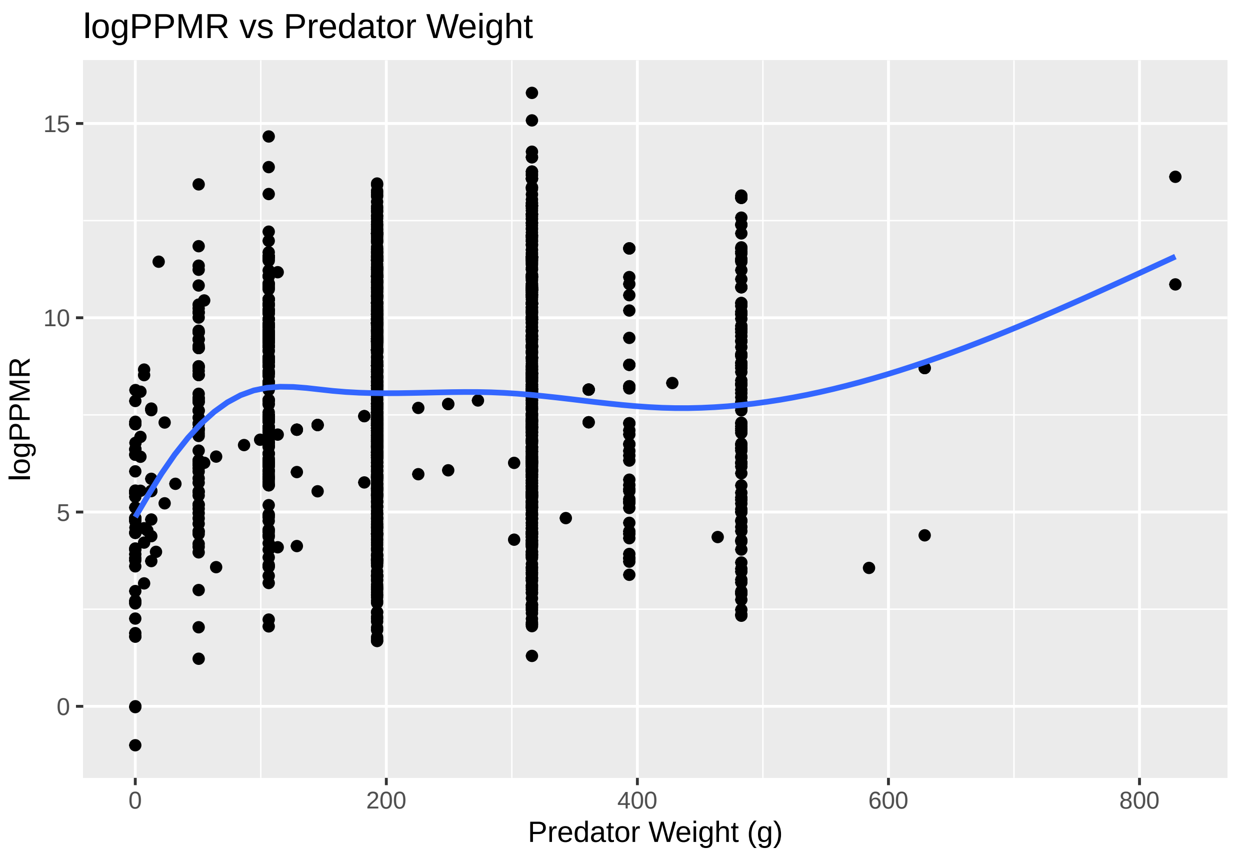
# Stomach Contents - Horse Mackerel

2024-06-28

This file will go through the process of generating the selection function for Horse Mackerel. Firstly, I need to check that it PPMR doesnt change

```r
library(dplyr)
library(ggplot2)
library(tidyr)
library(bbmle)
#reading in the data
load("C:/Users/lucab/Downloads/stomach_dataset.Rdata")

mack <- stom_df%>%filter(pred_species=="Trachurus trachurus")

mack <- mack%>%select(prey_weight_g, pred_weight_g, nprey_perpred, pred_species,)%>%
rename(wprey=prey_weight_g, wpredator=pred_weight_g, Nprey=nprey_perpred, Species=pred_species)%>%
  mutate(wprey=wprey/Nprey)

#ppmr
mack <- mack%>%mutate(ppmr=(wpredator/wprey))

#plotting the ppmr
ggplot(mack, aes(x=wpredator, y=log(ppmr)))+
  geom_point()+
  geom_smooth(method="gam", se=FALSE)+
  labs(title="logPPMR vs Predator Weight", x="Predator Weight (g)", y="logPPMR")
```

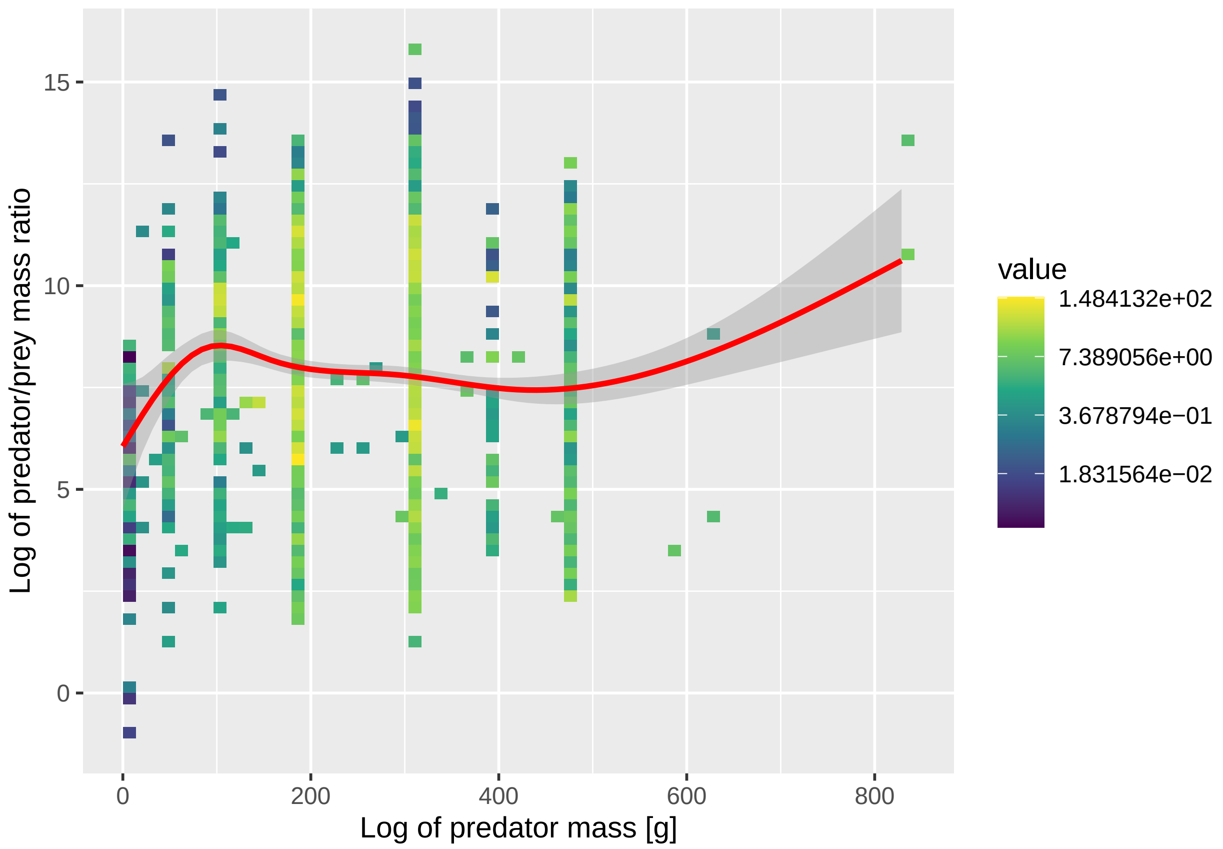


This doesnt look the best, but it seems to be only the smallest sizes, and the largest that have varying PPMR, it may be that at the largest sizes the only possible prey are outside the normal range. So at the largest sizes of mackerel, their usual prey species do not grow to the same relative sizes, so the PPMR increases. And at the lowest sizes, this could be due to a limitation in sampling, and the fact that we cannot sample extremely small prey, which mackerel would be eating when they are <5g.

