FEL results preliminary

sadie

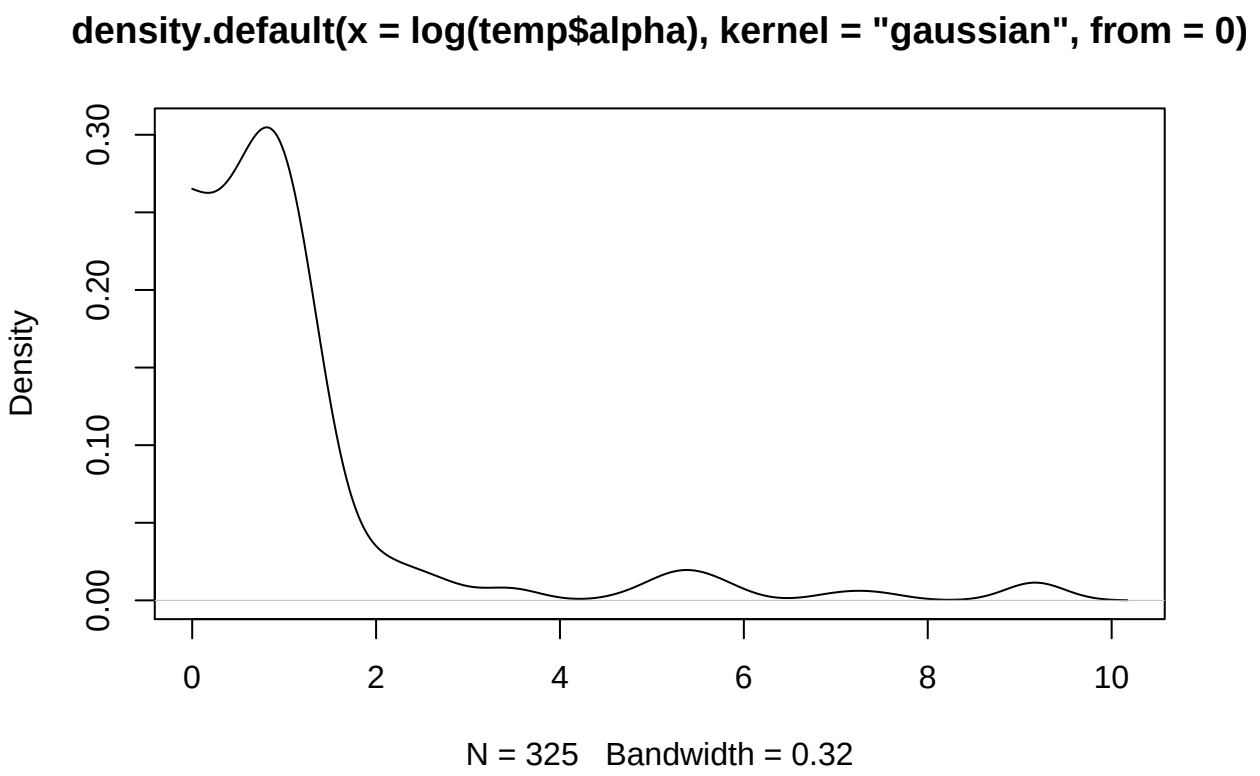
3/4/2020

```
filepath <- read_json("~/bin/mtDNA_redo/data/FEL/zeiformes-nd1-align-dna.fas.FEL.json") #read in json

heads <- filepath$MLE$headers  %>% unlist() %>% .[c(TRUE,FALSE)] #get headers and ignore header descrip
```

