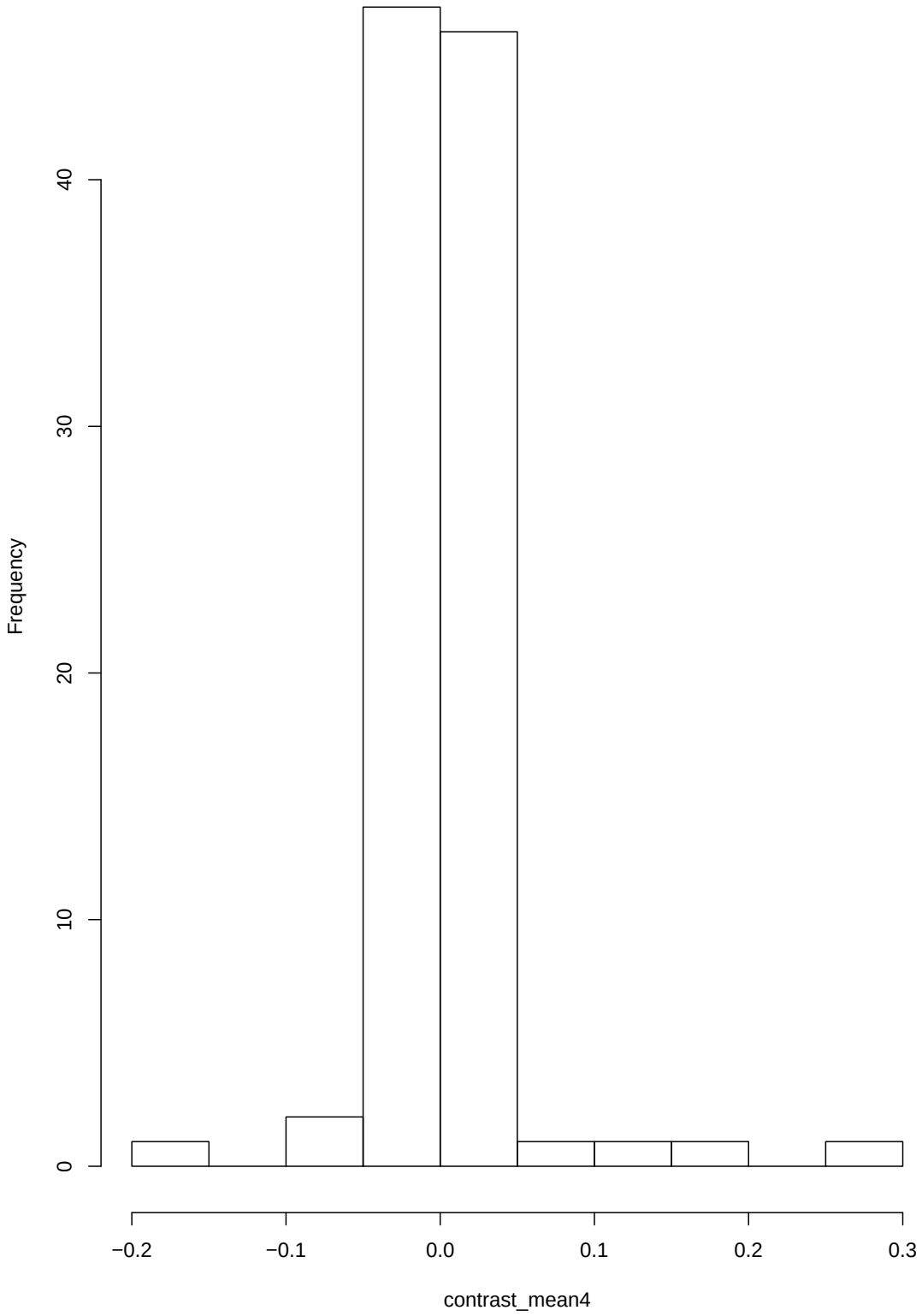
```
load("unknownconfig_alpha_4.1.RData")
load("unknownconfig_shifts_4.1.RData")
load("unknownconfig_sigma2_4.1.RData")
load("unknownconfig_mu_4.1.RData")
load("unknownconfig_shiftvalues_4.1.RData")
load("unknownconfig_contrast_4.1.RData")
#Calculate mean of contrasts
contrast_mean4=colMeans(contrast_table4)
head(contrast_mean4)
```

```
## [1] -0.0107097344 -0.0004473309 -0.0027052034 -0.0142488107  0.0026654166
## [6] -0.0041520054
```

