Applying Gaussian Mixture Model

Georgina Shaw

07/03/2022

Herrin

Only keeping the information we need

```
df <- stom_df %>%
  transmute(Species = pred_species,
            wprey = prey_weight_g,
            wpredator = pred_weight_g,
            Nprey = prey_count / n_stomachs,
            l = log(wpredator / wprey))
```

Choosing the type of fish to use, this time looking at cod

```
stomach <- df %>%
  filter(Species == "Clupea harengus",
         wprey > 0)

stomach %>%
  group_by(Species) %>%
  summarise(wprey_min = min(wprey),
            wprey_max = max(wprey),
            lmin = min(l),
            lmax = max(l))
```

```
## # A tibble: 1 x 5
##   Species         wprey_min wprey_max   lmin  lmax
##   <chr>               <dbl>     <dbl>  <dbl> <dbl>
## 1 Clupea harengus   0.00001      104. 0.0544  17.5
```

```
stomach %>%
  group_by(Species) %>%
  filter(wprey == max(wprey))
```

```
## # A tibble: 1 x 5
## # Groups:   Species [1]
##   Species         wprey wpredator Nprey     l
##   <chr>           <dbl>     <dbl> <dbl> <dbl>
## 1 Clupea harengus  104.      116.  35.2 0.111
```

Creating bins for the data

```
no_bins <- 30  # Number of bins
binsize <- (max(stomach$l) - min(stomach$l)) / (no_bins - 1)
breaks <- seq(min(stomach$l) - binsize/2,
              by = binsize, length.out = no_bins + 1)
```

Splitting the data into the bins that have been made

```
binned_stomach <- stomach %>%
  # bin data
  mutate(cut = cut(l, breaks = breaks, right = FALSE,
                   labels = FALSE)) %>%
  group_by(Species, cut) %>%
  summarise(Numbers = sum(Nprey),
            Biomass = sum(Nprey * wprey)) %>%
  # normalise
  mutate(Numbers = Numbers / sum(Numbers) / binsize,
         Biomass = Biomass / sum(Biomass) / binsize)  %>%
  # column for predator/prey size ratio
  mutate(l = map_dbl(cut, function(idx) breaks[idx] + binsize/2))
```

```
## `summarise()` has grouped output by 'Species'. You can override using the `.groups` argument.
```

```
binned_stomach
```

```
## # A tibble: 30 x 5
## # Groups:   Species [1]
##    Species           cut Numbers Biomass      l
##    <chr>           <int>   <dbl>   <dbl>  <dbl>
##  1 Clupea harengus     1 0.00110  0.0351 0.0544
##  2 Clupea harengus     2 0.0197   0.348  0.655
##  3 Clupea harengus     3 0.0362   0.503  1.26
##  4 Clupea harengus     4 0.0264   0.162  1.86
##  5 Clupea harengus     5 0.0197   0.0546 2.46
##  6 Clupea harengus     6 0.0230   0.0575 3.06
##  7 Clupea harengus     7 0.0254   0.0717 3.66
##  8 Clupea harengus     8 0.0618   0.0663 4.26
##  9 Clupea harengus     9 0.115    0.113  4.86
## 10 Clupea harengus    10 0.145    0.0913 5.46
## # ... with 20 more rows
```

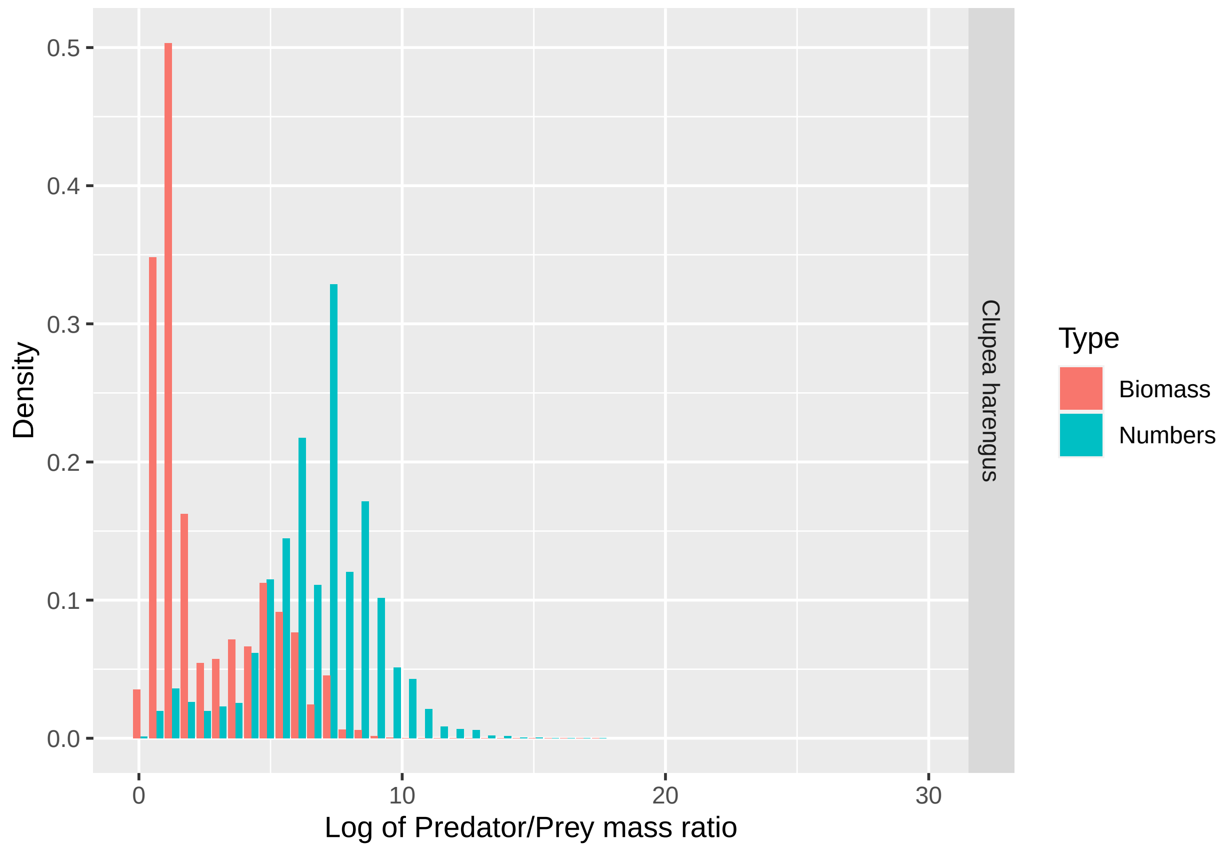
We convert this into the long table format preferred by ggplot2.

```
binned_stomach <- binned_stomach %>%
  gather(key = "Type", value = "Density", Numbers, Biomass)
```