I will do the diet contribution now, to see if it is the same. Then, I will plot only the middle ranges, to see if it changes much.

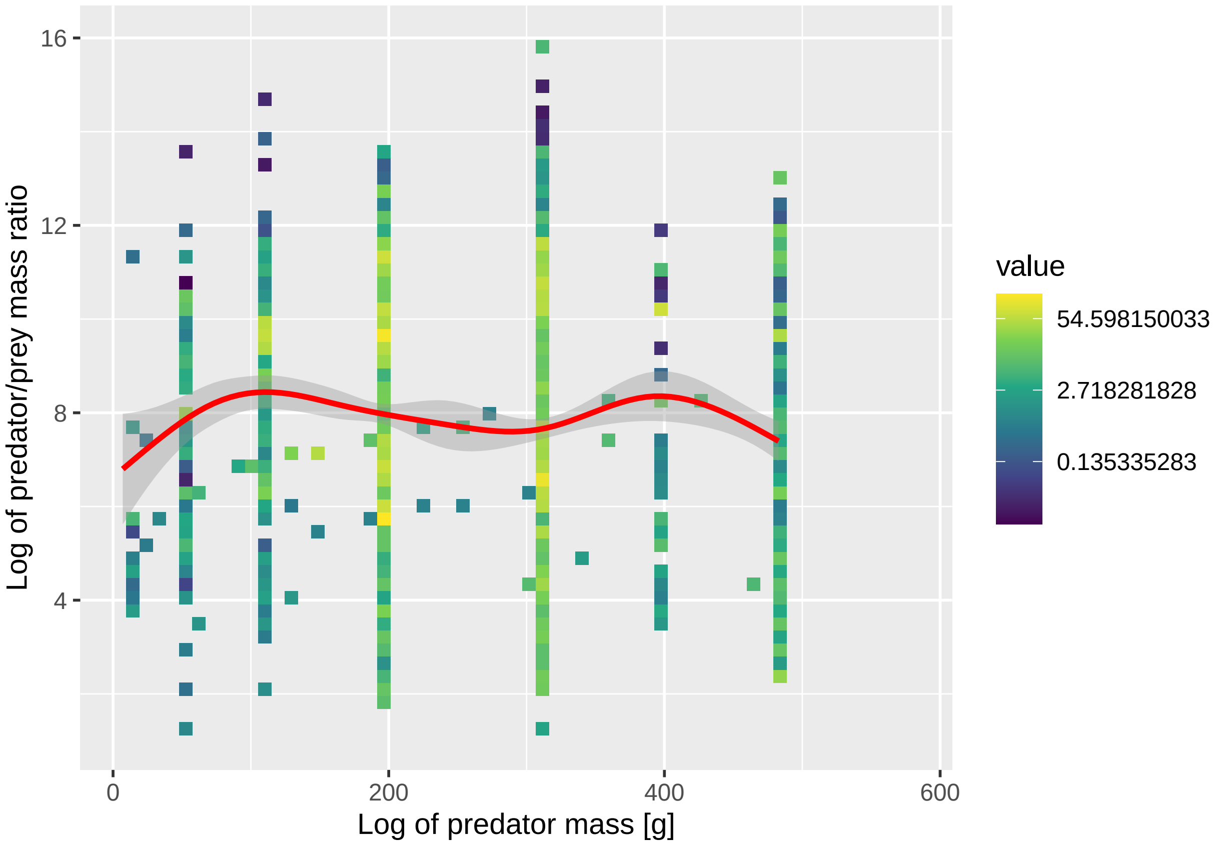
```
#diet contribution
dig <- 2/3

ggplot(mack, aes(x = wpredator, y = log(ppmr))) +
  stat_summary_2d(aes(z = Nprey * wprey^dig), fun = "sum", bins = 60) +
  scale_fill_viridis_c(trans = "log") +
  geom_smooth(aes(weight = Nprey * wprey^dig), colour = "red") +
  xlab("Log of predator mass [g]") +
  ylab("Log of predator/prey mass ratio")
```

```
## ‘geom_smooth()‘ using method = ’gam’ and formula = ’y ~ s(x, bs = "cs")’
```



```
#the same but with a limiation on the predator mass
ggplot(mack, aes(x = wpredator, y = log(ppmr))) +
  stat_summary_2d(aes(z = Nprey * wprey^dig), fun = "sum", bins = 60) +
  scale_fill_viridis_c(trans = "log") +
  geom_smooth(aes(weight = Nprey * wprey^dig), colour = "red") +
  xlab("Log of predator mass [g]") +
  ylab("Log of predator/prey mass ratio")+xlim(5, 580)
```

```
## Warning: Removed 57 rows containing non-finite outside the scale range
## (‘stat_summary2d()‘).

## ‘geom_smooth()‘ using method = ’gam’ and formula = ’y ~ s(x, bs = "cs")’

## Warning: Removed 57 rows containing non-finite outside the scale range
## (‘stat_smooth()‘).

## Warning: Removed 4 rows containing missing values or values outside the scale range
## (‘geom_tile()‘).
```



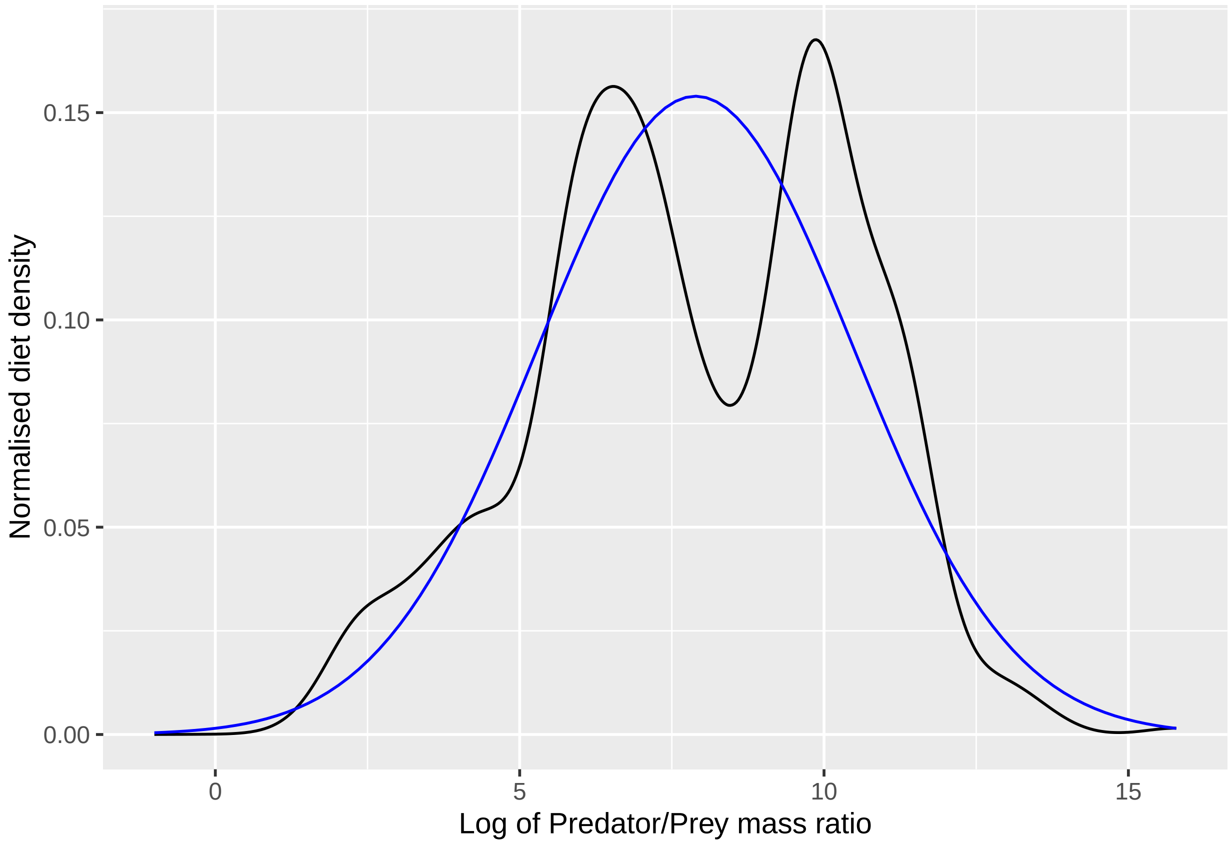
Okay this looks a lot better, but I have had to wiggle the predator mass to include by removing the largest sizes (>600g has a very high PPMR, but then the size range below (580-600g) this has a relatively lower PPMR strangely.) And I have had to make it so that any sizes less than 25g is not included, as the PPMR is much lower. I think this it is reasonable to assume that the sampling methods cannot sample the prey of <25g reliably, as the average PPMR across other size ranges is approximately 3000 (log(3000) = 8), so would correspond to a prey size of 25 / 3000 = 0.008g, which would be hard to sample especially at <25g and would be digested fast inside the mackerel as well.