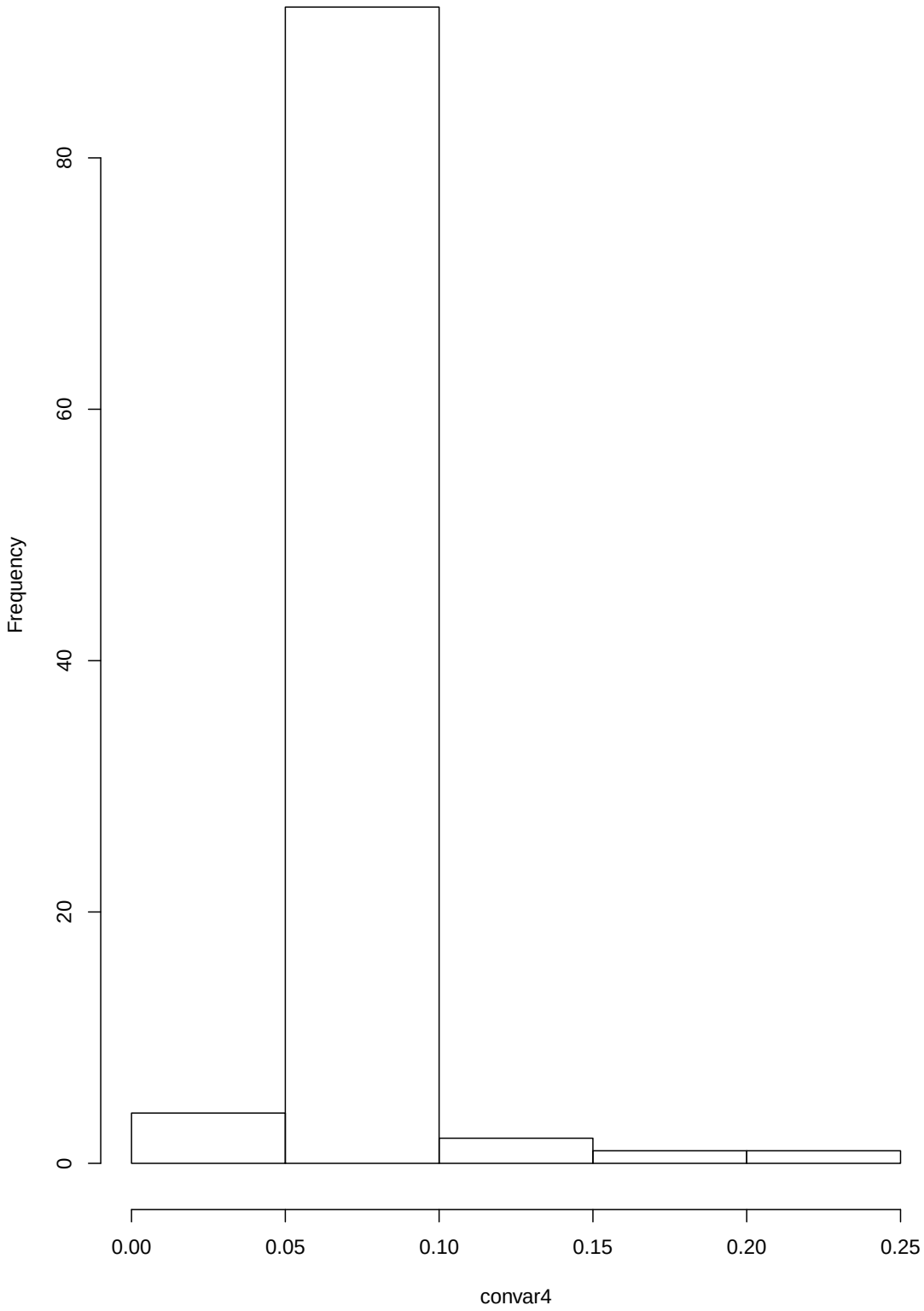
```
max(abs(contrast_mean4))
```

```
## [1] 0.2764788
```

```
hist(contrast_mean4)
```


Histogram of contrast_mean4
Frequency
0
10
20
30
40
−0.2
−0.1
0.0
0.1
0.2
0.3
contrast_mean4

```r
#Calculate variance of contrasts
convar4=apply(contrast_table4,2,ss)
hist(convar4)
```


Histogram of convar4
Frequency
0
20
40
60
80
0.00
0.05
0.10
0.15
0.20
0.25
convar4

```
#Shifts that occur the most often
vectorOfShift_1=unlist(vectorOfShift, recursive = TRUE, use.names = F)
freq_table=as.data.frame(table(vectorOfShift_1))
freq_table
```

```
##    vectorOfShift_1 Freq
## 1                1    1
## 2               14  625
## 3               15    2
## 4               26    1
## 5               27    1
## 6               32  799
## 7               43    1
## 8               52    1
## 9               53    1
## 10              54    1
## 11              55  624
## 12              56   15
## 13              70    2
## 14              71    5
## 15              72    2
## 16              73   21
## 17              74  495
## 18              77  770
## 19              79    3
## 20              83    1
## 21              89    1
## 22              95    2
## 23              98  730
## 24             107    1
## 25             112    1
## 26             118  793
## 27             120    2
## 28             122    1
## 29             124    2
## 30             132    1
## 31             135    1
## 32             137    1
## 33             138    1
## 34             141    1
## 35             163    1
## 36             164  667
## 37             165    2
## 38             173    1
## 39             179    1
## 40             191    1
## 41             192    1
```

Try to find the contrast with variance relatively small, median and large.

```
ind_small4=which(convar4<0.04)
ind_median4=which(convar4<0.06&convar4>=0.04)
ind_large4=which(convar4>=0.085)
median_node4=intnode[ind_median4]
small_node4=intnode[ind_small4]
```

```
large_node4=intnode[ind_large4]
large_node4
```

```
## [1] 193 192 172 163 118
```