Histograms

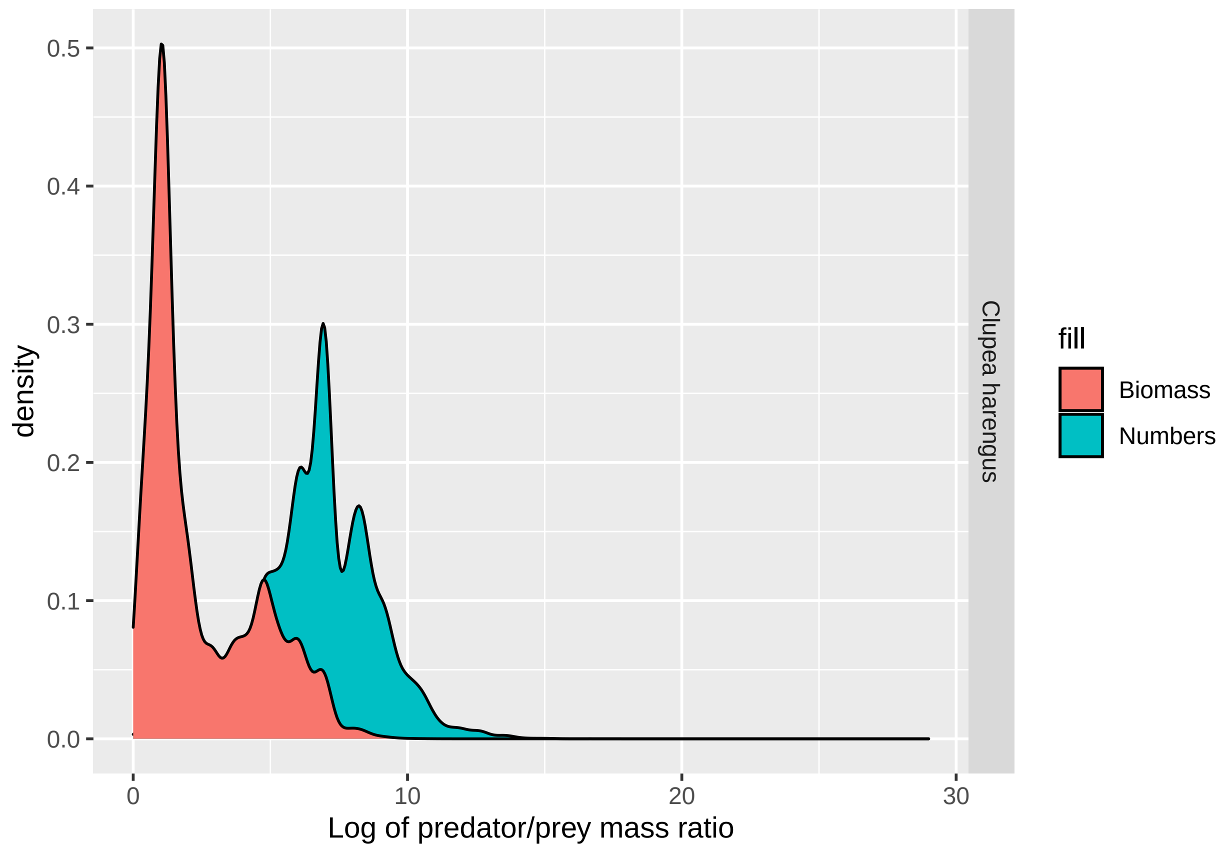
Plot the histogram that represents estimates of the normalised number density and the normalised biomass density

```
binned_stomach %>%
  ggplot(aes(l, Density, fill = Type)) +
  geom_col(position = "dodge") +
  facet_grid(rows = vars(Species), scales = "free_y") +
  xlab("Log of Predator/Prey mass ratio") +
  expand_limits(x = c(0, 30))
```



Kernal Density Estimation

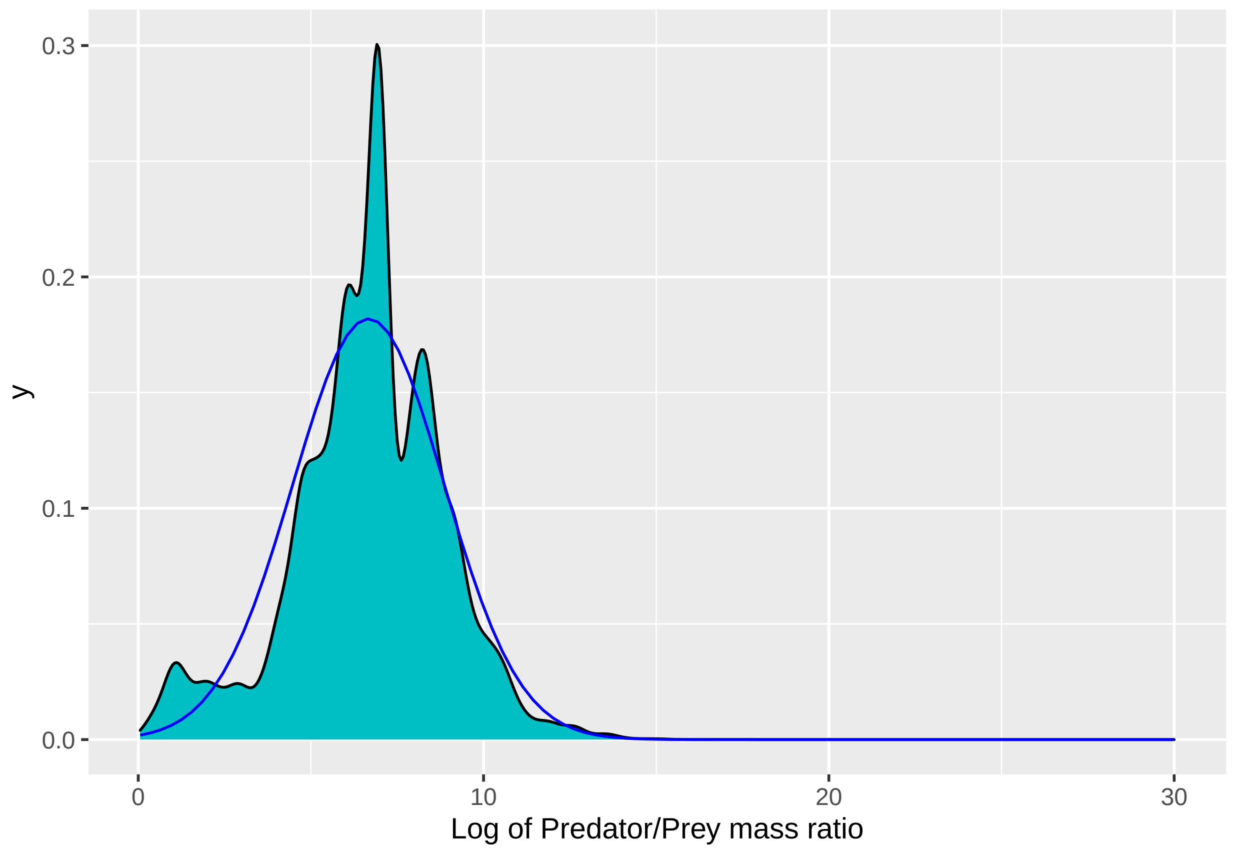
```
adjust <- 1/2  # decrease bandwidth for kernel estimate
stomach <- stomach %>%
  group_by(Species) %>%
  mutate(weight_numbers = Nprey / sum(Nprey),
         weight_biomass = Nprey * wprey / sum(Nprey * wprey))
ggplot(stomach) +
  geom_density(aes(l, weight = weight_numbers,
                   fill = "Numbers"),
               adjust = adjust) +
  geom_density(aes(l, weight = weight_biomass,
                   fill = "Biomass"),
               adjust = adjust) +
  facet_grid(rows = vars(Species), scales = "free_y") +
  xlab("Log of predator/prey mass ratio") +
  expand_limits(x = c(0, 29))
```