So therefore, the assumption that PPMR is the same irrespective of predator size is met.

Next, we will check the distribution of PPMR values and try to fit a normal distribution.

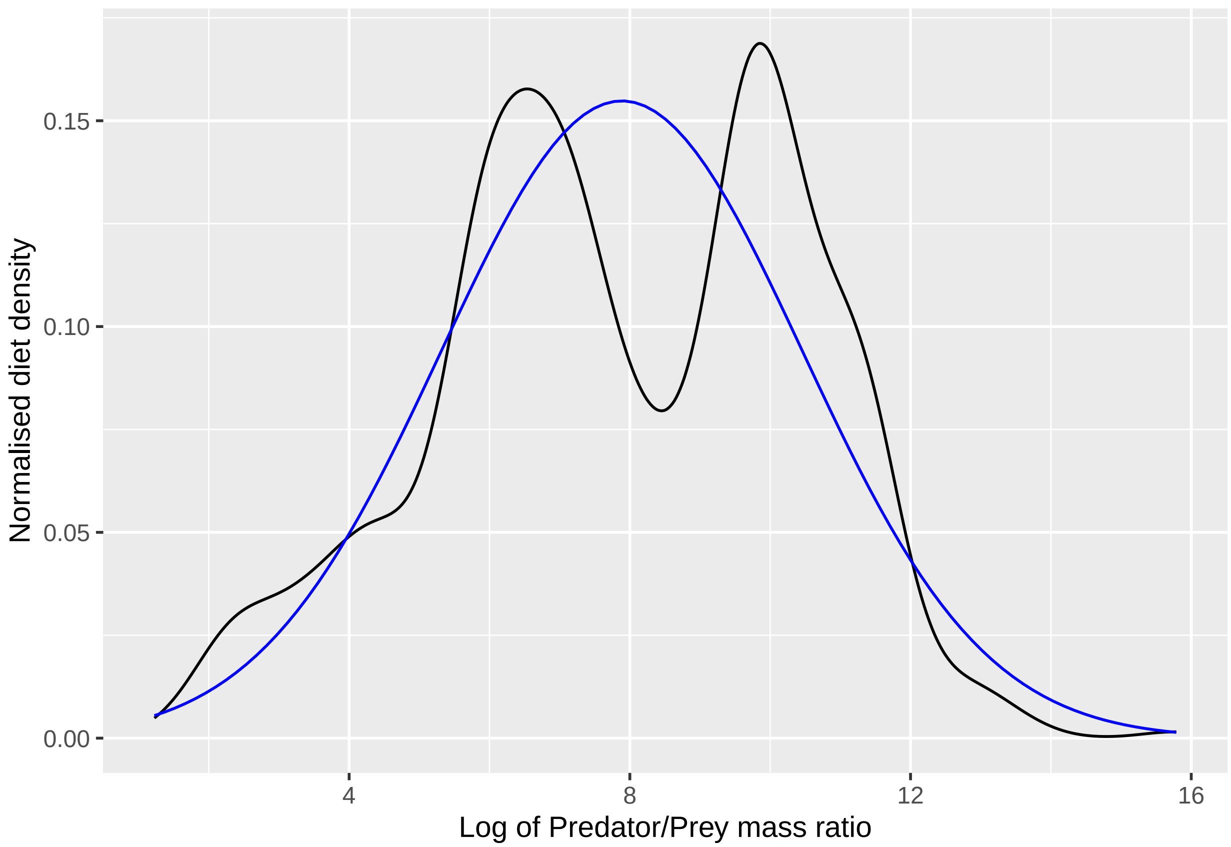
```
# I want to match the format I have to this code
colnames(mack) <- c("w_prey", "w_pred", "n_prey", "pred_species", "ppmr")
#now adding a log ppmr column
colnames(mack) <- make.names(colnames(mack), unique = TRUE)

mack <- mack%>%mutate(log_ppmr=log(ppmr))

stomach <- mack

weighted.sd <- function(x, w) { sqrt(sum(w * (x - weighted.mean(x, w))^2)) }
weight <- stomach$n_prey * stomach$w_prey^dig
weight <- weight / sum(weight)

est_mean <- weighted.mean(stomach$log_ppmr, weight)
est_sd <- weighted.sd(stomach$log_ppmr, weight)
```

```
ggplot(stomach) +
  geom_density(aes(log_ppmr, weight = n_prey * w_prey^dig), bw = 0.5) +
  stat_function(fun = dnorm, args = list(mean = est_mean, sd = est_sd), colour = "blue") +
  xlab("Log of Predator/Prey mass ratio") + ylab("Normalised diet density")
```



Ok, this isnt a terrible fit, but it seems Horse Mackerel eat multimodal diets, I will do the same but also remove the outliers (at the smallest and largest sizes)

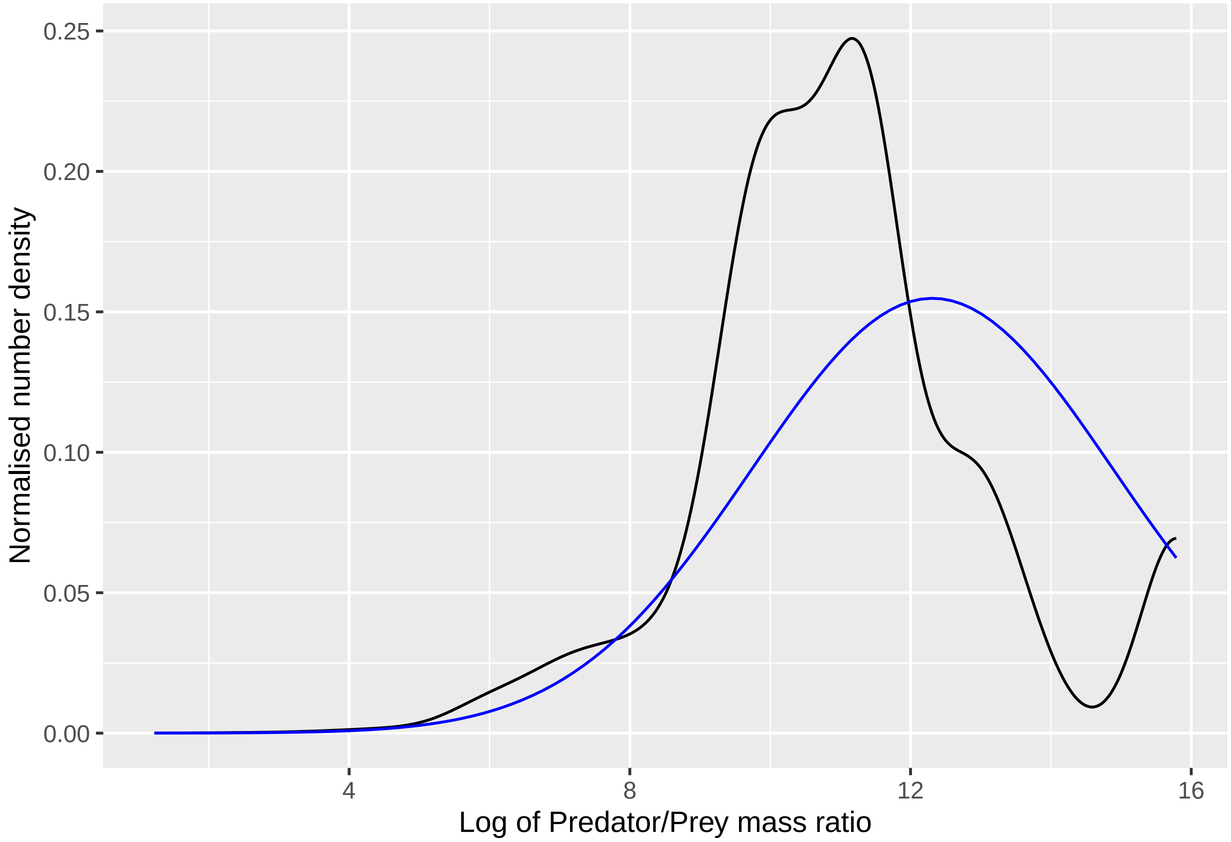
```
stomach <- stomach %>% filter(w_pred > 5, w_pred < 580)

weighted.sd <- function(x, w) { sqrt(sum(w * (x - weighted.mean(x, w))^2)) }
weight <- stomach$n_prey * stomach$w_prey^dig
weight <- weight / sum(weight)

est_mean <- weighted.mean(stomach$log_ppmr, weight)
est_sd <- weighted.sd(stomach$log_ppmr, weight)

ggplot(stomach) +
  geom_density(aes(log_ppmr, weight = weight), bw = 0.5) +
  stat_function(fun = dnorm, args = list(mean = est_mean, sd = est_sd), colour = "blue") +
  xlab("Log of Predator/Prey mass ratio") + ylab("Normalised diet density")
```



OK, this is the same.