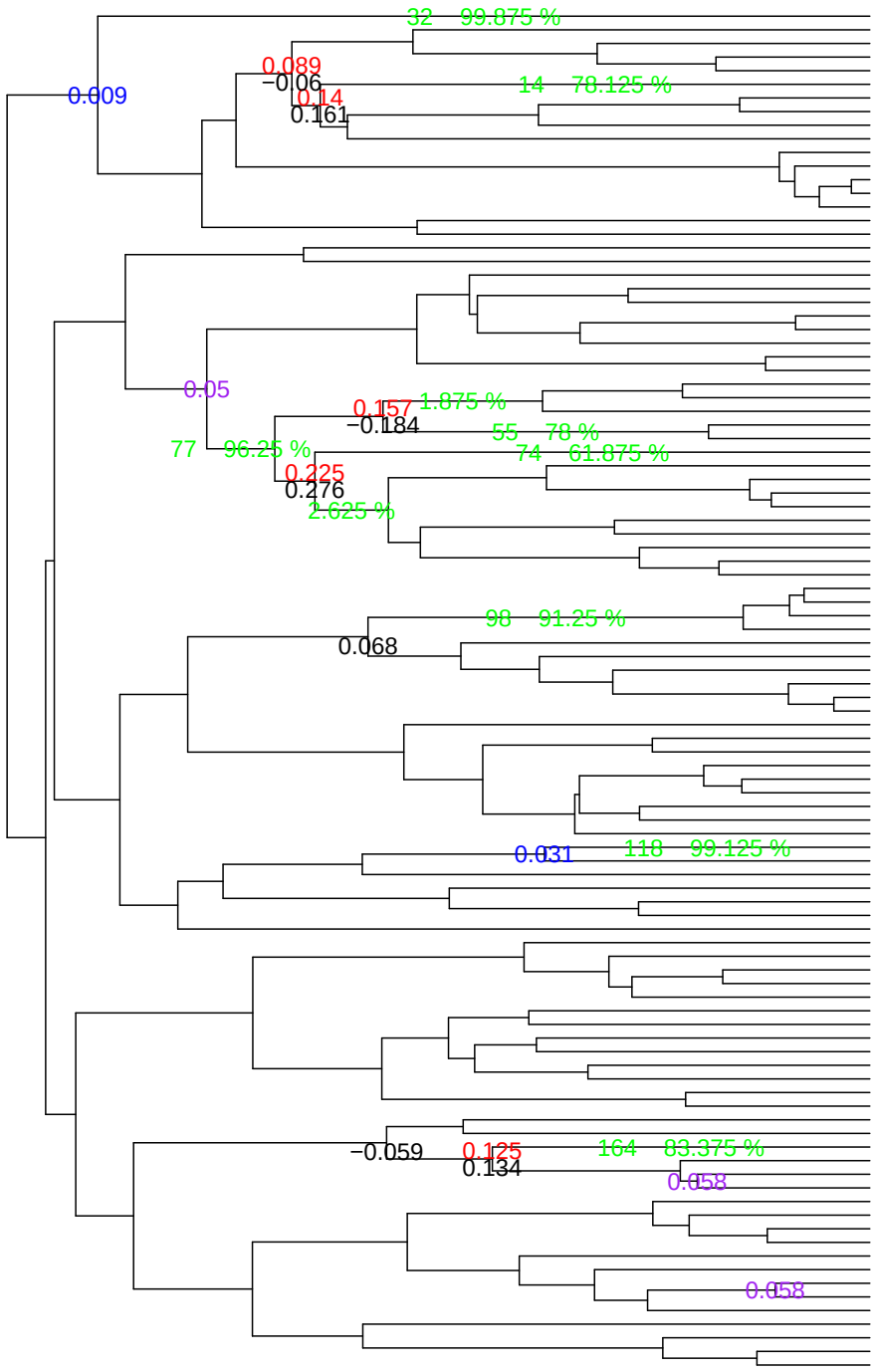
```
mean(convar4[-as.numeric(large_node4)])
```

```
## [1] 0.07022559
```

Even though we ignore those nodes with large variance of contrasts, the variances of contrasts on average are still overestimated.

```
mean_large4=intnode[which(abs(contrast_mean4)>0.05)]
median_node4=intnode[ind_median4]
small_node4=intnode[ind_small4]
large_node4=intnode[ind_large4]
plot(eModel$tree,show.tip.label = F,main="small,median,large contrast nodes",font=8)
nodelabels(round(convar4[ind_small4],3),small_node4,col="blue",frame = "none")
for(i in 1:length(shift_config)){
edgelabels(paste(shift_config[i],"  ",round(freq_table$Freq[which(freq_table$vectorOfShift_1==shift_con
}
#56,73

edgelabels(paste(round(freq_table$Freq[which(freq_table$vectorOfShift_1==56)]/8,3),"%"),56,col="green",
edgelabels(paste(round(freq_table$Freq[which(freq_table$vectorOfShift_1==73)]/8,3),"%"),73,col="green",
nodelabels(round(convar4[ind_median4],3),median_node4,col="purple",frame = "none")
nodelabels(round(convar4[ind_large4],3),large_node4,col="red",frame = "none",adj=c(0.5,0))
nodelabels(round(contrast_mean4[which(abs(contrast_mean4)>0.05)],3),mean_large4,col="black",frame = "no
```


small,median,large contrast nodes
32
99.875 %
0.089
−0.06
0.14
0.161
0.009
14
78.125 %
0.05
0.157
1.875 %
−0.184
77
96.25 %
55
78 %
74
61.875 %
0.225
0.276
2.625 %
98
91.25 %
0.068
0.031
118
99.125 %
−0.059
0.125
0.134
164
83.375 %
0.058
0.058