Gaussian Distribution fit

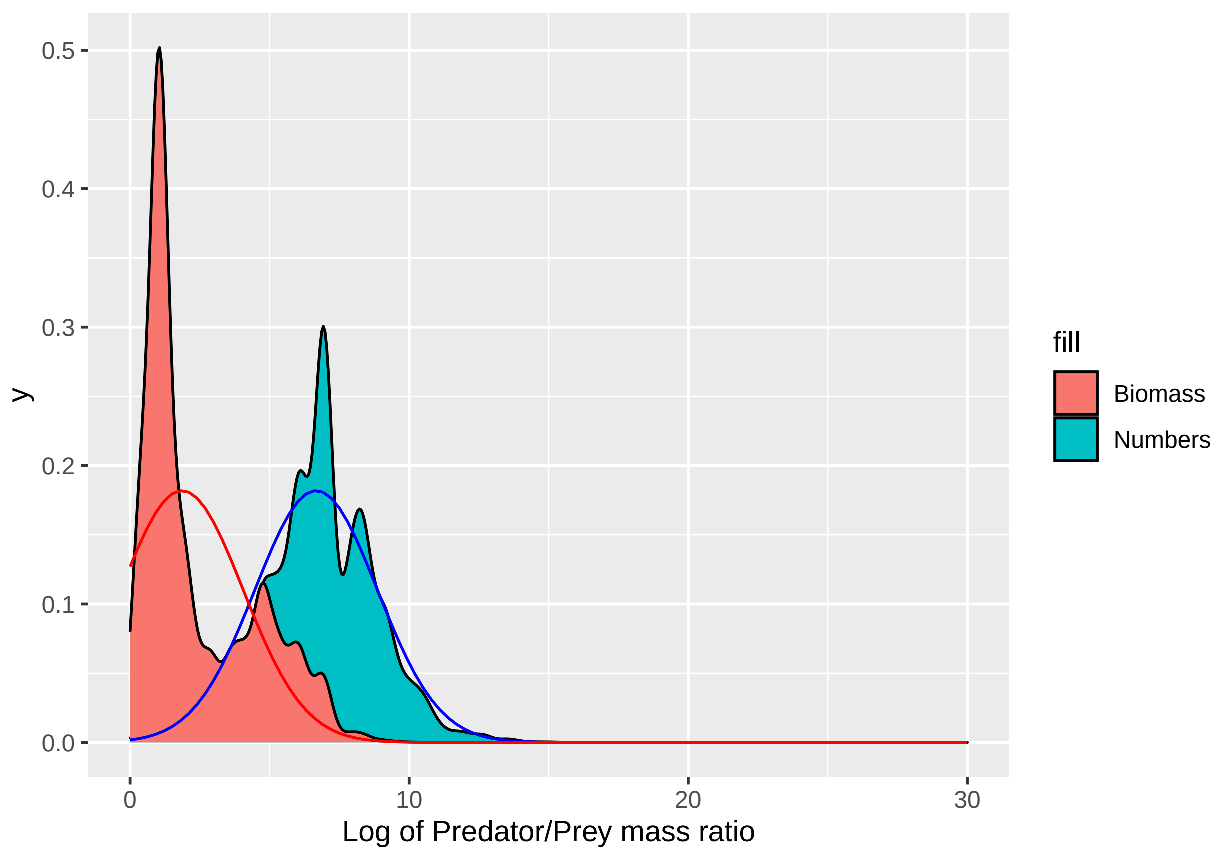
Plotting the normal distribution for numbers

```
weighted.sd <- function(x, w) {
  sqrt(sum(w * (x - weighted.mean(x, w))^2))
}
fit <- stomach %>%
  summarise(mean = weighted.mean(l, weight_numbers),
            sd = weighted.sd(l, weight_numbers))
stomach %>%
  ggplot() +
  geom_density(aes(l, weight = weight_numbers),
               fill = "#00BFC4", adjust = adjust) +
  xlab("Log of Predator/Prey mass ratio") +
  stat_function(fun = dnorm,
                args = list(mean = fit$mean,
                            sd = fit$sd),
                colour = "blue") +
  expand_limits(x = c(6, 30))
```