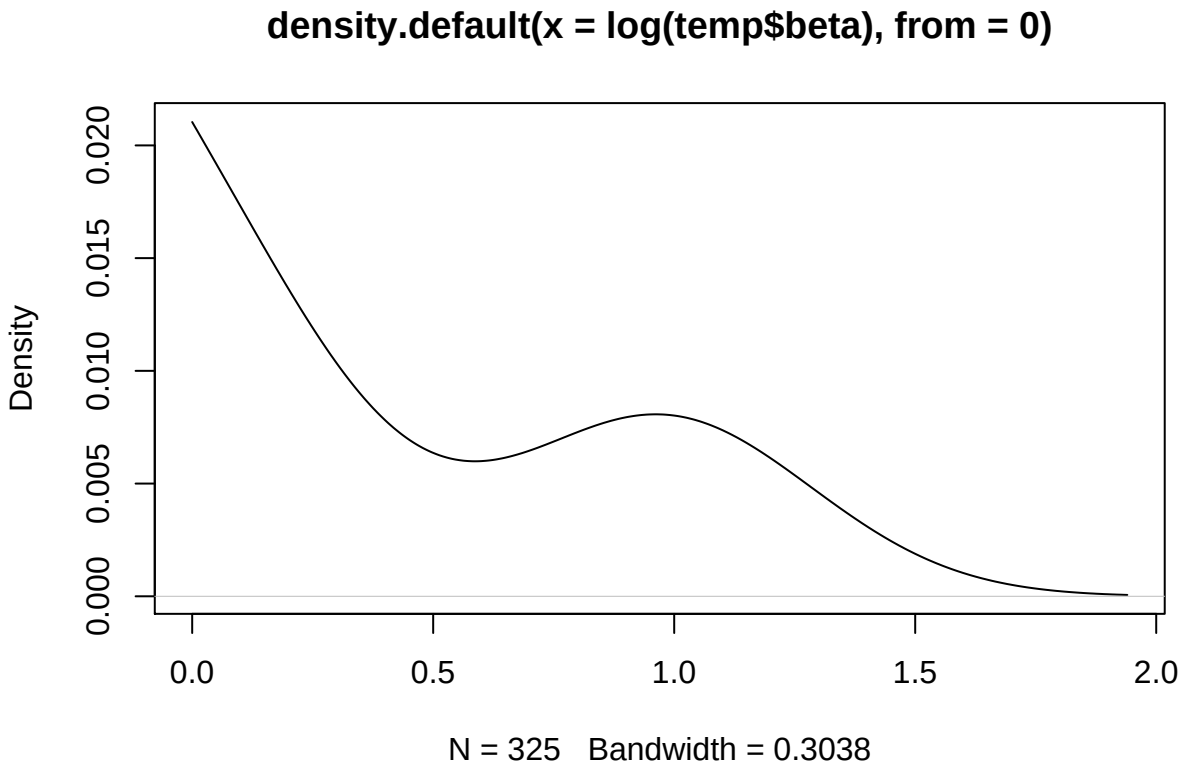
```
#get MLE contents and make them a data frame
temp <- filepath$MLE$content$`0` %>% unlist %>% matrix(ncol = 6, byrow = TRUE) %>% as.data.frame()
#make the headers the variable names
 names(temp)  <- heads
```

kernel density and plots?

```
d_alpha <- density(log(temp$alpha), kernel = "gaussian", from = 0)

d_alpha %>% plot()
```




```
d_beta <- density(log(temp$beta), from = 0)
d_beta %>% plot()
```



