Ancestral state reconstruction (incl. diagnostics)

Beatriz Willink Rachel Blow

May 2020

Contents

Analysis of BayesTraits output

Output from each independent run should be saved into its own subdirectory within the BayesTraits directory.

Set your working directory to the parent directory containing all of your MCMC analyses

```
knitr::opts_knit$set(root.dir = '..' )
```

1) Read in data as mcmc objects

Read in your traces and make sure they are named correctly

```
path <- ("BayesTraits/")
directories <- c("run1/","run2/")
filename <- "IschMultiState.Log.txt"

for (i in directories){
  x = c(path, i, filename)
  assign(paste("run", which(directories==i), sep=""),
         as.data.frame(read.delim(paste(x, collapse =""), header=T, skip = 129,
                                  na.strings = "--")))
}
```

Set the variables that will be used to create your MCMC object

```
thin <- run1$Iteration[2] - run1$Iteration[1]
start <- min(run1$Iteration)
end <- max(run1$Iteration)
```

Remove variables we are not interested in

```
output <- run1[ , !(names(run1) %in% c("X", "Model.string"))]
```

Divide rates by 100 to get realistic values (because ScaledTrees to 0.001 to prevent rates getting too small)

```
for(i in 6:ncol(output[,1:17])) {
  output[,i] <- output[,i]/100
}
```

Call the "coda" package for creating multiple mcmc objects

```
require(coda)
```

Create mcmc object

```
outputmc <- mcmc(data = output, start = start, end = end, thin = thin)
```

Calculate the percentage of iterations in which there was only a single parameter

```
(sum(outputmc[,4] == 1, na.rm = TRUE)/length(outputmc[,4]))*100
```

```
## [1] 92.975
```