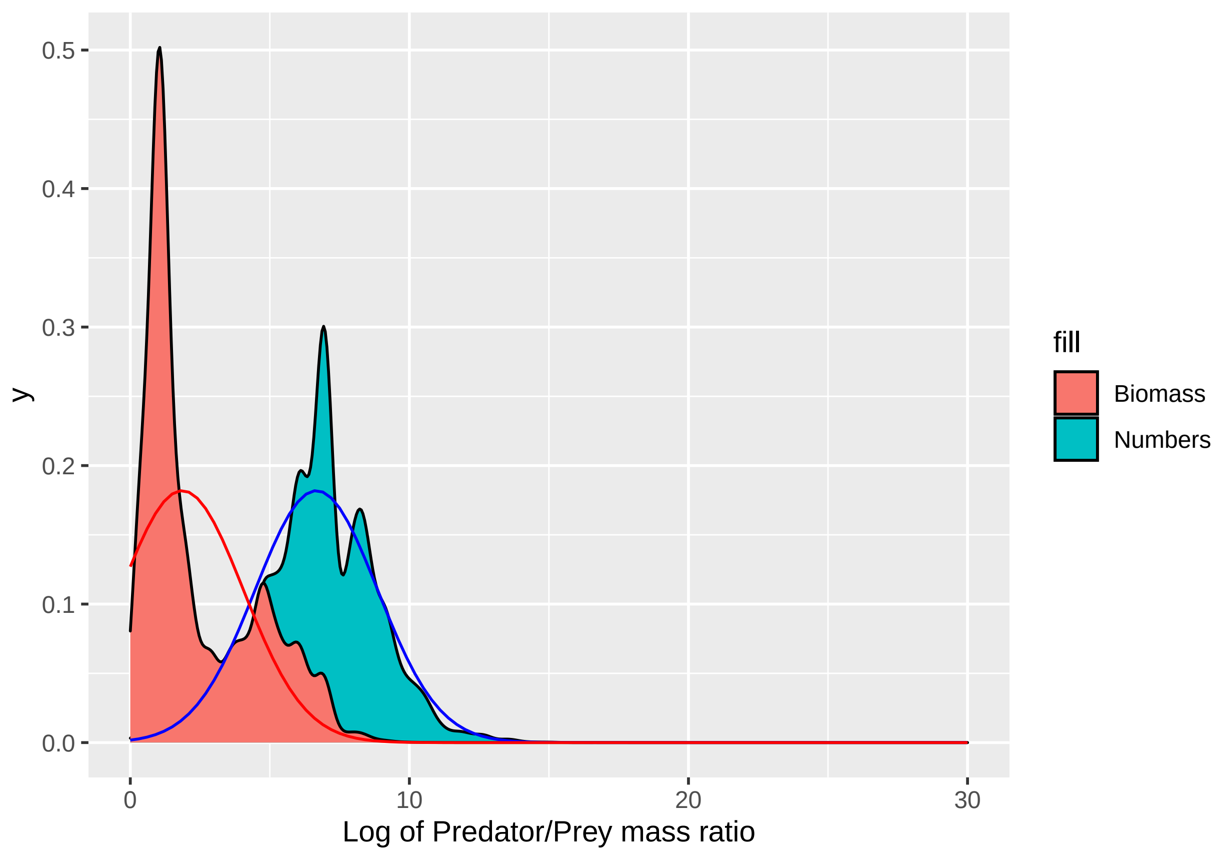


Now adding biomass

```
stomach %>%
  ggplot() +
  geom_density(aes(l, weight = weight_numbers,
                   fill = "Numbers"),
               adjust = adjust) +
  geom_density(aes(l, weight = weight_biomass,
                   fill = "Biomass"),
               adjust = adjust) +
  xlab("Log of Predator/Prey mass ratio") +
  stat_function(fun = dnorm,
                args = list(mean = fit$mean,
                            sd = fit$sd),
                colour = "blue") +
  stat_function(fun = dnorm,
                args = list(mean = fit$mean - fit$sd^2,
                            sd = fit$sd),
                colour = "red") +
  expand_limits(x = c(0, 30))
```



```
weighted.sd <- function(x, w) {
  sqrt(sum(w * (x - weighted.mean(x, w))^2))
}
fit <- stomach %>%
  summarise(mean = weighted.mean(l, weight_numbers),
            sd = weighted.sd(l, weight_numbers))
stomach %>%
  ggplot() +
  geom_density(aes(l, weight = weight_numbers,
                   fill = "Numbers"),
               adjust = adjust) +
  geom_density(aes(l, weight = weight_biomass,
                   fill = "Biomass"),
               adjust = adjust) +
  xlab("Log of Predator/Prey mass ratio") +
  stat_function(fun = dnorm,
                args = list(mean = fit$mean,
                            sd = fit$sd),
                colour = "blue") +
  stat_function(fun = dnorm,
                args = list(mean = fit$mean - fit$sd^2,
                            sd = fit$sd),
                colour = "red") +
  expand_limits(x = c(0, 30))
```



Applying Gaussian Mixture Models in different ways First way by calculating each individual PPMR by numbers and biomass and then applying the EM algorithm to find the estimated parameters and then plotting

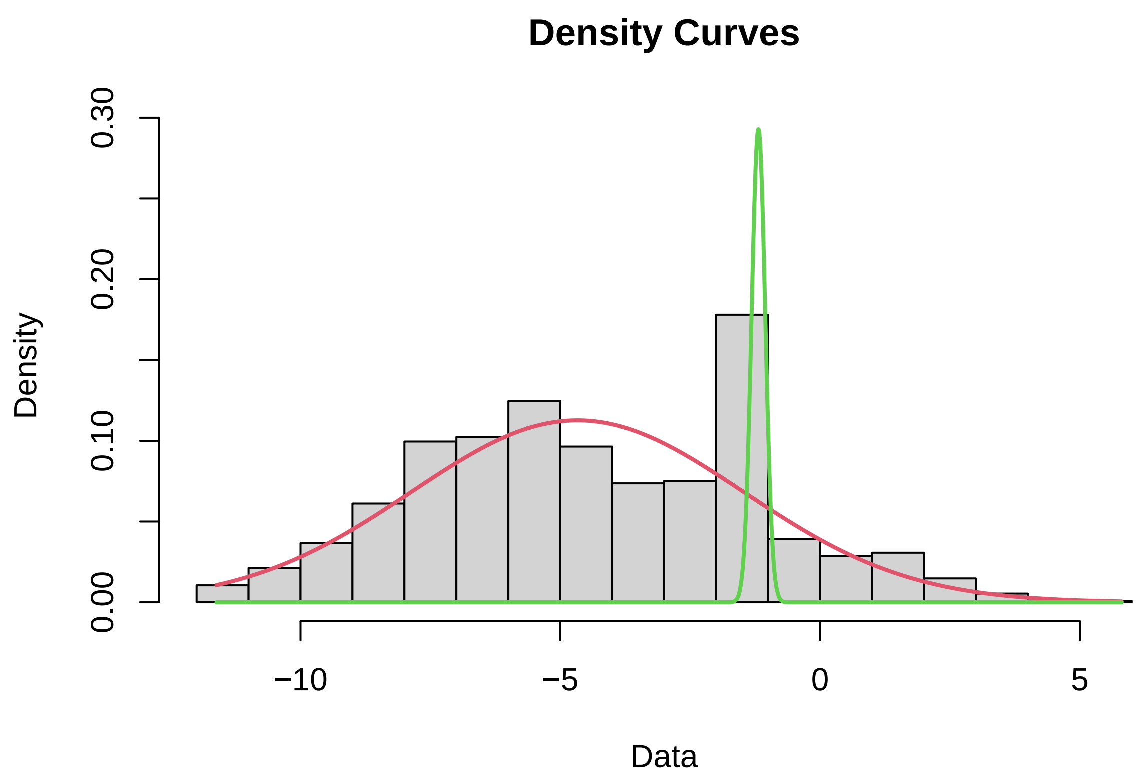
```
ppmr <- stomach %>%
  mutate(numbers = log(wpredator/(wprey * sum(Nprey))),
         biomass = log(wpredator*sum(Nprey)/wprey))
library(mixtools)
```

```
## mixtools package, version 1.2.0, Released 2020-02-05
## This package is based upon work supported by the National Science Foundation under Grant No. SES-051
```

```
my_mix1 <- normalmixEM(ppmr$numbers, k = 2)
```

```
## number of iterations= 54
```