The

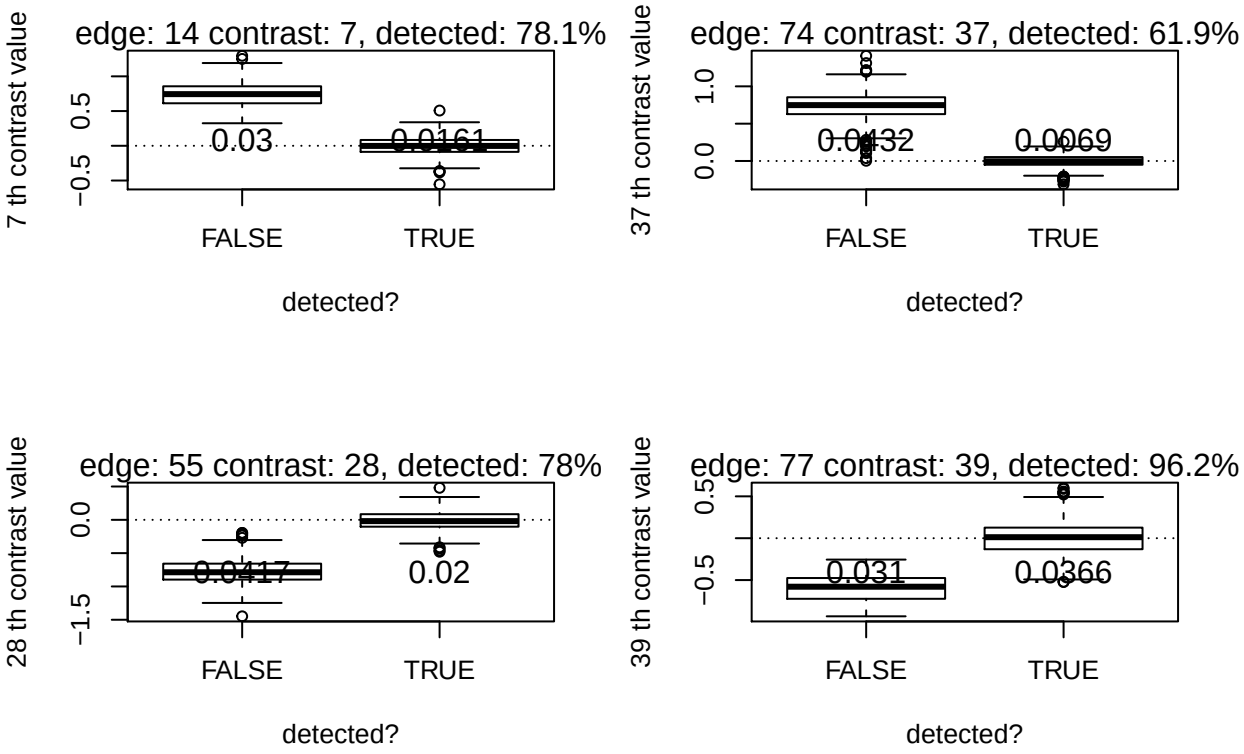
shifts being detected most often are: 32,77,118,164,98,14,55,74 which are exactly corresponding to the shift configuration 55, 98, 118, 74, 14, 77, 32, 164of the true model. A few nodes with shift edges have larger than average mean of contrasts. So, their variances are overestimated accordingly.