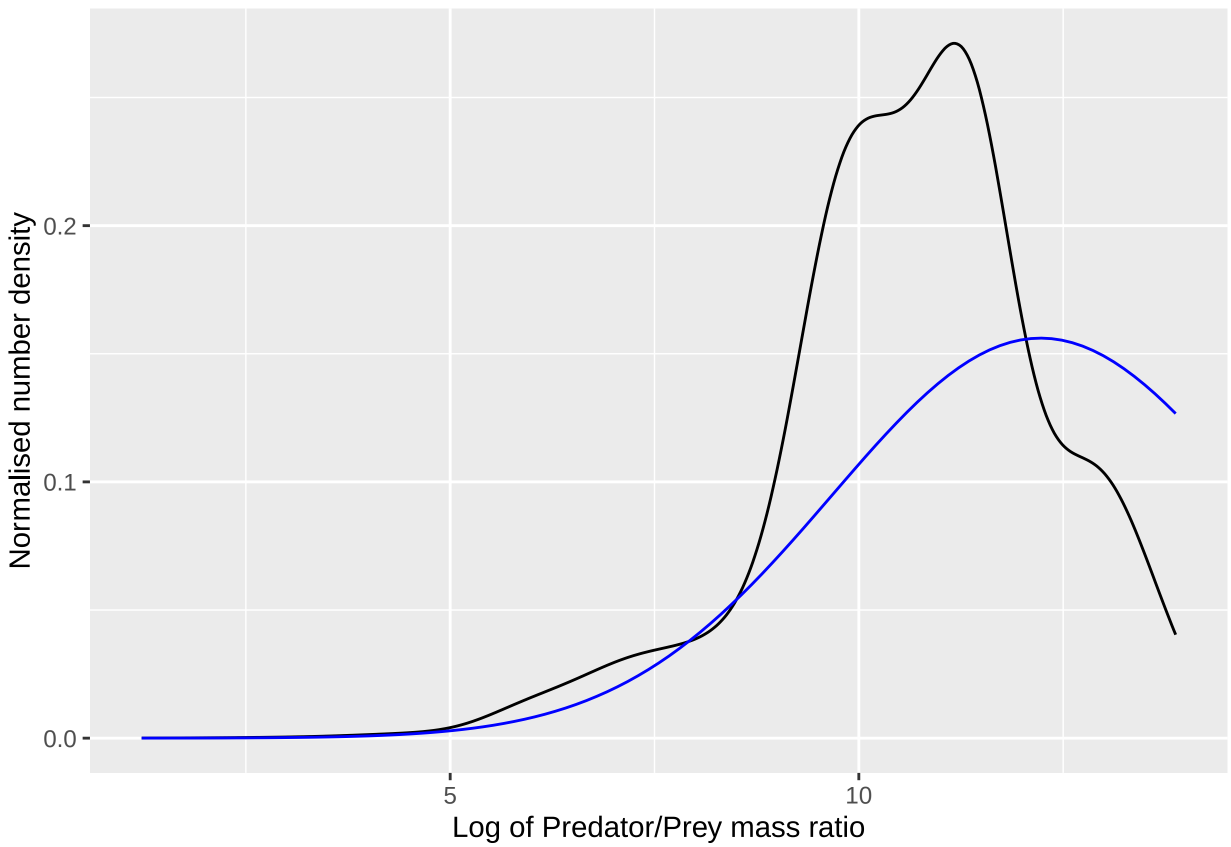
I am now going to check it for the number density as well. Fitting the same sd but with a transformed mean.

```
ggplot(stomach) +
  geom_density(aes(log_ppmr, weight = n_prey), bw = 0.5) +
  stat_function(fun = dnorm, args = list(mean = est_mean + dig * est_sd^2, sd = est_sd), colour = "blue
  xlab("Log of Predator/Prey mass ratio") + ylab("Normalised number density")
```



This doesnt fit well at all. But it look like to it could be better if the outliers at ppmr of 16 were removed.

```
stomach <- stomach %>% filter(log_ppmr < 14)

weighted.sd <- function(x, w) { sqrt(sum(w * (x - weighted.mean(x, w))^2)) }
weight <- stomach$n_prey * stomach$w_prey^dig
weight <- weight / sum(weight)

est_mean <- weighted.mean(stomach$log_ppmr, weight)
est_sd <- weighted.sd(stomach$log_ppmr, weight)

ggplot(stomach) +
  geom_density(aes(log_ppmr, weight = n_prey), bw = 0.5) +
  stat_function(fun = dnorm, args = list(mean = est_mean + dig * est_sd^2, sd = est_sd), colour = "blue
  xlab("Log of Predator/Prey mass ratio") + ylab("Normalised number density")
```



So I don't think that the normal distribution fits very well to Horse Mackerel, so we will need to fit another distribution.

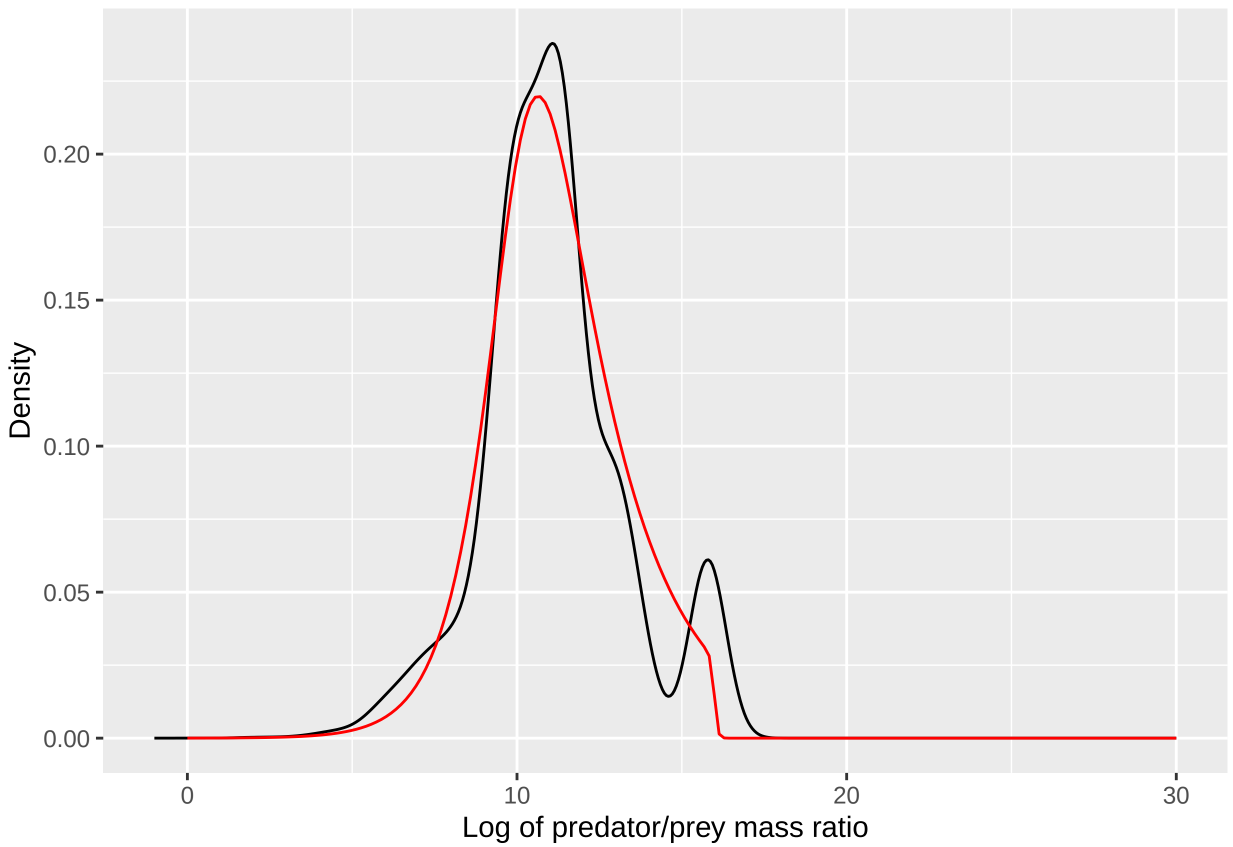
Now lets try the code to fit a truncated exponential distribution to the data.