Summarize each parameter into mean and HPD interval of percent support for each state per node

```
nodes_output <- outputmc[,22:181]
summary_table <- matrix(0,3,ncol(nodes_output))
rownames(summary_table) <- c("Mean", "LowerHPD", "UpperHPD")
colnames(summary_table) <- colnames(nodes_output)

for(i in 1:ncol(nodes_output)){
  summary_table[1,i] <- mean(nodes_output[,i], na.rm =TRUE)
  hpd <- HPDinterval(nodes_output[,i])
  summary_table[2,i] <- hpd[1]
  summary_table[3,i] <- hpd[2]
}
print(summary_table)
```

```
##          RecNode42.P.0. RecNode42.P.1. RecNode42.P.2. RecNode42.P.3.
## Mean         0.06197866     0.07791162      0.8126004     0.04750933
## LowerHPD     0.00000000     0.00000000      0.2522950     0.00000000
## UpperHPD     0.25115800     0.28243800      1.0000000     0.25000000
##          RecNode43.P.0. RecNode43.P.1. RecNode43.P.2. RecNode43.P.3.
```

```
## Mean         0.08894633     0.04577913      0.8137379     0.05153659
## LowerHPD     0.00000000     0.00000000      0.3099860     0.00000000
## UpperHPD     0.25874200     0.23402700      1.0000000     0.24652100
##          RecNode44.P.0. RecNode44.P.1. RecNode44.P.2. RecNode44.P.3.
## Mean         0.04960731     0.02749367      0.8825785     0.04032047
## LowerHPD     0.00000000     0.00000000      0.4610340     0.00000000
## UpperHPD     0.21952300     0.15276300      1.0000000     0.20951800
##          RecNode45.P.0. RecNode45.P.1. RecNode45.P.2. RecNode45.P.3.
## Mean         0.02376308     0.01031439       0.946610      0.0193125
## LowerHPD     0.00000000     0.00000000       0.710454      0.0000000
## UpperHPD     0.12795200     0.04826300       1.000000      0.1174190
##          RecNode46.P.0. RecNode46.P.1. RecNode46.P.2. RecNode46.P.3.
## Mean         0.03434901      0.0165108      0.9213977     0.02774252
## LowerHPD     0.00000000      0.0000000      0.6072060     0.00000000
## UpperHPD     0.16855000      0.0855070      1.0000000     0.15491400
##          RecNode47.P.0. RecNode47.P.1. RecNode47.P.2. RecNode47.P.3.
## Mean          0.1723913     0.09209879      0.6564426     0.07906733
## LowerHPD      0.0000000     0.00000000      0.2603650     0.00000000
## UpperHPD      0.4435180     0.26787700      1.0000000     0.25000000
##          RecNode48.P.0. RecNode48.P.1. RecNode48.P.2. RecNode48.P.3.
## Mean          0.2370579      0.1400341      0.5287665     0.09414155
## LowerHPD      0.0000000      0.0000000      0.1632390     0.00000000
## UpperHPD      0.6120400      0.3862060      1.0000000     0.26236100
##          RecNode49.P.0. RecNode49.P.1. RecNode49.P.2. RecNode49.P.3.
## Mean         0.06639444      0.1896542      0.6920832     0.05186825
## LowerHPD     0.00000000      0.0000000      0.2182490     0.00000000
## UpperHPD     0.28372000      0.4843350      1.0000000     0.25143300
##          RecNode50.P.0. RecNode50.P.1. RecNode50.P.2. RecNode50.P.3.
## Mean          0.4461582     0.08112336       0.388481     0.08423751
## LowerHPD      0.0000000     0.00000000       0.043522     0.00000000
## UpperHPD      0.9123970     0.36321800       1.000000     0.43593600
##          RecNode51.P.0. RecNode51.P.1. RecNode51.P.2. RecNode51.P.3.
## Mean         0.08544433      0.2189465      0.6281996     0.06740958
## LowerHPD     0.00000000      0.0000000      0.1669460     0.00000000
## UpperHPD     0.25118000      0.6680800      1.0000000     0.25015000
##          RecNode52.P.0. RecNode52.P.1. RecNode52.P.2. RecNode52.P.3.
## Mean         0.07384264     0.07831754      0.7960704     0.05176939
## LowerHPD     0.00000000     0.00000000      0.2707860     0.00000000
## UpperHPD     0.25111300     0.25967400      1.0000000     0.24986400
##          RecNode53.P.0. RecNode53.P.1. RecNode53.P.2. RecNode53.P.3.
## Mean         0.05837912     0.04902521      0.8451342     0.04746149
## LowerHPD     0.00000000     0.00000000      0.3389170     0.00000000
## UpperHPD     0.24827600     0.21705000      1.0000000     0.24414200
##          RecNode54.P.0. RecNode54.P.1. RecNode54.P.2. RecNode54.P.3.