Contrasts of node with shift edges being detected or not

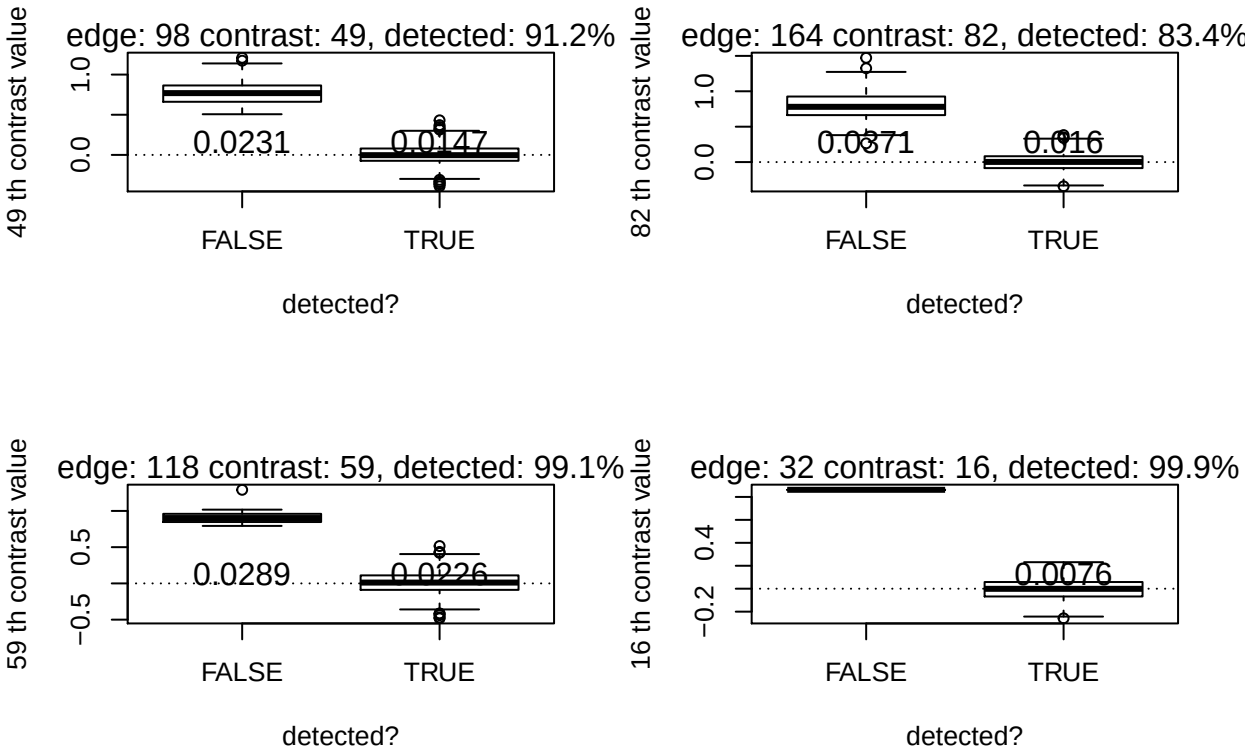
```
contrast_plot = function(edge, contrast){
  contrast_detect=rep(NA, 600)
  for(i in 1:length(vectorOfShift)){
    if(edge%in%vectorOfShift[[i]]){
      contrast_detect[i]=TRUE
    }
    else {
      contrast_detect[i]=FALSE
    }
  }
  plot(contrast_table4[,contrast] ~ factor(contrast_detect), xlab="detected?",ylab=paste(contrast,"th c
  mtext(paste("edge: ",edge," contrast: ",contrast,
              ", detected: ", round(100*sum(contrast_detect)/800, 1), "%", sep=""))
  abline(h=0, lty=3)
  convar_bygroup = tapply(contrast_table4[,contrast], contrast_detect, var)
  mtext(side=1, at=1:2, text=round(convar_bygroup,4), line=-2)
}
```

Those figures show below represent contrasts of different nodes with group1(shift detected) and group2(shift not detected).

```
layout(matrix(1:4,2,2))
contrast_plot(14,7) # edge 14, node 193, contrast 7
contrast_plot(55,28)# edge 55, node 172, contrast 28
contrast_plot(74,37) # edge 74, node 163, contrast 37
contrast_plot(77,39) # edge 77, node 161, contrast 39
```




```
contrast_plot(98,49) # edge 98, node 151, contrast 49
contrast_plot(118,59)# edge 118, node 141, contrast 59
contrast_plot(164,82)# edge 164, node 118, contrast 82
contrast_plot(32,16)# last edge 32, node 184 , contrast 16
```



It is clear to see nodes with shift edges being detected have mean of contrasts close to zero. In some cases when those shift edges have not been detected have very different contrast values. Most of contrasts with shift edge not being detected have larger variances than variance of contrasts with shift edges being detected. Edge 77 is especially strange because it does not follow last rule. (Need some explaination)