Okay, this is code to define the exponential distribution (taken from the PPMR distribution file.) First, I will do this for the number distribution

```
stomach <- mack
colnames(stomach) <- c("wprey", "w_pred", "Nprey", "pred_species", "ppmr", "l")
stomach <- stomach %>% mutate(weight_numbers = Nprey / sum(Nprey))
#stomach <- stomach%>%mutate(weight_numbers=Nprey*wprey^(2/3))

fl <- function(l, alpha, ll, ul, lr, ur) {
  dl <- ll - l
  dr <- l - lr
  fl <- exp(alpha * l) /
    (1 + exp(ul * dl)) /
    (1 + exp(ur * dr))
  # fl[fl <= 0] <- 0
}

dtexp <- function(l, alpha, ll, ul, lr, ur) {
  d <- fl(l, alpha, ll, ul, lr, ur) /
    integrate(fl, 0, 30, alpha = alpha,
              ll = ll, ul = ul, lr = lr, ur = ur)$value
  return(d)
}

mle_texp <- function(df) {
  loglik <- function(alpha, ll, ul, lr, ur) {
    L <- dtexp(stomach$l, alpha, ll, ul, lr, ur)
    - sum(log(L) *  stomach$weight_numbers)
  }
  mle2(loglik, start = list(
    alpha = 0.5,
    ll = min(stomach$l),
    lr = max(stomach$l),
    ul = 5,
    ur = 5))
}
#setting th weights
#stomach <- stomach%>%mutate(weight_numbers=Nprey*wprey^(2/3))
#or for number density
#stomach <- stomach%>%mutate(weight_numbers=Nprey/sum(Nprey))

est <- mle_texp(stomach)
```

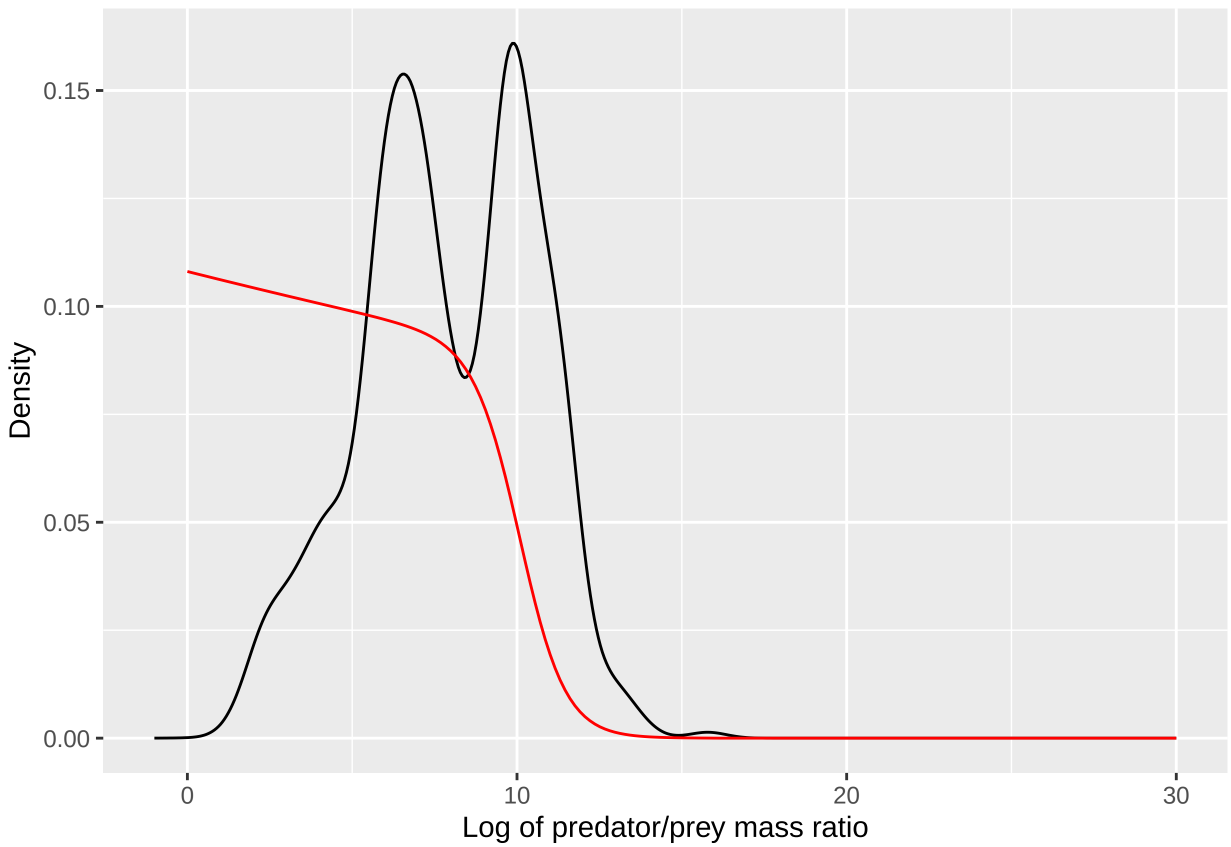
```
## Warning in mle2(loglik, start = list(alpha = 0.5, ll = min(stomach$l), lr =
## max(stomach$l), : convergence failure: code=1 (iteration limit 'maxit' reached)
```

```
#extracting coefficients
estco <- est@fullcoef

grid = seq(0, 30, length.out = 200)

dist <- dtexp(grid, alpha = estco[1], ll = estco[2], ul = estco[3], lr = estco[4], ur = estco[5])

dist <- data.frame(l=grid, Density=dist)

ggplot(stomach) +
  geom_density(aes(l, weight=weight_numbers))+
 xlab("Log of predator/prey mass ratio") +
  geom_line(aes(l, Density), data = dist, color = "red")
```



Ok, this works, it fits well. Now I have to fit this same distribution to the biomass density, which as defined in the PPMR distributions document, you do alpha-1

```
stomach <- stomach%>%mutate(biomass=Nprey*wprey^(2/3))

grid = seq(0, 30, length.out = 200)
#here, the alpha is meant to be -1, but I have to subtract 0.7 to make it work,
#so I am going to run the distribution again
#for the biomass, and see the difference
dist <- dtexp(grid, alpha = (estco[1]-1), ll = estco[2], ul = estco[3], lr = estco[4], ur = estco[5])

dist <- data.frame(l=grid, Density=dist)

ggplot(stomach) +
  geom_density(aes(l, weight=biomass))+
 xlab("Log of predator/prey mass ratio") +
  geom_line(aes(l, Density), data = dist, color = "red")
```