## Mean          0.0754051     0.09201646      0.7723016     0.06027682
## LowerHPD      0.0000000     0.00000000      0.2913970     0.00000000
## UpperHPD      0.2646710     0.26547600      1.0000000     0.25366700
##          RecNode55.P.0. RecNode55.P.1. RecNode55.P.2. RecNode55.P.3.
## Mean         0.04916498     0.02539382      0.8862738     0.03916735
## LowerHPD     0.00000000     0.00000000      0.5248300     0.00000000
## UpperHPD     0.19892800     0.12800100      1.0000000     0.18990700
##          RecNode56.P.0. RecNode56.P.1. RecNode56.P.2. RecNode56.P.3.
## Mean          0.1454618      0.1898386      0.5819883     0.08271131
## LowerHPD      0.0000000      0.0000000      0.2291870     0.00000000
```

```
## UpperHPD      0.3300860      0.4977900      1.0000000     0.25730200
##           RecNode57.P.0. RecNode57.P.1. RecNode57.P.2. RecNode57.P.3.
## Mean           0.2225106      0.6769667     0.03708742     0.06343525
## LowerHPD       0.0000000      0.4044330     0.00000000     0.00000000
## UpperHPD       0.4090590      1.0000000     0.11522600     0.23751200
##           RecNode58.P.0. RecNode58.P.1. RecNode58.P.2. RecNode58.P.3.
## Mean          0.01925603      0.9476715     0.01290252     0.02016991
## LowerHPD      0.00000000      0.7022860     0.00000000     0.00000000
## UpperHPD      0.11554200      1.0000000     0.05872600     0.12275000
##           RecNode59.P.0. RecNode59.P.1. RecNode59.P.2. RecNode59.P.3.
## Mean          0.02233915      0.9413872     0.01294699     0.02332662
## LowerHPD      0.00000000      0.6523680     0.00000000     0.00000000
## UpperHPD      0.14952200      1.0000000     0.04899000     0.16446000
##           RecNode60.P.0. RecNode60.P.1. RecNode60.P.2. RecNode60.P.3.
## Mean         0.003800219      0.9922199    0.001025845    0.002954035
## LowerHPD     0.000000000      0.9676790    0.000000000    0.000000000
## UpperHPD     0.010561000      1.0000000    0.002446000    0.011079000
##           RecNode61.P.0. RecNode61.P.1. RecNode61.P.2. RecNode61.P.3.
## Mean         0.004153032      0.9942771   0.0003266431    0.001243226
## LowerHPD     0.000000000      0.9835680   0.0000000000    0.000000000
## UpperHPD     0.008832000      1.0000000   0.0007350000    0.003394000
##           RecNode62.P.0. RecNode62.P.1. RecNode62.P.2. RecNode62.P.3.
## Mean          0.01757444      0.9806371   0.0002397577    0.001548669
## LowerHPD      0.00000000      0.9094960   0.0000000000    0.000000000
## UpperHPD      0.07850300      1.0000000   0.0005830000    0.002692000
##           RecNode63.P.0. RecNode63.P.1. RecNode63.P.2. RecNode63.P.3.
## Mean        0.0001827141      0.9995795   5.212723e-05   0.0001856493
## LowerHPD    0.0000000000      0.9993340   0.000000e+00   0.0000000000
## UpperHPD    0.0002000000      1.0000000   1.020000e-04   0.0002390000
##           RecNode64.P.0. RecNode64.P.1. RecNode64.P.2. RecNode64.P.3.
## Mean           0.4183228      0.5790258   0.0002818835    0.002369453
## LowerHPD       0.0000000      0.4943030   0.0000000000    0.000000000
## UpperHPD       0.5011900      1.0000000   0.0010490000    0.004015000
##           RecNode65.P.0. RecNode65.P.1. RecNode65.P.2. RecNode65.P.3.
## Mean          0.01098469      0.9709242    0.006161697     0.01192937
## LowerHPD      0.00000000      0.8516320    0.000000000     0.00000000
## UpperHPD      0.05215700      1.0000000    0.021023000     0.06179000
##           RecNode66.P.0. RecNode66.P.1. RecNode66.P.2. RecNode66.P.3.
## Mean           0.0493281     0.04683073      0.8642792     0.03956194
## LowerHPD       0.0000000     0.00000000      0.3760630     0.00000000
## UpperHPD       0.2475100     0.19503900      1.0000000     0.23750400
##           RecNode67.P.0. RecNode67.P.1. RecNode67.P.2. RecNode67.P.3.
## Mean          0.08221074       0.190986      0.6628412     0.06396204
## LowerHPD      0.00000000       0.000000      0.2334620     0.00000000
## UpperHPD      0.27508800       0.474359      1.0000000     0.25427600
##           RecNode68.P.0. RecNode68.P.1. RecNode68.P.2. RecNode68.P.3.