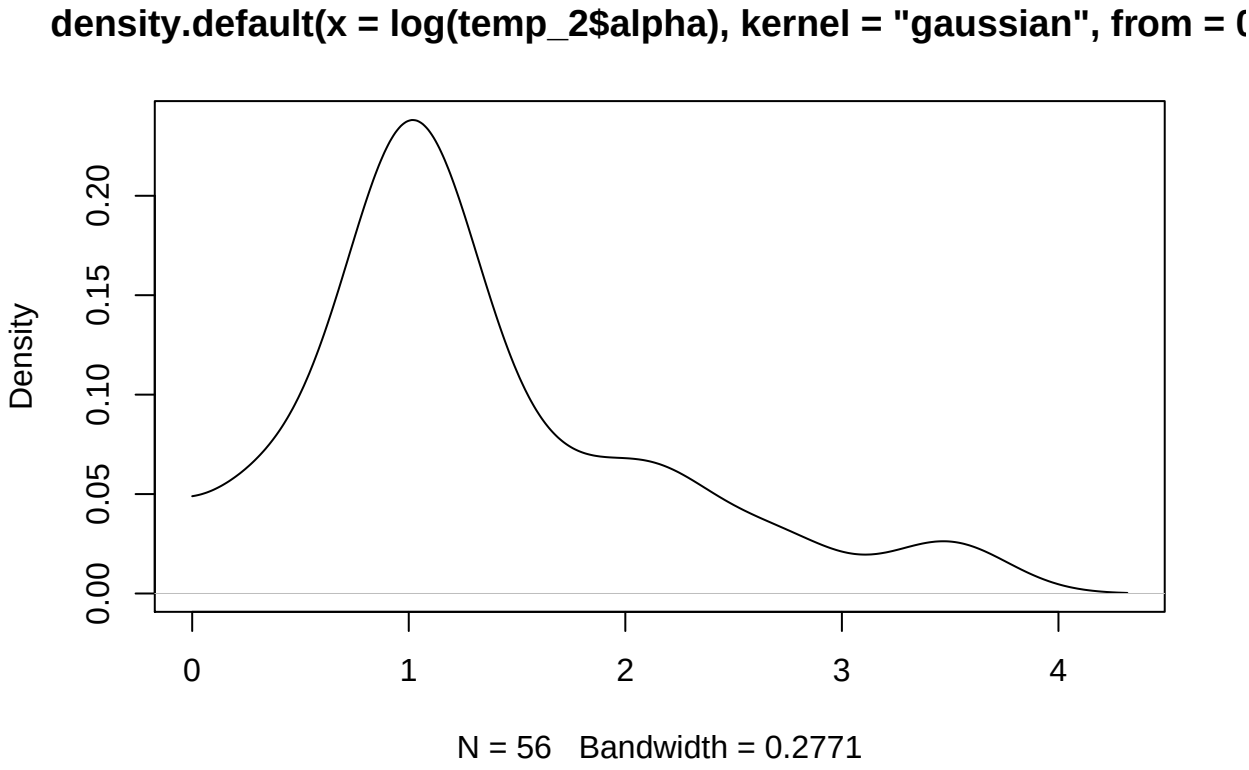
check a second alignment:

```
filepath <- read_json("~/bin/mtDNA_redo/data/FEL/acipenseriformes-atp8-align-dna.fas.FEL.json") #read i

heads <- filepath$MLE$headers  %>% unlist() %>% .[c(TRUE,FALSE)] #get headers and ignore header descrip
```

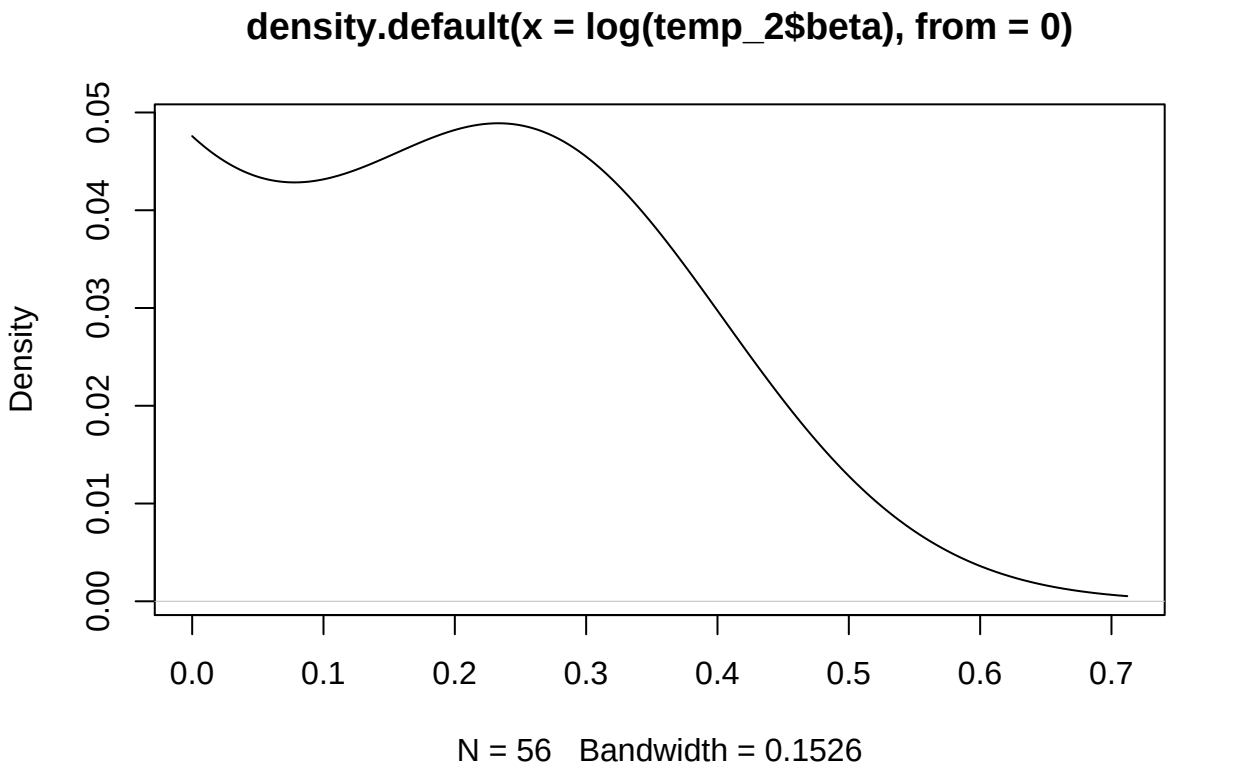
```
#get MLE contents and make them a data frame
temp_2 <- filepath$MLE$content$`0` %>% unlist %>% matrix(ncol = 6, byrow = TRUE) %>% as.data.frame()
#make the headers the variable names
 names(temp_2)  <- heads
```