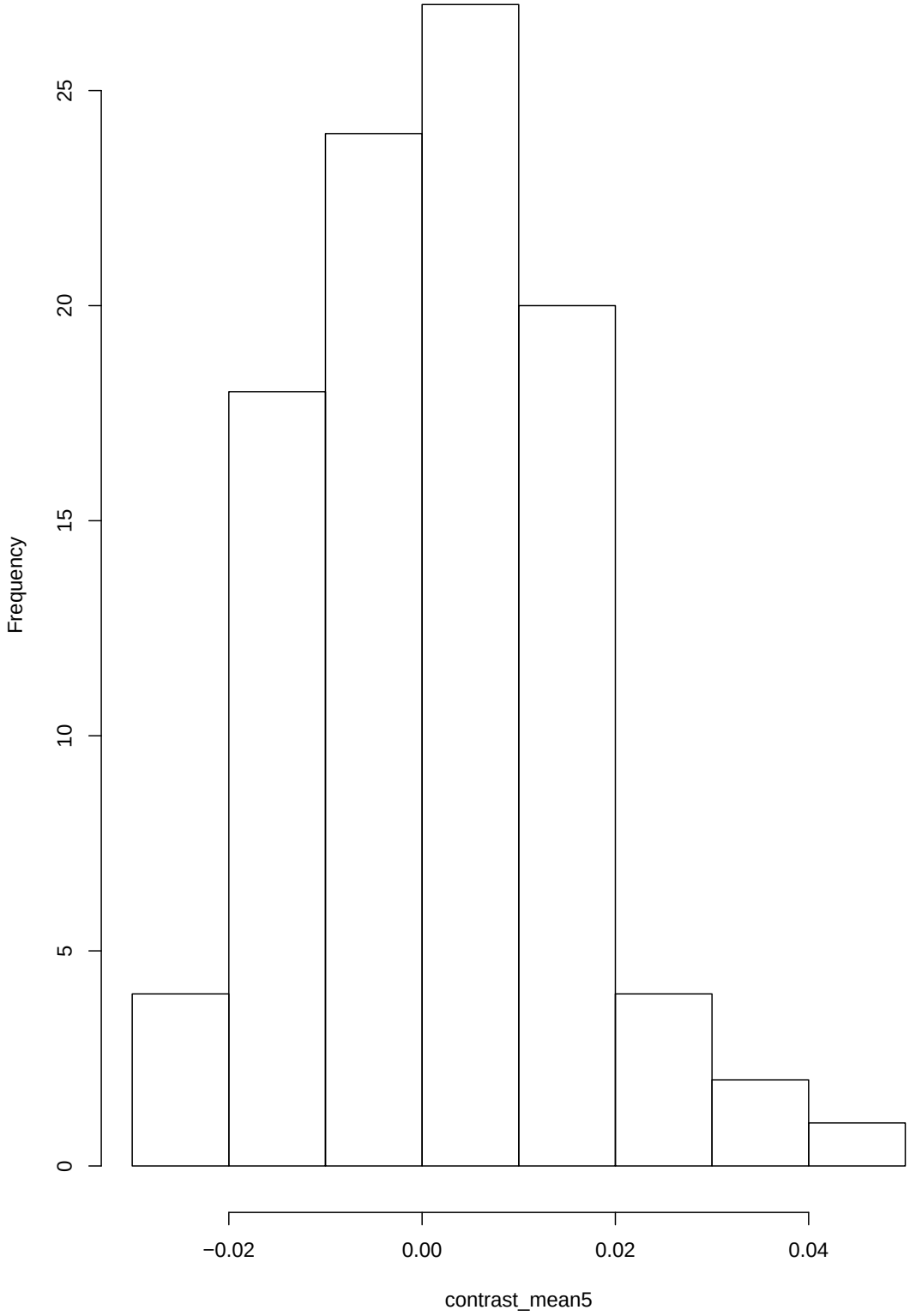
```
load("unknownconfig_alpha_4.2.RData")
load("unknownconfig_shifts_4.2.RData")
load("unknownconfig_sigma2_4.2.RData")
load("unknownconfig_mu_4.2.RData")
load("unknownconfig_shiftvalues_4.2.RData")
load("unknownconfig_contrast_4.2.RData")
#Calculate mean of contrasts
contrast_mean5=colMeans(contrast_table5)
head(contrast_mean5)
```

```
## [1] -0.013177648  0.004979592 -0.002567778 -0.013514359  0.020714921
## [6]  0.015617662
```

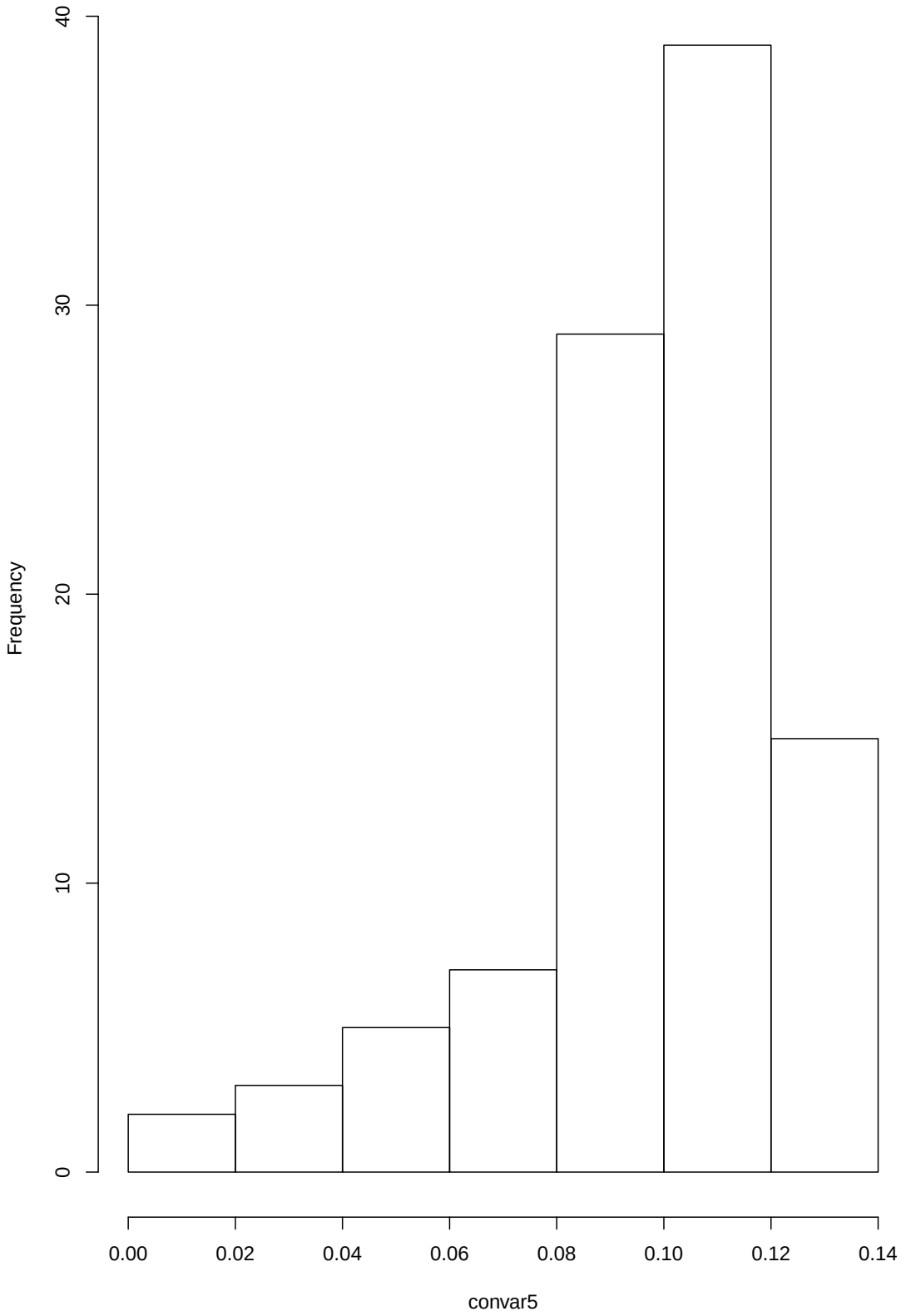
```
max(abs(contrast_mean5))
```

```
## [1] 0.04368802
```

```
hist(contrast_mean5)
```