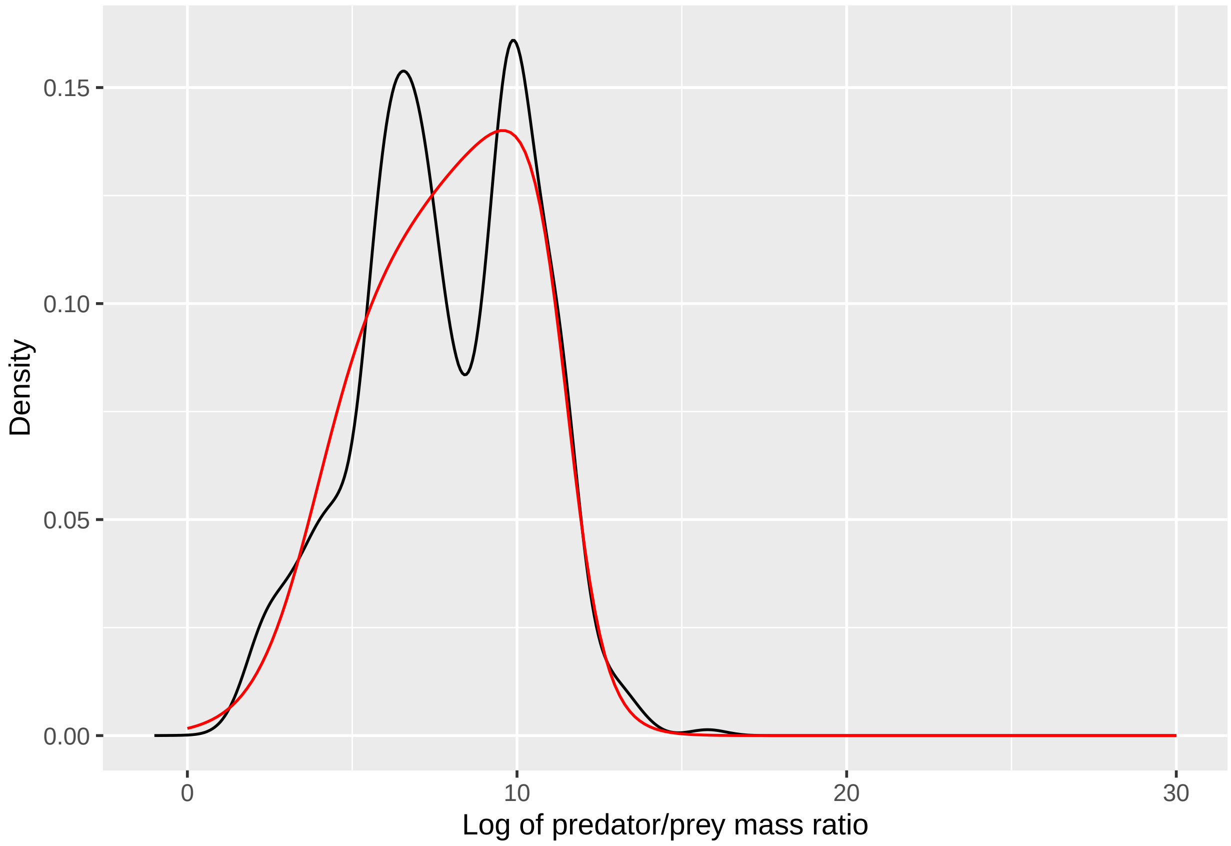


Ok this doesn't fit as well. I will calculate the proper distribution independently and see what the difference is.

It isn't a terrible fit when the alpha is -0.7, which is strange, I am not sure why this might be.

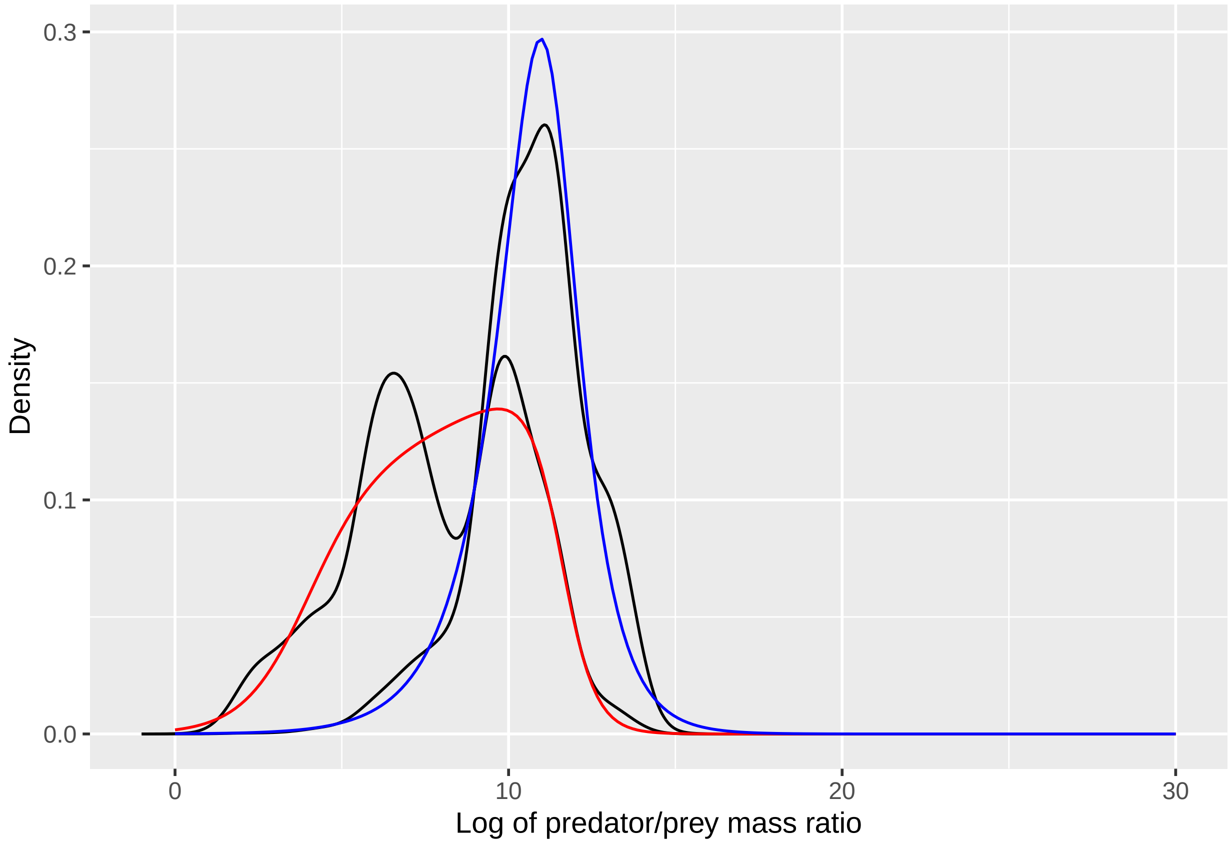
(when trying to use this code to calculate the biomass distribution, I get an error, Error in integrate(fl, 0, 30, alpha = alpha, ll = ll, ul = ul, lr = lr,: non-finite function value), this code works for it though, but provides a slightly different distribution for the number density. The difference in the code is this line , method = "L-BFGS-B", control = list(maxit = 10000) in the MLE function, changing the optimisation algorithm and the maximum number of iterations.)

```
#library(bbmle)
#library(dplyr)
#
##884 is problem line for the number density (idk why)
##stomach <- stomach[-884,]

## Define the function fl with debugging
fl <- function(l, alpha, ll, ul, lr, ur) {
  dl <- ll - l
  dr <- l - lr
  fl_values <- exp(alpha * l) / (1 + exp(ul * dl)) / (1 + exp(ur * dr))

  # Debugging output
  if (any(!is.finite(fl_values))) {
    print("Non-finite fl values found")
    print(fl_values)
  }

  return(fl_values)
}

## Define the truncated exponential PDF with debugging
dtexp <- function(l, alpha, ll, ul, lr, ur) {
  fl_values <- fl(l, alpha, ll, ul, lr, ur)

  integral_result <- tryCatch(
    integrate(fl, 0, 30, alpha = alpha, ll = ll, ul = ul, lr = lr, ur = ur),
    error = function(e) {
      print("Integration failed")
     print(e)
      return(NULL)
    }
   )

   if (is.null(integral_result)) {
     return(rep(NA, length(l)))
   }

   d <- fl_values / integral_result$value

  # Debugging output
  if (any(!is.finite(d))) {
    print("Non-finite d values found")
    print(d)
  }

  return(d)
}

 #Define the MLE function with debugging
mle_texp <- function(df) {
  loglik <- function(alpha, ll, ul, lr, ur) {
    L <- dtexp(df$l, alpha, ll, ul, lr, ur)

    # Debugging output
    if (any(!is.finite(L) | L <= 0)) {
     print("Non-finite or non-positive L values found")
      print(which(!is.finite(L) | L <= 0))
      return(Inf)
    }

    -sum(log(L) * df$weight_numbers)
  }

  result <- tryCatch(
    mle2(loglik, start = list(
      alpha = 0.5,
      ll = min(df$l),
      lr = max(df$l),
      ul = 5,
      ur = 5
    ), method = "L-BFGS-B", control = list(maxit = 10000)),
    error = function(e) {
      print("MLE fitting failed")
      print(e)
      return(NULL)
    }
  )

  return(result)
}

# Assuming 'stomach' is already defined
# setting the weights
stomach <- stomach %>% mutate(weight_numbers = Nprey * wprey^(2/3))
# or for number density
#stomach <- stomach %>% mutate(weight_numbers = Nprey / sum(Nprey))

# Fit the model
est <- mle_texp(stomach)

biomassco <- est@coef

grid = seq(0, 30, length.out = 200)
#here, the alpha is meant to be -1, but I have to subtract 0.7 to make it work, so I am going to run th
#for the biomass, and see the difference
dist <- dtexp(grid, alpha = (biomassco[1]), ll = biomassco[2], ul = biomassco[3],
              lr = biomassco[4], ur = biomassco[5])

dist <- data.frame(l=grid, Density=dist)

  ggplot(stomach) +
  geom_density(aes(l, weight=weight_numbers))+
 xlab("Log of predator/prey mass ratio") +
  geom_line(aes(l, Density), data = dist, color = "red")
```