## Mean          0.02634249     0.01227446      0.9400527     0.02133032
## LowerHPD      0.00000000     0.00000000      0.6755690     0.00000000
## UpperHPD      0.13970300     0.06721200      1.0000000     0.12671200
##           RecNode69.P.0. RecNode69.P.1. RecNode69.P.2. RecNode69.P.3.
## Mean          0.08661702      0.7326912     0.09113272     0.08955902
## LowerHPD      0.00000000      0.3138350     0.00000000     0.00000000
## UpperHPD      0.24643900      1.0000000     0.23574500     0.25092700
##           RecNode70.P.0. RecNode70.P.1. RecNode70.P.2. RecNode70.P.3.
```

```
## Mean         0.06663678      0.7985777       0.063143     0.07164248
## LowerHPD     0.00000000      0.3494320       0.000000     0.00000000
## UpperHPD     0.23710700      0.9999390       0.197986     0.25014500
##          RecNode71.P.0. RecNode71.P.1. RecNode71.P.2. RecNode71.P.3.
## Mean         0.03521686      0.8969575     0.02120357     0.04662205
## LowerHPD     0.00000000      0.4738710     0.00000000     0.00000000
## UpperHPD     0.19198700      0.9999820     0.07918200     0.24940100
##          RecNode72.P.0. RecNode72.P.1. RecNode72.P.2. RecNode72.P.3.
## Mean         0.02733537      0.9140991    0.009930028     0.04863546
## LowerHPD     0.00000000      0.4794890    0.000000000     0.00000000
## UpperHPD     0.16675900      0.9999980    0.037031000     0.27649900
##          RecNode73.P.0. RecNode73.P.1. RecNode73.P.2. RecNode73.P.3.
## Mean         0.06682086       0.696290      0.0227247      0.2141645
## LowerHPD     0.00000000       0.118964      0.0000000      0.0000000
## UpperHPD     0.26057700       0.999362      0.0700810      0.7209070
##          RecNode74.P.0. RecNode74.P.1. RecNode74.P.2. RecNode74.P.3.
## Mean         0.04246134      0.4362448    0.002071195      0.5192227
## LowerHPD     0.00000000      0.0107080    0.000000000      0.0000000
## UpperHPD     0.23842800      1.0000000    0.006025000      0.9608110
##          RecNode75.P.0. RecNode75.P.1. RecNode75.P.2. RecNode75.P.3.
## Mean        0.000159899   1.605505e-05   5.277868e-06      0.9998188
## LowerHPD    0.000000000   0.000000e+00   0.000000e+00      0.9996020
## UpperHPD    0.000316000   6.900000e-05   2.000000e-06      1.0000000
##          RecNode76.P.0. RecNode76.P.1. RecNode76.P.2. RecNode76.P.3.
## Mean       8.361413e-05   9.962544e-06   4.319435e-06      0.9999021
## LowerHPD   0.000000e+00   0.000000e+00   0.000000e+00      0.9998030
## UpperHPD   1.460000e-04   3.600000e-05   1.000000e-06      1.0000000
##          RecNode77.P.0. RecNode77.P.1. RecNode77.P.2. RecNode77.P.3.
## Mean         0.01755004      0.9495664     0.01137519     0.02150835
## LowerHPD     0.00000000      0.6690070     0.00000000     0.00000000
## UpperHPD     0.12419300      0.9999940     0.04536700     0.15643500
##          RecNode78.P.0. RecNode78.P.1. RecNode78.P.2. RecNode78.P.3.
## Mean         0.03766707      0.8943355     0.03065718     0.03734029
## LowerHPD     0.00000000      0.5254530     0.00000000     0.00000000
## UpperHPD     0.18292700      0.9999250     0.11899000     0.19161400
##          RecNode79.P.0. RecNode79.P.1. RecNode79.P.2. RecNode79.P.3.
## Mean         0.02186025      0.9406234     0.01456517     0.02295122
## LowerHPD     0.00000000      0.6833900     0.00000000     0.00000000
## UpperHPD     0.12649100      0.9999700     0.05657000     0.14760400
##          RecNode80.P.0. RecNode80.P.1. RecNode80.P.2. RecNode80.P.3.
## Mean         0.02125843      0.9430864     0.01353523     0.02211992
## LowerHPD     0.00000000      0.6967180     0.00000000     0.00000000
## UpperHPD     0.12159000      1.0000000     0.04726100     0.13578900
##          RecNode81.P.0. RecNode81.P.1. RecNode81.P.2. RecNode81.P.3.
## Mean        0.003183999      0.9920005    0.001254813     0.00356068
## LowerHPD    0.000000000      0.9669360    0.000000000     0.00000000
## UpperHPD    0.011047000      1.0000000    0.003312000     0.01290200
```