Histogram of contrast_mean5
Frequency
0
5
10
15
20
25
−0.02
0.00
0.02
0.04
contrast_mean5

```r
#Calculate variance of contrasts
convar5=apply(contrast_table5,2,ss)
hist(convar5)
```


Histogram of convar5
Frequency
0
10
20
30
40
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
convar5

```r
#Shifts that occur the most often
vectorOfShift_2=unlist(vectorOfShift2, recursive = TRUE, use.names = F)
freq_table2=as.data.frame(table(vectorOfShift_2))
freq_table2
```

```
##     vectorOfShift_2 Freq
## 1                 1    1
## 2                 2    1
## 3                 3   15
## 4                 4    6
## 5                 5   17
## 6                 6   10
## 7                 7    2
## 8                 8    2
## 9                 9    5
## 10               10   12
## 11               11   14
## 12               12   21
## 13               13   20
## 14               14  777
## 15               15   30
## 16               16   31
## 17               25   34
## 18               26   24
## 19               27   13
## 20               28   12
## 21               29   32
## 22               30   28
## 23               31   55
## 24               32  782
## 25               33   24
## 26               34   12
## 27               35    2
## 28               36    5
## 29               39    6
## 30               40   14
## 31               41   13
## 32               42   13
## 33               43   19
## 34               44   16
## 35               47    6
## 36               48   16
## 37               49    1
## 38               50    5
## 39               51    6
## 40               52   14
## 41               53    2
## 42               54    1
## 43               55  789
## 44               56   43
## 45               57    1
## 46               58    1
## 47               59    1
## 48               61    6
```

```
## 49                62    6
## 50                63    2
## 51                64    9
## 52                65    4
## 53                67    2
## 54                68    9
## 55                69   24
## 56                70   20
## 57                71   30
## 58                72   18
## 59                73   29
## 60                74  752
## 61                77  795
## 62                78   41
## 63                79   35
## 64                80   31
## 65                82    1
## 66                83    1
## 67                84    1
## 68                89    1
## 69                90    1
## 70                91    6
## 71                92    6
## 72                93    7
## 73                94   10
## 74                95   21
## 75                96    6
## 76                97   41
## 77                98  794
## 78                99    4
## 79               100    4
## 80               101    2
## 81               103    6
## 82               104    3
## 83               105    5
## 84               106    3
## 85               107   10
## 86               108   10
## 87               109    5
## 88               110   19
## 89               111   18
## 90               112   14
## 91               113   19
## 92               114   11
## 93               115   30
## 94               116   28
## 95               117   21
## 96               118  800
## 97               119    9
## 98               120   13
## 99               121    1
## 100              122    5
## 101              123   19
## 102              124   12
```

```
## 103             125   29
## 104             126   14
## 105             127   28
## 106             128   33
## 107             129   32
## 108             130   37
## 109             131   36
## 110             132   31
## 111             133    1
## 112             134    2
## 113             135    7
## 114             136    7
## 115             137   14
## 116             138    6
## 117             139   23
## 118             140    9
## 119             141    9
## 120             142    6
## 121             143    2
## 122             144    6
## 123             145    6
## 124             146    6
## 125             147   14
## 126             148   16
## 127             149   18
## 128             150   10
## 129             151    3
## 130             152    2
## 131             153   23
## 132             154   29
## 133             155   32
## 134             156   33
## 135             157   16
## 136             158    5
## 137             160    4
## 138             161    3
## 139             162    2
## 140             163   28
## 141             164  781
## 142             165   24
## 143             166   26
## 144             169    4
## 145             170    4
## 146             171    5
## 147             172    3
## 148             173    1
## 149             175    4
## 150             176    2
## 151             177    9
## 152             178    9
## 153             179   24
## 154             180   10
## 155             181   26
## 156             182   36
```

```
## 157             183    1
## 158             186    1
## 159             187   22
## 160             188   15
## 161             189   18
## 162             190   41
## 163             191   34
## 164             192   29
## 165             193   39
## 166             194   42
## 167             195   45
## 168             196   31
## 169             197   74
```