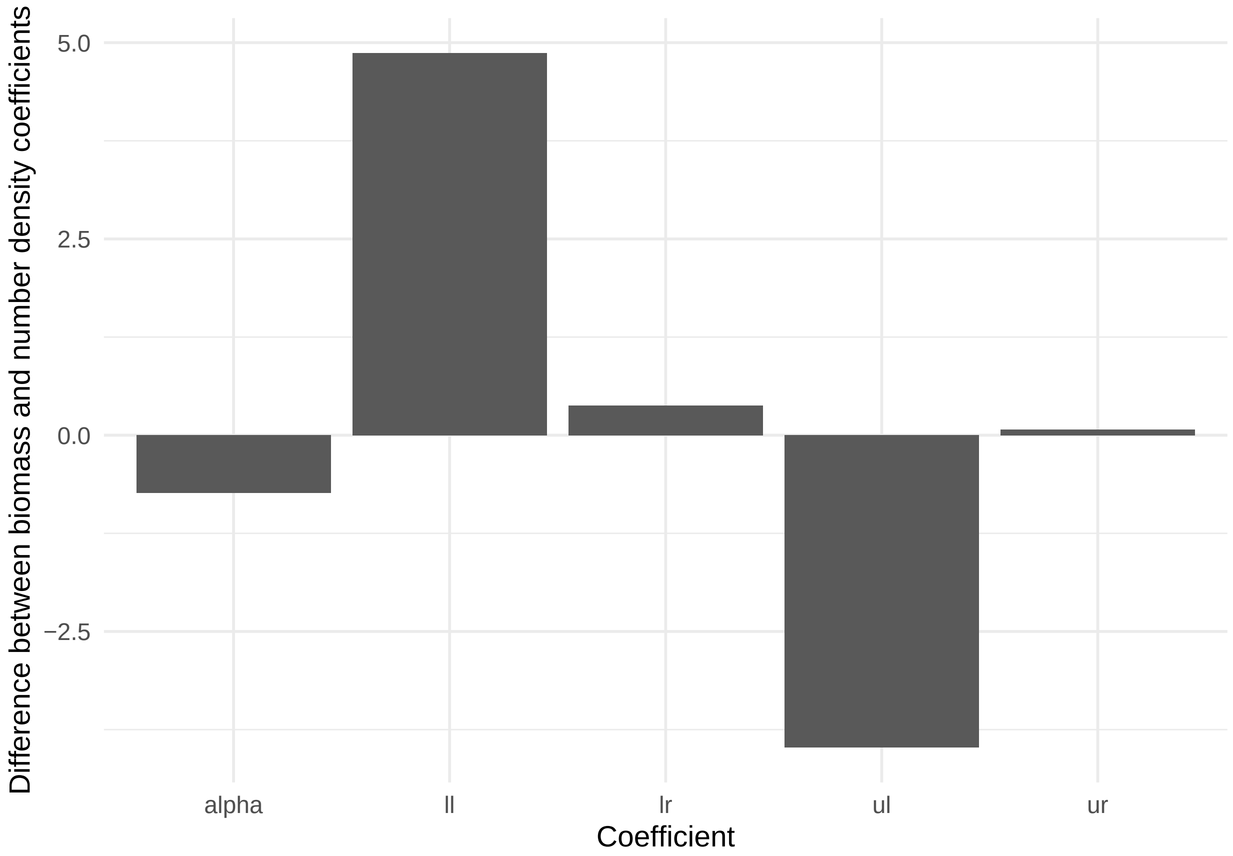
Ok, fitting the biomass distribution works well. I will plot the fits of both distributions below, while using the same code for both distributions

```
#884 is problem line for the number density (idk why)
stomach <- stomach[-884,]

stomach <- stomach %>% mutate(weight_numbers = Nprey * wprey^(2/3))
est <- mle_texp(stomach)
biomassestco <- est@coef

stomach <- stomach %>% mutate(weight_numbers = Nprey / sum(Nprey))
est <- mle_texp(stomach)
numberestco <- est@coef

grid = seq(0, 30, length.out = 200)
dist <- dtexp(grid, alpha = (biomassestco[1]), ll = biomassestco[2], ul = biomassestco[3], lr = biomasse
biomassdist <- data.frame(l=grid, Density=dist)

dist <- dtexp(grid, alpha = (numberestco[1]), ll = numberestco[2], ul = numberestco[3], lr = numberestc
numberdist <- data.frame(l=grid, Density=dist)

#now plot these two together

stomach <- stomach %>% mutate(biomass = Nprey * wprey^(2/3))

ggplot(stomach) +
  geom_density(aes(l, weight=weight_numbers))+
  geom_density(aes(l, weight=biomass))+
 xlab("Log of predator/prey mass ratio") +
  geom_line(aes(l, Density), data = biomassdist, color = "red")+
  geom_line(aes(l, Density), data = numberdist, color = "blue")
```



So they fit their individual distributions well. but the differences between the coefficients are quite large. I will calculate the difference between the two coefficients

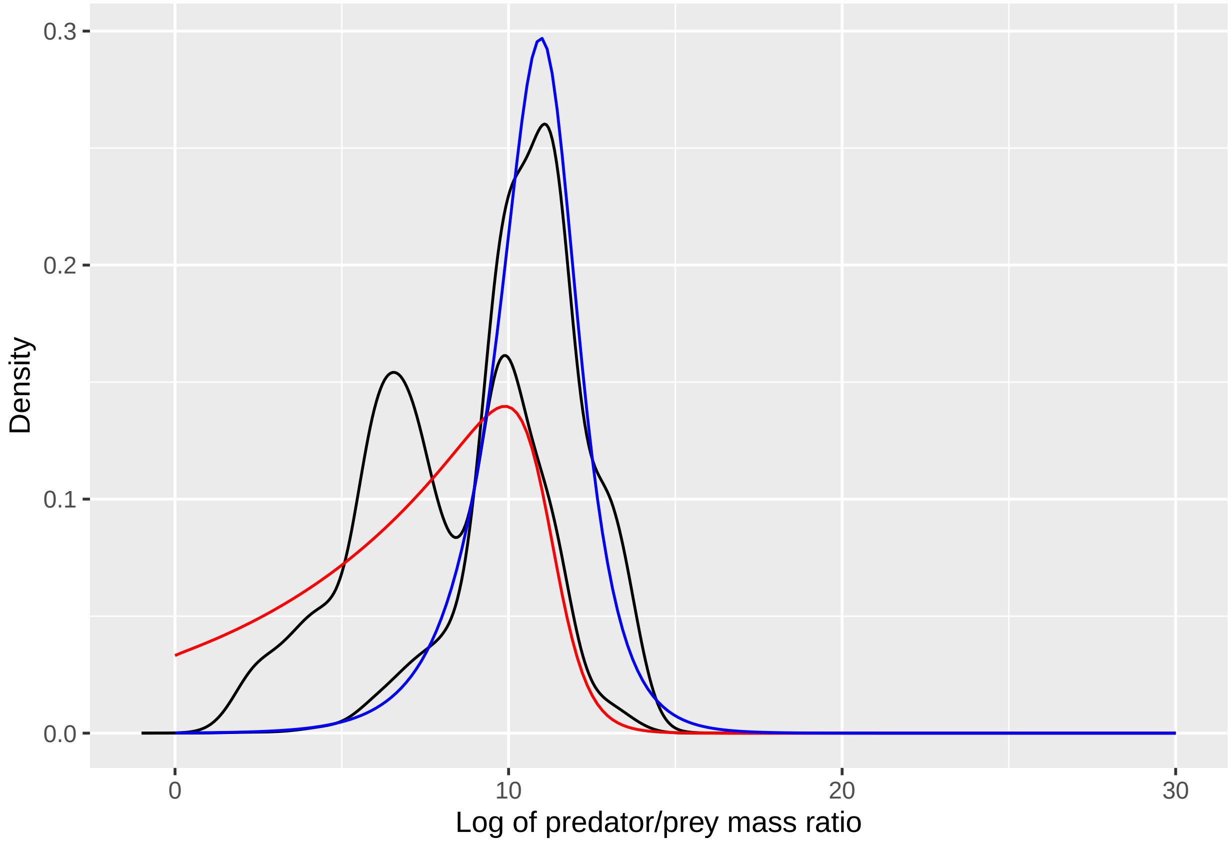
```
diffco <- biomassestco-numberestco

#now ggplot the diffco
diffco <- data.frame(Coefficient = labels <- c("alpha", "ll", "ul", "lr", "ur"),diffco)

ggplot(diffco, aes(x = factor(Coefficient), y = diffco)) +
  geom_bar(stat = "identity") +
  xlab("Coefficient") +
  ylab("Difference between biomass and number density coefficients") +
  theme_minimal()
```