kernel density and plots?

```
d_alpha_2 <- density(log(temp_2$alpha), kernel = "gaussian", from = 0)

d_alpha_2 %>% plot()
```




```
d_beta_2 <- density(log(temp_2$beta), from = 0)
d_beta_2 %>% plot()
```




```
#uses entropy library to calculate Kullback-Leibler divergence (KL) as it is needed to do the JSD
library("entropy")
```

```
KL <- KL.plugin(freqs1 = d_alpha$x, freqs2 = d_beta$x)
```

```
## Warning in KL.plugin(freqs1 = d_alpha$x, freqs2 = d_beta$x): Vanishing value(s)
## in argument freqs2!
```

```
KL_2 <- KL.plugin(freqs1 = d_alpha_2$x, freqs2 = d_beta_2$x)
```

```
## Warning in KL.plugin(freqs1 = d_alpha_2$x, freqs2 = d_beta_2$x): Vanishing
## value(s) in argument freqs2!
```

```
#from slackoverflow https://stackoverflow.com/questions/11226627/jensen-shannon-divergence-in-r
p <- d_beta_2$x
q <- d_alpha_2$x
n <- 0.5 * (p + q)
JS <- 0.5 * (sum(p * log(p / n)) + sum(q * log(q / n)))
```

```
#actually, lets try the way this site does it: https://enterotype.embl.de/enterotypes.html

JSD<- function(x,y) sqrt(0.5 * KLD(x, (x+y)/2) + 0.5 * KLD(y, (x+y)/2))

KLD <- function(x,y) sum(x * log(x/y))

KLD(d_beta_2$x, d_alpha_2$x)
```

```
## [1] NaN
```

```
JSD(d_beta_2$x, d_alpha_2$x)
```

```
## [1] NaN
```

```
 dist.JSD <- function(inMatrix, pseudocount=0.000001, ...) {
    KLD <- function(x,y) sum(x *log(x/y))
    JSD<- function(x,y) sqrt(0.5 * KLD(x, (x+y)/2) + 0.5 * KLD(y, (x+y)/2))
    matrixColSize <- length(colnames(inMatrix))
    matrixRowSize <- length(rownames(inMatrix))
    colnames <- colnames(inMatrix)
    resultsMatrix <- matrix(0, matrixColSize, matrixColSize)

  inMatrix = apply(inMatrix,1:2,function(x) ifelse (x==0,pseudocount,x))

    for(i in 1:matrixColSize) {
        for(j in 1:matrixColSize) {
            resultsMatrix[i,j]=JSD(as.vector(inMatrix[,i]),
            as.vector(inMatrix[,j]))
        }
    }
    colnames -> colnames(resultsMatrix) -> rownames(resultsMatrix)
    as.dist(resultsMatrix)->resultsMatrix
    attr(resultsMatrix, "method") <- "dist"
    return(resultsMatrix)
 }
```

calculate the JSD between alpha and beta

```
d.temp <- dist.JSD(temp %>% select(alpha, beta))

d.temp
```

```
##          alpha
## beta 108.0882
```

this gives me th JSD between the nonsynonymous (beta) and synonymous (alpha) rate distributions.