Reshape data into mean posterior value by node (row) and state (column)

```
mean_state_per_node <- data.frame("node" = 42:81, matrix(summary_table[1,], ncol = 4, byrow = TRUE))
```

2) Build tree displaying ancestral state reconstruction (Fig. 4)

Call "treeio" and "ggtree" packages for building phylogenetic tree

```
require(treeio)
require(ggtree)
```

Read in StarBEAST2 summary tree

```
beast_tree <- read.beast("StarBEAST2/summary.tree")
```

Create a new tip label variable in tree data

```
relabelling <- data.frame("label" = fortify(beast_tree)$label[1:41])
```

```
## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
relabelling$label2 <- sub("_", " ", relabelling$label)
relabelling$label2 <- plyr::revalue(relabelling$label2,
                                    c("Amorphostigma armstrongi" = "Ischnura armstrongi",
                                      "Amorphostigma sp." = "Ischnura sp.",
                                      "Rhodischnura nursei" = "Ischnura nursei"))

beast_tree <- full_join(beast_tree, relabelling, by ="label")
```

```
## Warning: Column `label` joining character vector and factor, coercing into
## character vector
```

Read in female morph state data

```
extant_state <- read.table("BayesTraits/FMorph.txt")
```

Call "dplyr" for manipulating dataframe

```
require(dplyr)
```

Create dataframe for pie charts at tips (female morph state of extant species as proposed by the literature) and add it to tree data

```
extant_state <- data.frame("node"= 1:41,
                           arrange(extant_state, match(extant_state$V1,
                                                       fortify(beast_tree)$label[1:41])))
extant_state$V2 <- factor(extant_state$V2)
beast_tree <- full_join(beast_tree, extant_state, by ="node")
```

Create dataframe for pie charts at nodes (mean posterior estimate for each state at each node and percentage of posterior that significantly predicted the dominant state)

```
pie_labels <- data.frame(mean_state_per_node,
                         "x" = fortify(beast_tree)$x[42:81],
                         "y" = fortify(beast_tree)$y[42:81])
```

Call "ggplot2" for drawing phylogeny

```
require(ggplot2)
require(scatterpie)
```

Create phylogeny with female morph states as pies and confidence percentage as text labels at nodes

```
tree <- ggtree(beast_tree) +
  coord_cartesian(clip = 'off') +
  geom_tiplab(aes(label = label2), size=3, offset = 1) +
  theme_tree(plot.margin=margin(6, 160, 6, 20), legend.position = c(0.34, 1)) +
  geom_tippoint(aes(x = x+0.5, color = V2), size = 3) +
  geom_scatterpie(data=pie_labels, mapping = aes(x=x, y=y, r=0.5), color=NA, cols = 2:5,
                  alpha=0.8) +
  scale_color_manual(values = c("0"="#56B4E9", "1"="#D55E00", "2"="#009E73", "3"="#CC79A7")) +
  scale_fill_manual(values = c("#56B4E9", "#D55E00", "#009E73", "#CC79A7"),
                    labels = c("Female-monomorphic with androchrome females",
                               "Female-dimorphic",
                               "Female-monomorphic with heterochrome females",
                               "Female-trimorphic"),
                    name = "Female Morph Status")
```