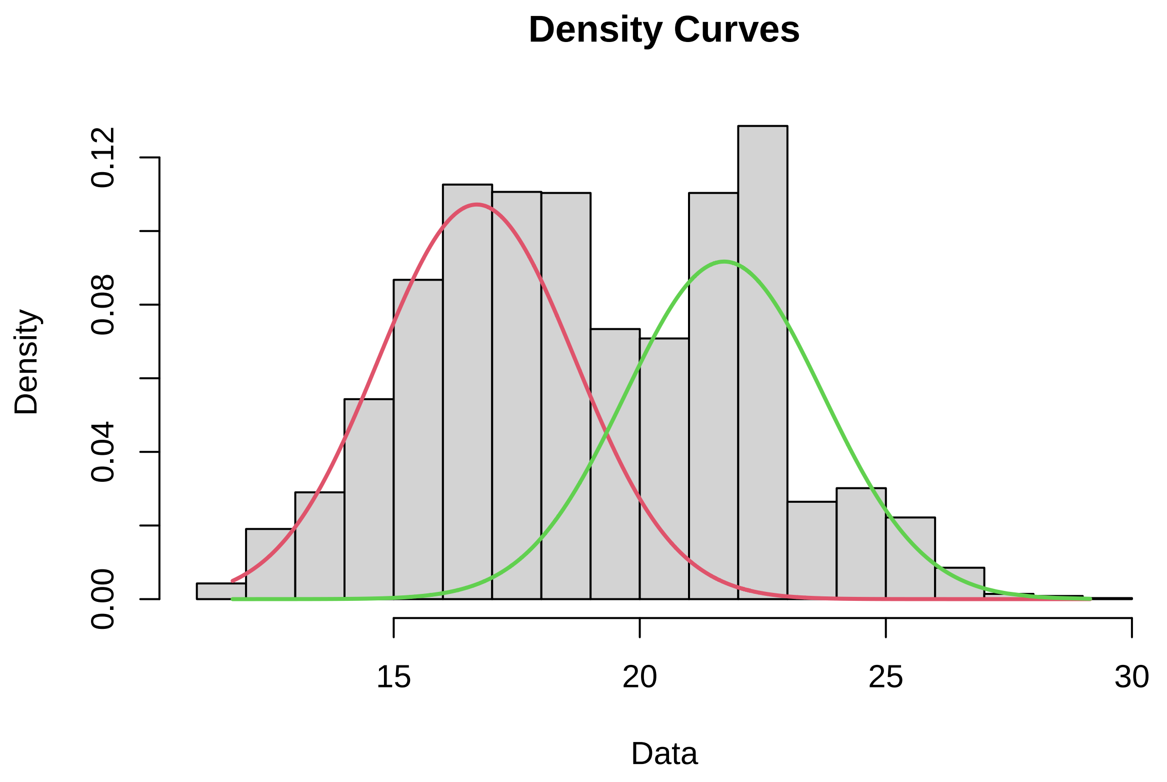
```r
plot(my_mix1, which = 2)
```



```r
my_mix2 <- normalmixEM(ppmr$biomass, k = 2)
```

```
## number of iterations= 202
```

```r
plot(my_mix2, which = 2)
```



```
my_mix1$lambda
```

```
## [1] 0.90316981 0.09683019
```

```
my_mix1$mu
```

```
## [1] -4.669121 -1.185401
```

```
my_mix1$sigma
```

```
## [1] 3.1984786 0.1319283
```

```
my_mix2$lambda
```

```
## [1] 0.5378012 0.4621988
```