Try to find the contrast with variance relatively small, median and large.

```
ind_small5=which(convar5<0.04)
ind_median5=which(convar5<0.06&convar5>=0.04)
ind_large5=which(convar5>=0.085)
median_node5=intnode[ind_median5]
small_node5=intnode[ind_small5]
large_node5=as.numeric(intnode[ind_large5])
large_node5
```

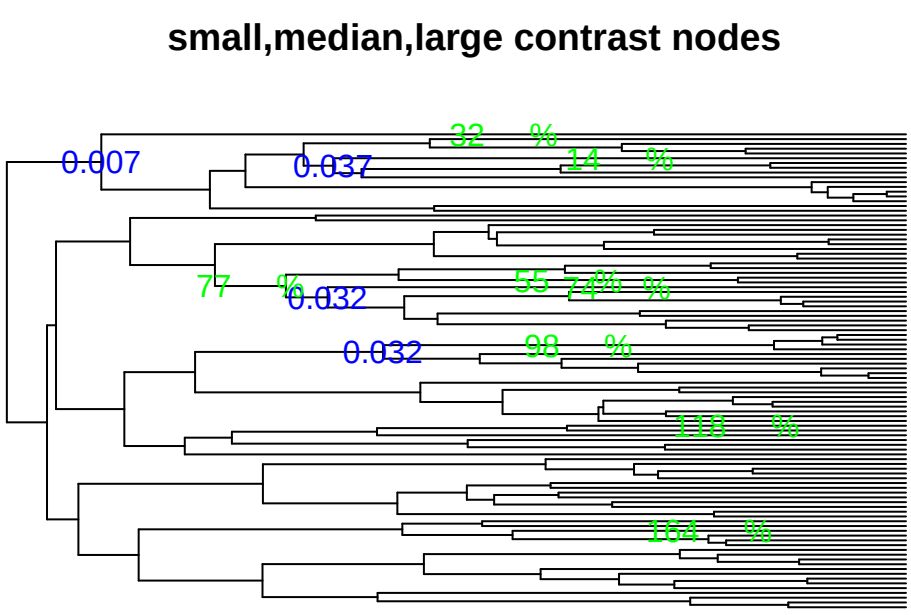
```
##  [1] 199 198 197 196 195 194 192 188 187 186 185 183 182 180 179 178 176
## [18] 175 174 173 170 169 168 167 166 165 164 162 160 157 154 153 152 150
## [35] 149 148 147 146 145 144 143 142 140 139 138 137 136 135 133 132 131
## [52] 130 129 128 127 126 125 124 123 122 121 120 119 117 115 114 112 111
## [69] 110 109 108 107 106 105 104 103
```

```
mean(convar5[-large_node5])
```

```
## [1] 0.09682729
```

Even though we ignore those nodes with large variance of contrasts, the variances of contrasts on average are still overestimated.

```
mean_large5=intnode[which(abs(contrast_mean5)>0.05)]
median_node5=intnode[ind_median5]
small_node5=intnode[ind_small5]
large_node5=intnode[ind_large5]
plot(eModel$tree,show.tip.label = F,main="small,median,large contrast nodes",font=8)
nodelabels(round(convar5[ind_small5],3),small_node5,col="blue",frame = "none")
for(i in 1:length(shift_config)){
edgelabels(paste(shift_config[i],"  ",round(freq_table2$Freq[which(freq_table2$vectorOfShift_1==shift_c
}
```



#56,73

The shifts being detected most often are: 32,77,118,164,98,14,55,74 which are exactly corresponding to the shift configuration 55, 98, 118, 74, 14, 77, 32, 164of the true model. A few nodes with shift edges have larger than average mean of contrasts. So, their variances are overestimated accordingly.

Contrasts of node with shift edges being detected or not

```
contrast_plot = function(edge, contrast){
  contrast_detect=rep(NA, 600)
  for(i in 1:length(vectorOfShift2)){
    if(edge%in%vectorOfShift2[[i]]){
      contrast_detect[i]=TRUE
    }
    else {
      contrast_detect[i]=FALSE
    }
  }
  plot(contrast_table5[,contrast] ~ factor(contrast_detect), xlab="detected?",ylab=paste(contrast,"th c
  mtext(paste("edge: ",edge," contrast: ",contrast,
              ", detected: ", round(100*sum(contrast_detect)/800, 1), "%", sep=""))
  abline(h=0, lty=3)
  convar_bygroup = tapply(contrast_table5[,contrast], contrast_detect, var)
  mtext(side=1, at=1:2, text=round(convar_bygroup,4), line=-2)
}
```

Those figures show below represent contrasts of different nodes with group1(shift detected) and group2(shift not detected).

```
layout(matrix(1:4,2,2))
contrast_plot(14,7) # edge 14, node 193, contrast 7
contrast_plot(55,28)# edge 55, node 172, contrast 28
contrast_plot(74,37) # edge 74, node 163, contrast 37
contrast_plot(77,39) # edge 77, node 161, contrast 39

contrast_plot(98,49) # edge 98, node 151, contrast 49
contrast_plot(118,59)# edge 118, node 141, contrast 59
contrast_plot(164,82)# edge 164, node 118, contrast 82
contrast_plot(32,16)# last edge 32, node 184 , contrast 16
```