Save phylogeny

```
ggsave("Fig4.pdf", tree, path = "Figures", width = 7, height = 9)
```

3) Prepare data for diagnostics

Remove any parameters that are not of further interest (i.e. anything that is not an iteration, likelihood or rate parameter)

```
attributes(run1)$names
```

```
##   [1] "Iteration"        "Lh"               "Tree.No"
##   [4] "No.Off.Parmeters" "No.Off.Zero"      "Model.string"
##   [7] "q01"              "q02"              "q03"
##  [10] "q10"              "q12"              "q13"
##  [13] "q20"              "q21"              "q23"
##  [16] "q30"              "q31"              "q32"
##  [19] "Root.P.0."        "Root.P.1."        "Root.P.2."
##  [22] "Root.P.3."        "RecNode42.P.0."   "RecNode42.P.1."
##  [25] "RecNode42.P.2."   "RecNode42.P.3."   "RecNode43.P.0."
##  [28] "RecNode43.P.1."   "RecNode43.P.2."   "RecNode43.P.3."
##  [31] "RecNode44.P.0."   "RecNode44.P.1."   "RecNode44.P.2."
```

```
##  [34] "RecNode44.P.3."   "RecNode45.P.0."   "RecNode45.P.1."
##  [37] "RecNode45.P.2."   "RecNode45.P.3."   "RecNode46.P.0."
##  [40] "RecNode46.P.1."   "RecNode46.P.2."   "RecNode46.P.3."
##  [43] "RecNode47.P.0."   "RecNode47.P.1."   "RecNode47.P.2."
##  [46] "RecNode47.P.3."   "RecNode48.P.0."   "RecNode48.P.1."
##  [49] "RecNode48.P.2."   "RecNode48.P.3."   "RecNode49.P.0."
##  [52] "RecNode49.P.1."   "RecNode49.P.2."   "RecNode49.P.3."
##  [55] "RecNode50.P.0."   "RecNode50.P.1."   "RecNode50.P.2."
##  [58] "RecNode50.P.3."   "RecNode51.P.0."   "RecNode51.P.1."
##  [61] "RecNode51.P.2."   "RecNode51.P.3."   "RecNode52.P.0."
##  [64] "RecNode52.P.1."   "RecNode52.P.2."   "RecNode52.P.3."
##  [67] "RecNode53.P.0."   "RecNode53.P.1."   "RecNode53.P.2."
##  [70] "RecNode53.P.3."   "RecNode54.P.0."   "RecNode54.P.1."
##  [73] "RecNode54.P.2."   "RecNode54.P.3."   "RecNode55.P.0."
##  [76] "RecNode55.P.1."   "RecNode55.P.2."   "RecNode55.P.3."
##  [79] "RecNode56.P.0."   "RecNode56.P.1."   "RecNode56.P.2."
##  [82] "RecNode56.P.3."   "RecNode57.P.0."   "RecNode57.P.1."
##  [85] "RecNode57.P.2."   "RecNode57.P.3."   "RecNode58.P.0."
##  [88] "RecNode58.P.1."   "RecNode58.P.2."   "RecNode58.P.3."
##  [91] "RecNode59.P.0."   "RecNode59.P.1."   "RecNode59.P.2."
##  [94] "RecNode59.P.3."   "RecNode60.P.0."   "RecNode60.P.1."
##  [97] "RecNode60.P.2."   "RecNode60.P.3."   "RecNode61.P.0."
## [100] "RecNode61.P.1."   "RecNode61.P.2."   "RecNode61.P.3."
## [103] "RecNode62.P.0."   "RecNode62.P.1."   "RecNode62.P.2."
## [106] "RecNode62.P.3."   "RecNode63.P.0."   "RecNode63.P.1."
## [109] "RecNode63.P.2."   "RecNode63.P.3."   "RecNode64.P.0."
## [112] "RecNode64.P.1."   "RecNode64.P.2."   "RecNode64.P.3."
## [115] "RecNode65.P.0."   "RecNode65.P.1."   "RecNode65.P.2."
## [118] "RecNode65.P.3."   "RecNode66.P.0."   "RecNode66.P.1."
## [121] "RecNode66.P.2."   "RecNode66.P.3."   "RecNode67.P.0."
## [124] "RecNode67.P.1."   "RecNode67.P.2."   "RecNode67.P.3."
## [127] "RecNode68.P.0."   "RecNode68.P.1."   "RecNode68.P.2."
## [130] "RecNode68.P.3."   "RecNode69.P.0."   "RecNode69.P.1."
## [133] "RecNode69.P.2."   "RecNode69.P.3."   "RecNode70.P.0."
## [136] "RecNode70.P.1."   "RecNode70.P.2."   "RecNode70.P.3."
## [139] "RecNode71.P.0."   "RecNode71.P.1."   "RecNode71.P.2."
## [142] "RecNode71.P.3."   "RecNode72.P.0."   "RecNode72.P.1."
## [145] "RecNode72.P.2."   "RecNode72.P.3."   "RecNode73.P.0."
## [148] "RecNode73.P.1."   "RecNode73.P.2."   "RecNode73.P.3."
## [151] "RecNode74.P.0."   "RecNode74.P.1."   "RecNode74.P.2."
## [154] "RecNode74.P.3."   "RecNode75.P.0."   "RecNode75.P.1."
## [157] "RecNode75.P.2."   "RecNode75.P.3."   "RecNode76.P.0."
## [160] "RecNode76.P.1."   "RecNode76.P.2."   "RecNode76.P.3."
## [163] "RecNode77.P.0."   "RecNode77.P.1."   "RecNode77.P.2."
## [166] "RecNode77.P.3."   "RecNode78.P.0."   "RecNode78.P.1."
## [169] "RecNode78.P.2."   "RecNode78.P.3."   "RecNode79.P.0."
## [172] "RecNode79.P.1."   "RecNode79.P.2."   "RecNode79.P.3."
## [175] "RecNode80.P.0."   "RecNode80.P.1."   "RecNode80.P.2."
## [178] "RecNode80.P.3."   "RecNode81.P.0."   "RecNode81.P.1."
## [181] "RecNode81.P.2."   "RecNode81.P.3."   "RJRates...Mean"
## [184] "X"
```

```
for (i in 1:length(directories)){
  temp <- eval(parse(text=paste("run", i, "[,c(1,2,7:22)]", sep = "")))
```

```
  assign(paste("run", i, sep = ""), temp)
}
```