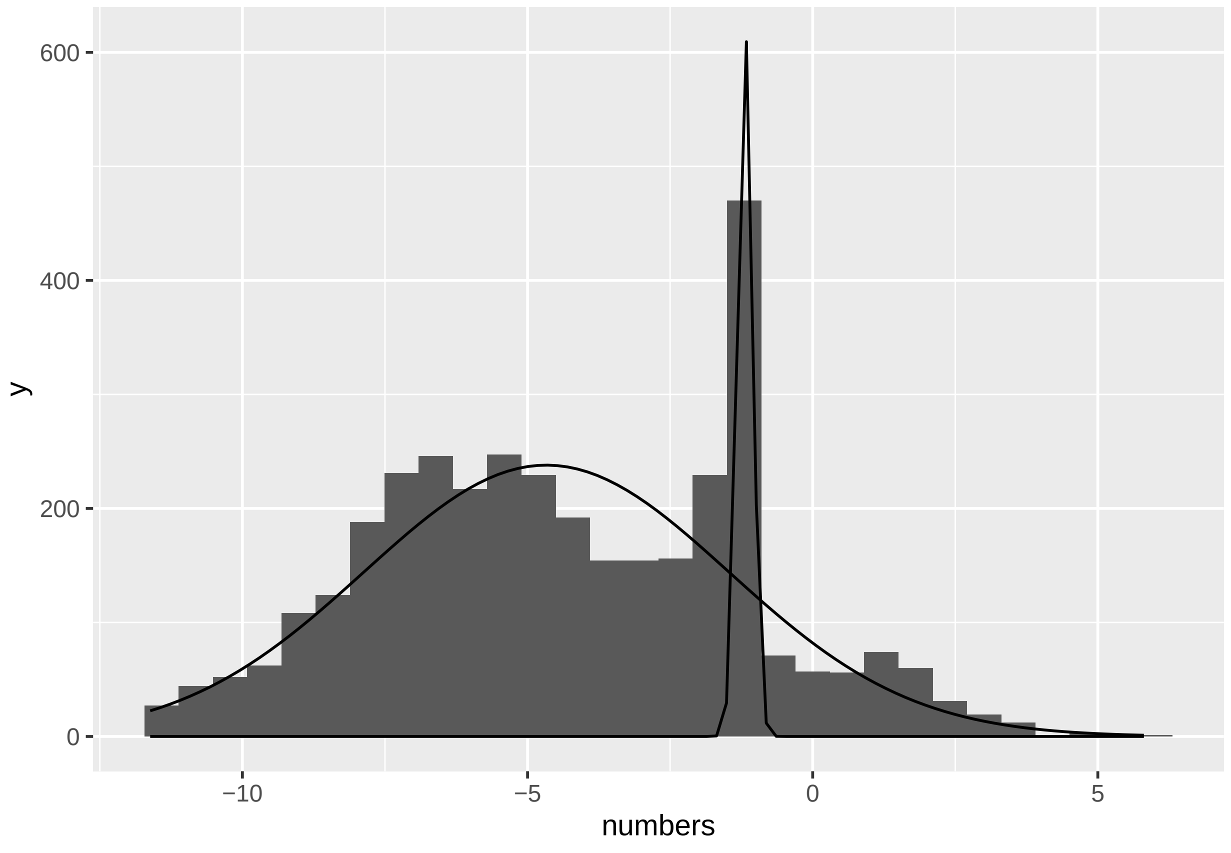
```
my_mix2$mu
```

```
## [1] 16.68859 21.71483
```

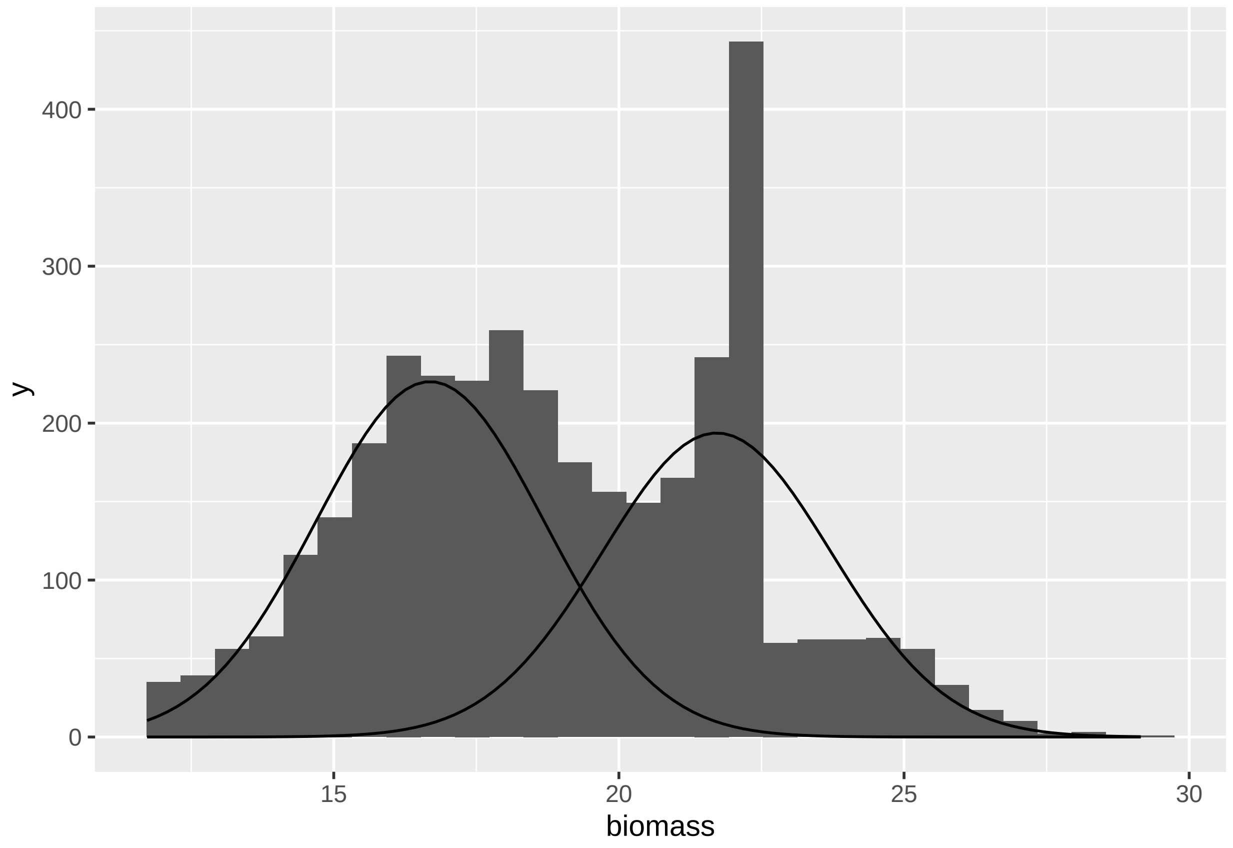
```r
my_mix2$sigma
```

```
## [1] 2.001409 2.010790
```

```r
#plot of numbers
ggplot(ppmr, aes(x = numbers)) +
  geom_histogram(binwidth = binsize) +
  mapply(
    function(mean, sd, lambda, n, binwidth) {
      stat_function(
        fun = function(x) {
          (dnorm(x, mean = mean, sd = sd)) * n * binwidth * lambda
        }
      )
    },
    mean = my_mix1[["mu"]], #mean
    sd = my_mix1[["sigma"]], #standard deviation
    lambda = my_mix1[["lambda"]], #amplitude
    n = length(ppmr$numbers), #sample size
    binwidth = binsize #binwidth used for histogram
  )
```



```r
#plot of biomass
ggplot(ppmr, aes(x = biomass)) +
  geom_histogram(binwidth = binsize) +
  mapply(
    function(mean, sd, lambda, n, binwidth) {
      stat_function(
        fun = function(x) {
          (dnorm(x, mean = mean, sd = sd)) * n * binwidth * lambda
        }
      )
    },
    mean = my_mix2[["mu"]], #mean
    sd = my_mix2[["sigma"]], #standard deviation
    lambda = my_mix2[["lambda"]], #amplitude
    n = length(ppmr$biomass), #sample size
    binwidth = binsize #binwidth used for histogram
  )
```