The difference between the alpha is not 1, it is about 0.625. I am not sure why, as stated in the PPMR document, the difference should be 1 (I am also not sure why this is)

Now I will just plot the distribution derived from the number density, and I will try to change the alpha value to calculate the truncated exponential distribution for the biomass density by the difference seen above (-0.625).

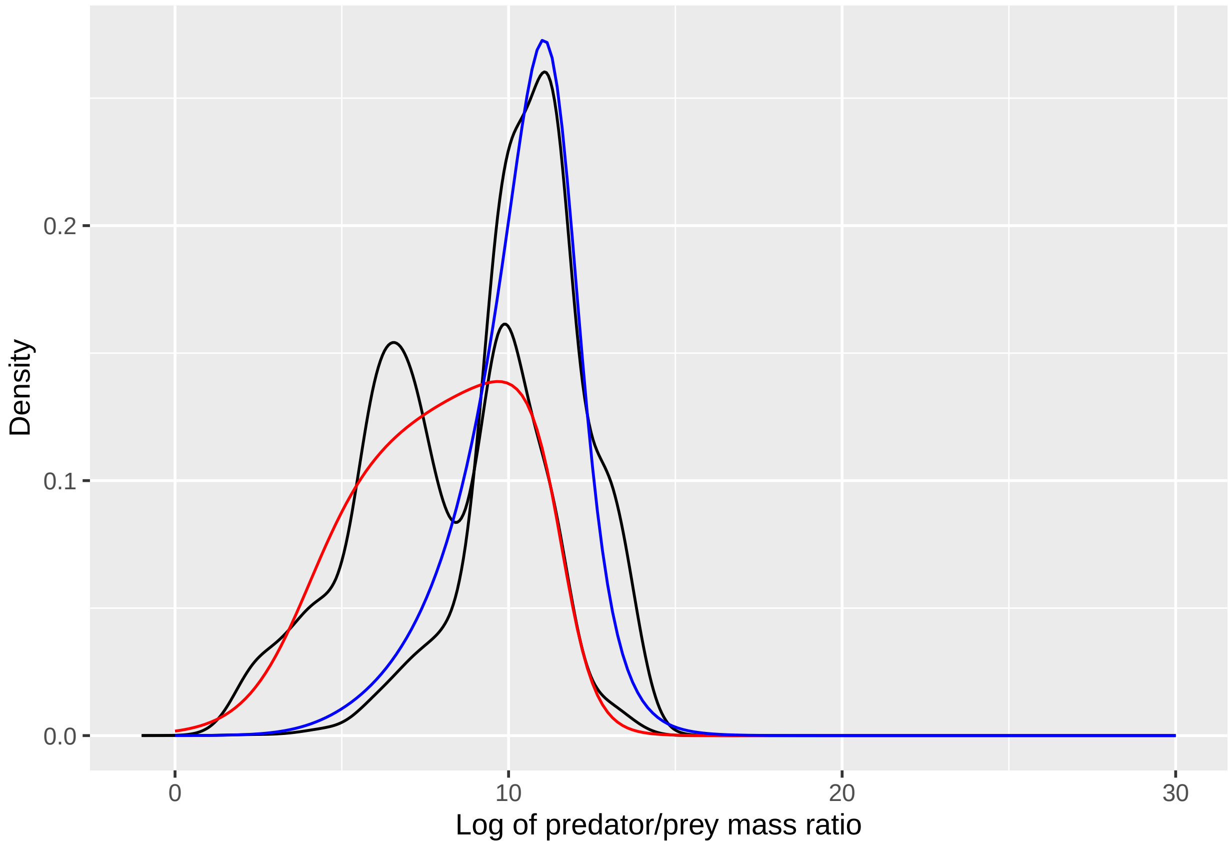
```
dist <- dtexp(grid, alpha = (numberestco[1]), ll = numberestco[2], ul = numberestco[3], lr = numberestc
numberdist <- data.frame(l=grid, Density=dist)
dist <- dtexp(grid, alpha = (numberestco[1]-0.625), ll = numberestco[2], ul = numberestco[3], lr = numbe
biomassdist <- data.frame(l=grid, Density=dist)
#now plot these two together

stomach <- stomach %>% mutate(biomass = Nprey * wprey^(2/3))

ggplot(stomach) +
  geom_density(aes(l, weight=weight_numbers))+
  geom_density(aes(l, weight=biomass))+
 xlab("Log of predator/prey mass ratio") +
  geom_line(aes(l, Density), data = biomassdist, color = "red")+
  geom_line(aes(l, Density), data = numberdist, color = "blue")
```



It fits ok, but it is still not very good for the biomass density (the black line on the left)

I will try to use the density function derived from the biomass density to get a distribution to use for the number density.

```
dist <- dtexp(grid, alpha = (biomassestco[1]), ll = biomassestco[2], ul = biomassestco[3], lr = biomass
biomassdist <- data.frame(l=grid, Density=dist)
dist <- dtexp(grid, alpha = (biomassestco[1]+0.5), ll = biomassestco[2], ul = biomassestco[3], lr = bio
numberdist <- data.frame(l=grid, Density=dist)
#now plot these two together

stomach <- stomach %>% mutate(biomass = Nprey * wprey^(2/3))

ggplot(stomach) +
  geom_density(aes(l, weight=weight_numbers))+
  geom_density(aes(l, weight=biomass))+
 xlab("Log of predator/prey mass ratio") +
  geom_line(aes(l, Density), data = biomassdist, color = "red")+
  geom_line(aes(l, Density), data = numberdist, color = "blue")
```



This fits a lot better, and much better with a difference of 0.5 (I tried to use 0.625, but trial and error showed that 0.5 was better).

Now to understand why the difference has to be 0.5, when it should be 1. This code is taken from the PPMR_distributions document and I have swapped the weighting used there as I am trying to find out what the lowest PPMR should be so that alpha+1 in the biomass distribution gives a good fit for the distribution of the number, whereas in the document they are trying to find out the reverse.

```
est3 <- stomach  %>%
  summarise(lbar = weighted.mean(l, biomass),
            lmax = max(l),
            lmin = min(l),
            alpha = 1/(lmax - lbar),
            lbarw = weighted.mean(l, weight_numbers),
            lmin_B = lbarw + 1 / (alpha - 1),
            diff = lmin - lmin_B )

print(est3)
```

```
## # A tibble: 1 x 7
##    lbar  lmax  lmin alpha lbarw lmin_B  diff
##   <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl>
## 1  7.88  15.1 -1.00 0.139  10.7   9.51 -10.5
```

So the lowest PPMR should be higher if we want the distribution to work?

I don't really follow this all at all.