Make your mcmc objects and throw away the burnin

```
for (i in 1:length(directories)){
  temp <- eval(parse(text=paste("run", i,sep = "")))
  assign (paste ("mcmc",i ,sep = ""),
          mcmc(data = temp , start = start, end = end, thin = thin))
}
```

Put them into a list

```
mh.list <-mcmc.list()
for (i in 1:length(directories)){
      temp <- eval(parse(text=paste("mcmc", i, sep = "")))
      mh.list[[i]] <- temp
}
```

2) Diagnostics for a single run

a) Autocorrelation

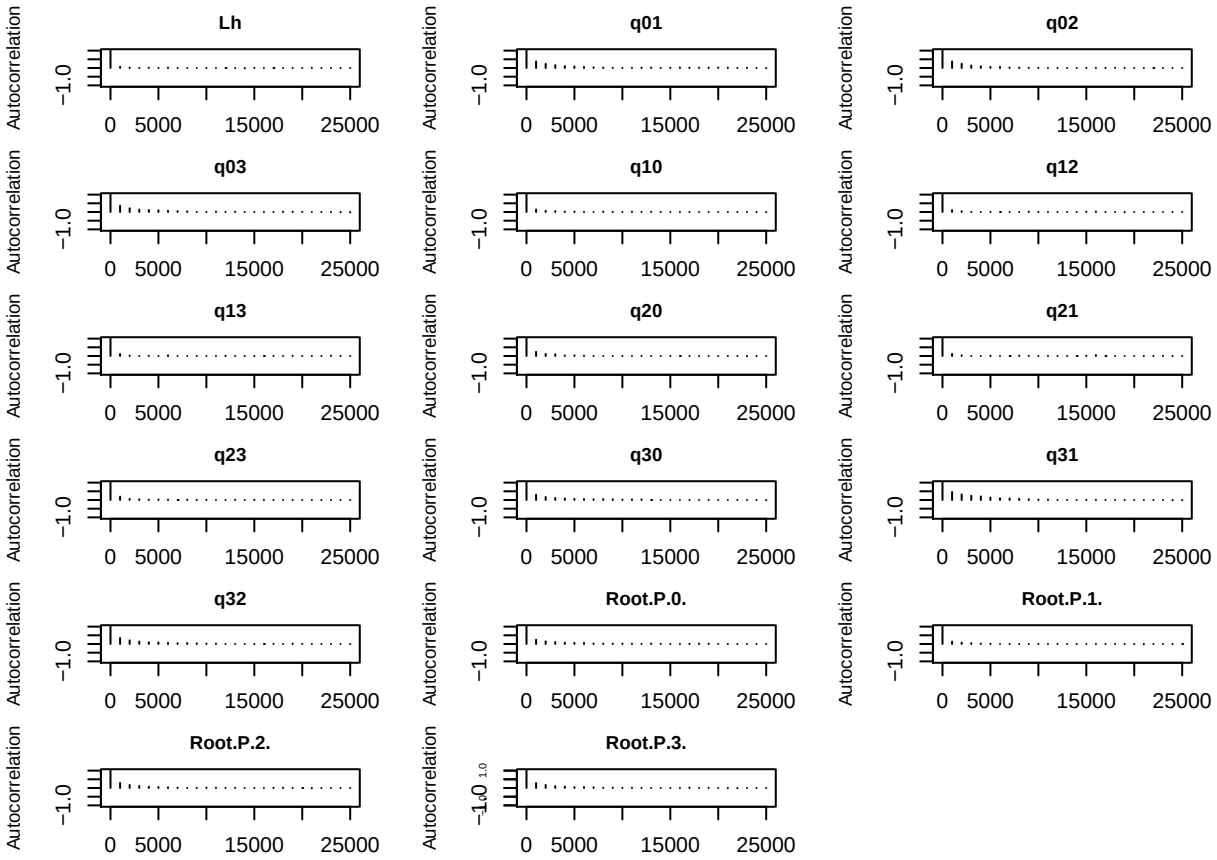
Calculate autocorrelation between draws

```
diag(autocorr(mcmc1)[2, , ])
```

```
##  Iteration         Lh        q01        q02        q03        q10        q12
## 0.99962500 0.06861575 0.38057845 0.37888107 0.35985231 0.14839570 0.11215309
##        q13        q20        q21        q23        q30        q31        q32
## 0.11671573 0.24293926 0.12119111 0.19631771 0.30339351 0.45913126 0.34788231
##  Root.P.0.  Root.P.1.  Root.P.2.  Root.P.3.
## 0.25213281 0.14174144 0.29067403 0.28218649
```

Present autocorrelation in plot form

```
par(mfrow=c(6,3), mar=c(2,4,2,2)+0.1, cex.main = 0.9, cex.lab = 0.9)
autocorr.plot(mcmc1[,-1], lag.max = 25, auto.layout = F)
axis(2,cex.axis=0.5)
```



b) Effective sample size

Calculate effective sample size

```
effectiveSize(mcmc1[,-1])
```

```
##         Lh        q01        q02        q03        q10        q12        q13        q20
##   6657.776   2408.116   2414.162   2448.909   4651.900   6002.864   5460.636   3811.765
##        q21        q23        q30        q31        q32 Root.P.0. Root.P.1. Root.P.2.
##   4712.132   4499.143   2490.259   1731.885   2345.895   2921.722   4973.877   3115.883
## Root.P.3.
##   2830.857
```

3) Test for convergence of multiple runs

a) Gelman-Rubin statistic

Calculate Gelman-Rubin diagnostic of convergence

```
gelman.diag(mh.list,confidence = 0.95, transform=TRUE, autoburnin=FALSE,
            multivariate=FALSE)
```

```
## Potential scale reduction factors:
##
```

```
##           Point est. Upper C.I.
## Iteration        NaN        NaN
## Lh              1.00       1.00
## q01             1.00       1.00
## q02             1.00       1.00
## q03             1.00       1.00
## q10             1.00       1.00
## q12             1.01       1.01
## q13             1.00       1.00
## q20             1.00       1.00
## q21             1.00       1.01
## q23             1.00       1.00
## q30             1.00       1.00
## q31             1.00       1.00
## q32             1.00       1.00
## Root.P.0.       1.00       1.00
## Root.P.1.       1.00       1.00
## Root.P.2.       1.00       1.00
## Root.P.3.       1.00       1.00
```