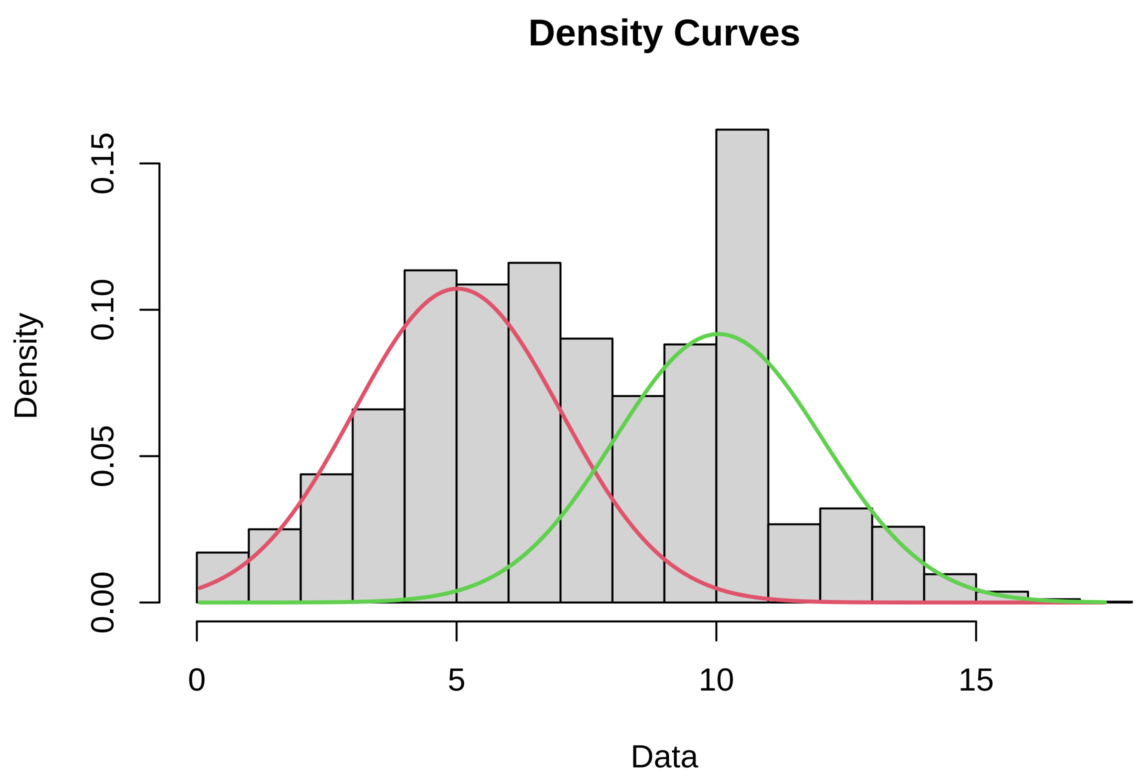


Second Method - not sure if this is correct Estimating the parameters for log PPMR and adjusting these for numbers and biomass, depending on the shift they have on the orginal plot of the log of the PPMR

```
my_mix <- normalmixEM(stomach$l, k = 2)
```

```
## number of iterations= 170
```

```r
plot(my_mix, which = 2)
```



```r
my_mix$lambda
```

```
## [1] 0.5377801 0.4622199
```

```r
my_mix$mu
```

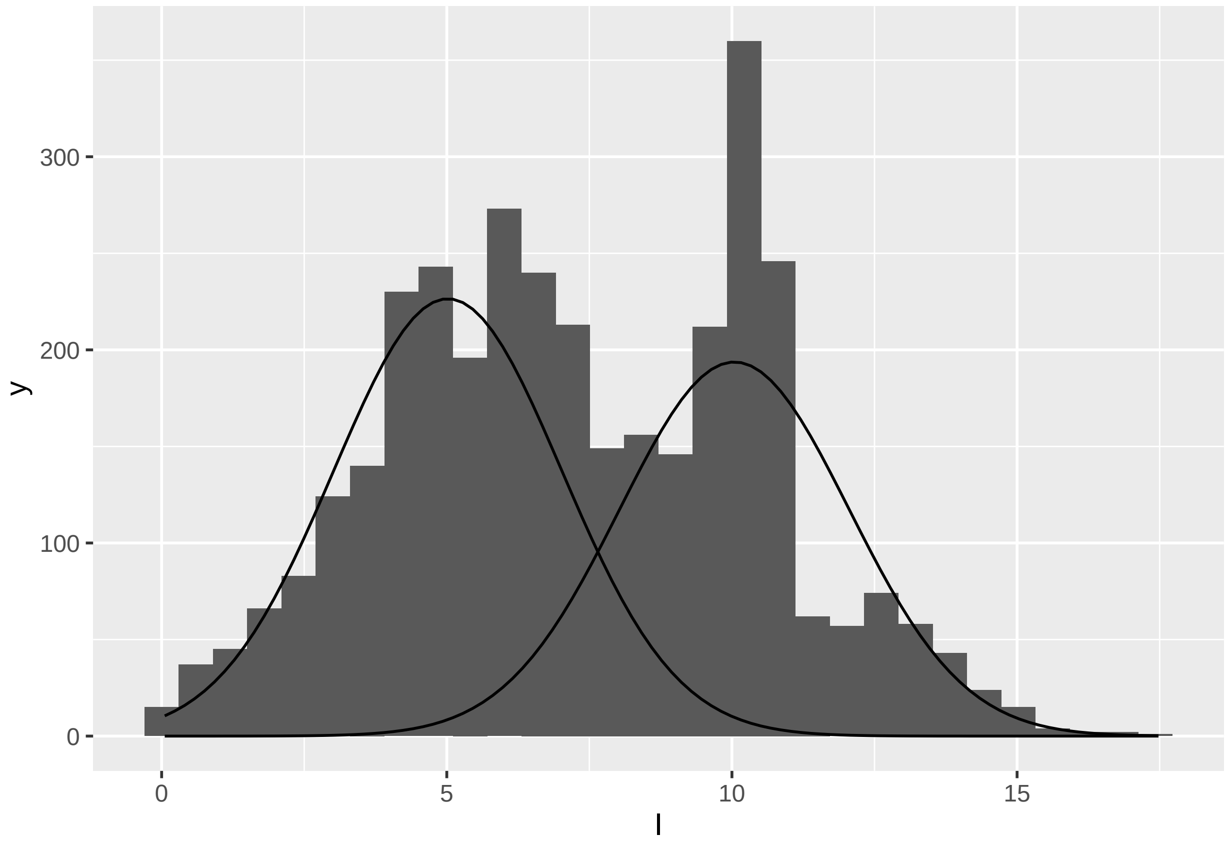
```
## [1]  5.016739 10.042959
```

```r
my_mix$sigma
```

```
## [1] 2.001367 2.010852
```

```r
#plot of the log PPMR with the estimated parameters

ggplot(stomach, aes(x = l)) +
  geom_histogram(binwidth = binsize) +
  mapply(
    function(mean, sd, lambda, n, binwidth) {
      stat_function(
        fun = function(x) {
          (dnorm(x, mean = mean, sd = sd)) * n * binwidth * lambda
        }
      )
    },
    mean = my_mix[["mu"]], #mean
    sd = my_mix[["sigma"]], #standard deviation
    lambda = my_mix[["lambda"]], #amplitude
    n = length(stomach$l), #sample size
    binwidth = binsize #binwidth used for histogram
  )
```



```
#how found the mean and sd of l and the difference in weightings
fit <- stomach %>%
  summarise(mean = weighted.mean(l, weight_numbers),
            sd = weighted.sd(l, weight_numbers))
mean(stomach$l)
```

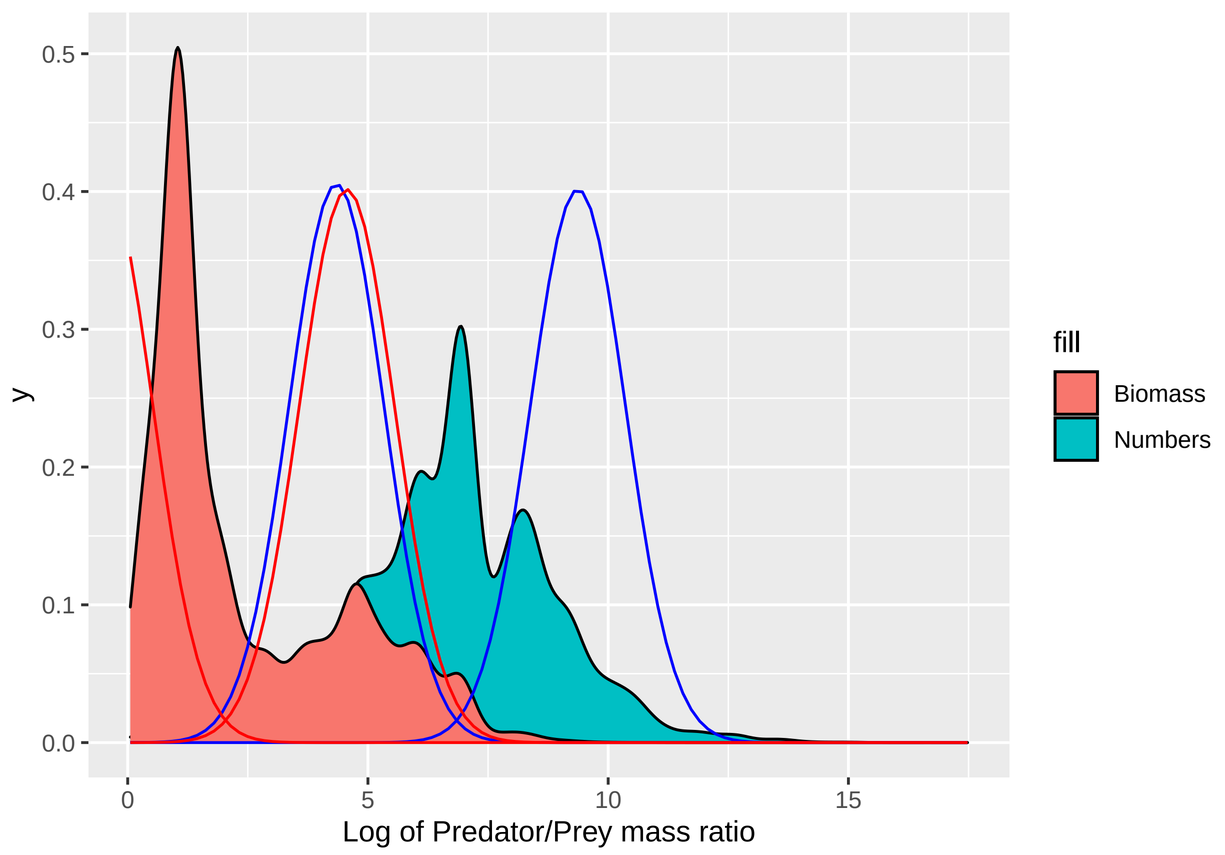
```
## [1] 7.339959
```

```
sd(stomach$l)
```

```
## [1] 3.210242
```

```r
#then figured out the shift that the weighting of numbers and biomass had on the mean and standard devi
#used this difference to then add or subtract the difference from the new estimated parameters
#plotted these on the original plot of the log of PPMR with the weighting of numbers and biomass

stomach %>%
  ggplot() +
  geom_density(aes(l,weight = weight_numbers,
                   fill = "Numbers"),
               adjust = adjust) +
  geom_density(aes(l, weight = weight_biomass,
                   fill = "Biomass"),
               adjust = adjust) +
  xlab("Log of Predator/Prey mass ratio") +
  stat_function(fun = dnorm,
                args = list(mean = 4.344325,
                            sd = 0.984058),
                colour = "blue") +
  stat_function(fun = dnorm,
                args = list(mean = 9.370544,
                            sd = 0.993544),
                colour = "blue") +
  stat_function(fun = dnorm,
                args = list(mean = -0.464636,
                            sd = 0.984058),
                colour = "red") +
  stat_function(fun = dnorm,
                args = list(mean = 4.561583,
                            sd = 0.993544),
                colour = "red")
```



0.5
0.4
0.3
0.2
0.1
0.0
y
0
5
10
15
Log of Predator/Prey mass ratio
fill
Biomass
Numbers