b) Plotting traces

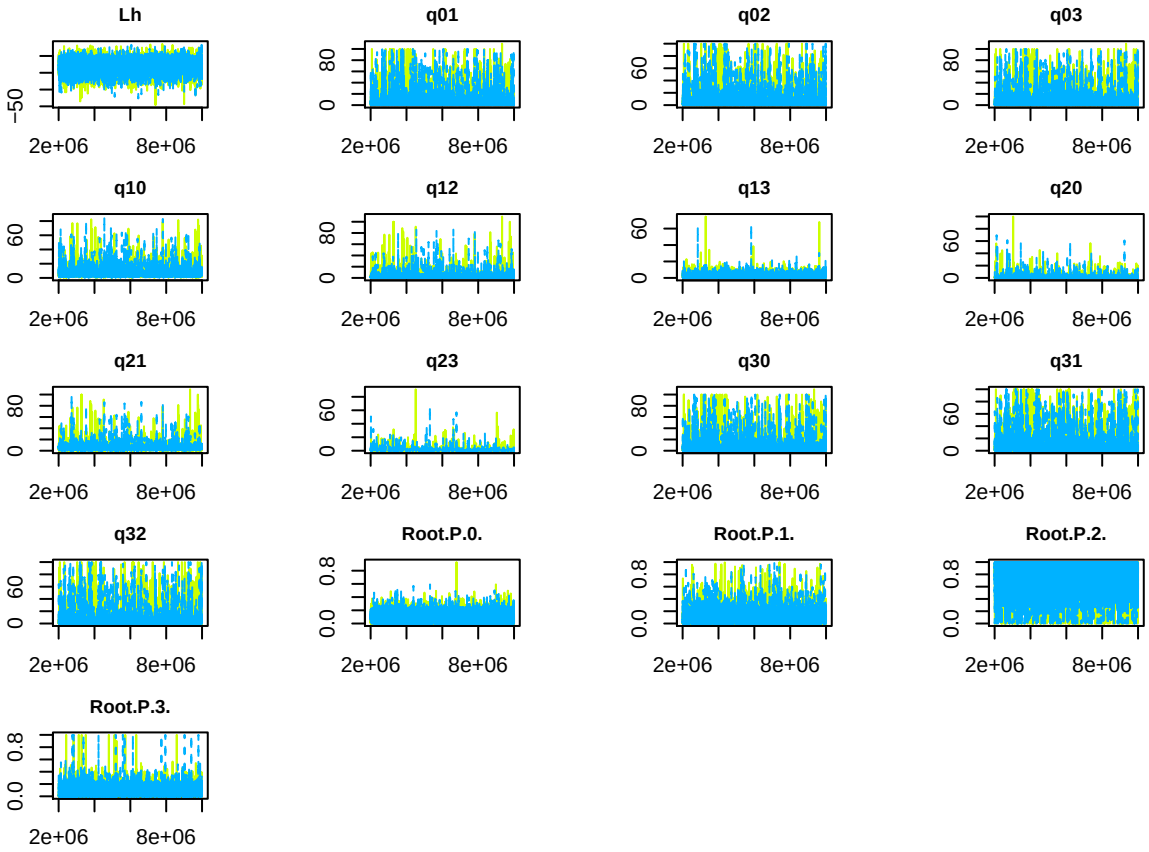
Get data into correct format for plotting traces from both runs simultaneously

```
for (i in 2:length(run1)){
  for (j in 1:length(directories)){
    temp <- eval(parse(text=paste("mcmc", j, "[,i]", sep = "")))
    assign(paste("trace_","mcmc", j, colnames(run1[i]), sep=""), temp)
  }
}

temp_list = mcmc.list()
for (i in 2:length(run1)){
  for (j in 1:length(directories)){
    temp_list[[j]] <-  eval(parse(text=paste("trace_mcmc", j , colnames(run1[i]),
                                             sep = "")))
    assign(paste("trace_" , colnames(run1[i]), sep = ""), temp_list)
 }
}
```

Plot traces

```
par(mfrow=c(5,4), mar = c(2,4,2,2)+0.1, cex.main = 0.9)
for (i in 2:length(run1)){
  temp <-  eval(parse(text = paste("trace_", colnames(run1[i]), sep = "")))
  traceplot(temp,col = rainbow(3, start=0.2, end=0.9))
  title(main = colnames(run1[i]), ylab = " ")
}
```



c) Plotting density curves

Get data into correct format for plotting density curves of both runs and throw out burnin

```
reshaping <- function(x) {
  y <- rbind(t(x[,-1]))
  result <- setNames(as.data.frame.table(y),c("variable","run","value"))
}

for (i in 1:length(directories)){
  temp <- eval(parse(text=paste("run", i,sep = "")))
  reshape <- reshaping(temp)
  reshape$run <- as.factor(rep(i, nrow(reshape)))
  assign(paste("reshape", i, sep = ""), reshape)
}

plot_df <- rbind(reshape1,reshape2)
```

Call the "ggplot2" package for plotting and "ggforce" for paginate function

```
require(ggplot2)
require(ggforce)
```

Plot by each parameter

```
p <- ggplot(plot_df, aes(x= value, fill = run, colour = run)) +
  geom_density(alpha = 0.1) +
  theme_bw(base_size = 12)+
  theme(legend.position = "none", strip.text.x = element_text(size = 6),
        axis.text.y = element_text(size=5), axis.text.x = element_text(size=5),
        panel.grid.major = element_blank(), panel.grid.minor = element_blank())+
  labs(x="Parameter value", y = "Density") +
  facet_wrap(~variable, scales = "free", ncol = 4)

p
```