A preliminary model for Quad-Rig catch comparison

Updated to include additional trial data

Methods for twin-rig catch comparison analysis are set out in Holst and Revill (2009). Here, this model is preliminarily extended to greater than 2 cod-ends, in particular we focus on the quad-rig with 4 cod-ends. All treatment of the data is included as in a tutorial, which can be used as a basis for capacity building in the analysis of gear technology trials.

1 Data

The data used for this example come from the July 2014 diamond cod-end mesh size trials conducted by BIM aboard MFV Celtic Warrior II on the Smalls grounds. The data are read into R and processed as follows:

```
library(gdata)

neph.dat <- read.xls("../data/Celtic Warrior Diamond mesh July 2014 Celtic Sea.xls",
                     sheet = "Nephrops Lengths",
                     stringsAsFactors = FALSE)

## remove Haul 22, as no recordings for 90mm
neph.dat <- subset(neph.dat, HAUL != 22)

## Show the first 2 rows
head(neph.dat, 2)


##           Vessel       DATE HAUL COMPARTMENT Mesh.Size  SPECIES
## 1 Celtic Warrior 2014-07-19    1     Control      70mm Nephrops
## 2 Celtic Warrior 2014-07-19    1     Control      70mm Nephrops
##   Carapace.Length..mm.. COUNT SUBSRATIO
## 1                    16     1         1
## 2                    17    11         1


## Change the carapace length name
names(neph.dat)[names(neph.dat) == "Carapace.Length..mm.."] <- "Carapace.Length"

## Make the "HAUL" variable character
neph.dat$HAUL <- paste("H", neph.dat$HAUL, sep ="")
```

```
## make some factor variables used in the analyses
neph.dat$fHAUL <- factor(neph.dat$HAUL, levels = unique(neph.dat$HAUL))
neph.dat$fMesh.Size <- factor(neph.dat$Mesh.Size, levels = unique(neph.dat$Mesh.Size))

## remove observations above 99th and below 1th length percentile
## these can be highly influential on the fits
neph.dat <- subset(neph.dat, Carapace.Length < quantile(Carapace.Length, 0.99) &
                   Carapace.Length > quantile(Carapace.Length, 0.01)
                   )
```

Prepare the data for a multinomial fit.

```
## get count per length bin per haul by mesh size
## using the reshape package (makes it easier to process data)
library(reshape)

## variables to keep
vars2keep <- c("fMesh.Size", "Carapace.Length", "fHAUL", "COUNT")

## melt the data frame
neph.melt <- melt(neph.dat[, vars2keep],
                  id = c("fMesh.Size", "Carapace.Length", "fHAUL"))

## re-form the dataframe in required format
neph.cast <- cast(neph.melt, Carapace.Length + fHAUL ~ fMesh.Size  + variable)
neph.cast <- neph.cast[order(neph.cast$fHAUL, neph.cast$Carapace.Length), ]
neph.cast[is.na(neph.cast)] <- 0

## show the first few rows
head(neph.cast, 2)


##    Carapace.Length fHAUL 70mm_COUNT 80mm_COUNT 90mm_COUNT 100mm_COUNT
## 1               15    H1          0          2          1           0
## 24              16    H1          1          3          9           1


## format the subsampling ratio similarly
vars2keep <- c("fMesh.Size", "fHAUL", "SUBSRATIO")

subs.melt <- melt(unique(neph.dat[, vars2keep]), id = c("fMesh.Size", "fHAUL"))

subs.cast <- cast(subs.melt, fHAUL  ~ fMesh.Size + variable)

## merge counts and subsampling ratio back together
neph.cast <- merge(neph.cast, subs.cast, by = "fHAUL", all.x = TRUE)

## show first few lines
head(neph.cast, 2)


##   fHAUL Carapace.Length 70mm_COUNT 80mm_COUNT 90mm_COUNT 100mm_COUNT
```

```
## 1    H1              15          0          2          1           0
## 2    H1              16          1          3          9           1
##   70mm_SUBSRATIO 80mm_SUBSRATIO 90mm_SUBSRATIO 100mm_SUBSRATIO
## 1              1              1              1               1
## 2              1              1              1               1
```

```
## Extract the matrix of counts
count.vars <- c("70mm_COUNT", "80mm_COUNT", "90mm_COUNT", "100mm_COUNT")

neph.count.mat <- as.matrix(neph.cast[, count.vars])

colnames(neph.count.mat) <- c("70mm_COUNT", "80mm_COUNT", "90mm_COUNT",
                              "100mm_COUNT")

## Extract the matrix of subsampling ratios
subsratio.vars <- c("70mm_SUBSRATIO", "80mm_SUBSRATIO", "90mm_SUBSRATIO",
                    "100mm_SUBSRATIO")

subsratio.mat <- as.matrix(neph.cast[, subsratio.vars])

## Create the offset (NEED TO CHECK THIS)
offset.mat <- log(apply(subsratio.mat, 2, FUN =
                        function(zz){zz/subsratio.mat[,1]}))
```

Plot the data

```
library(ggplot2)

## Get the proportions
count.mesh <- as.matrix(neph.cast[, count.vars])

prop.mesh <- prop.table(count.mesh, margin = 1)

m <- dim(prop.mesh)[1]

## make a dataframe of the proportions for ggplot
prop.mesh.df <- data.frame(
                  Mesh.Size = factor(rep(c("70mm", "80mm", "90mm", "100mm"),
                    each = m), levels = c("70mm", "80mm", "90mm", "100mm")),
                  Carapace.Length = rep(neph.cast$Carapace.Length, times = 4),
                  proportion = c(prop.mesh),
                  count = c(count.mesh))

ggplot(prop.mesh.df, aes(x = Carapace.Length, y = proportion)) +
  geom_point(colour = "#F8766D", alpha = 0.2, aes(size = log(count))) +
  facet_wrap(~ Mesh.Size) + ylab("Proportion of Nephrops per cod-end")
```

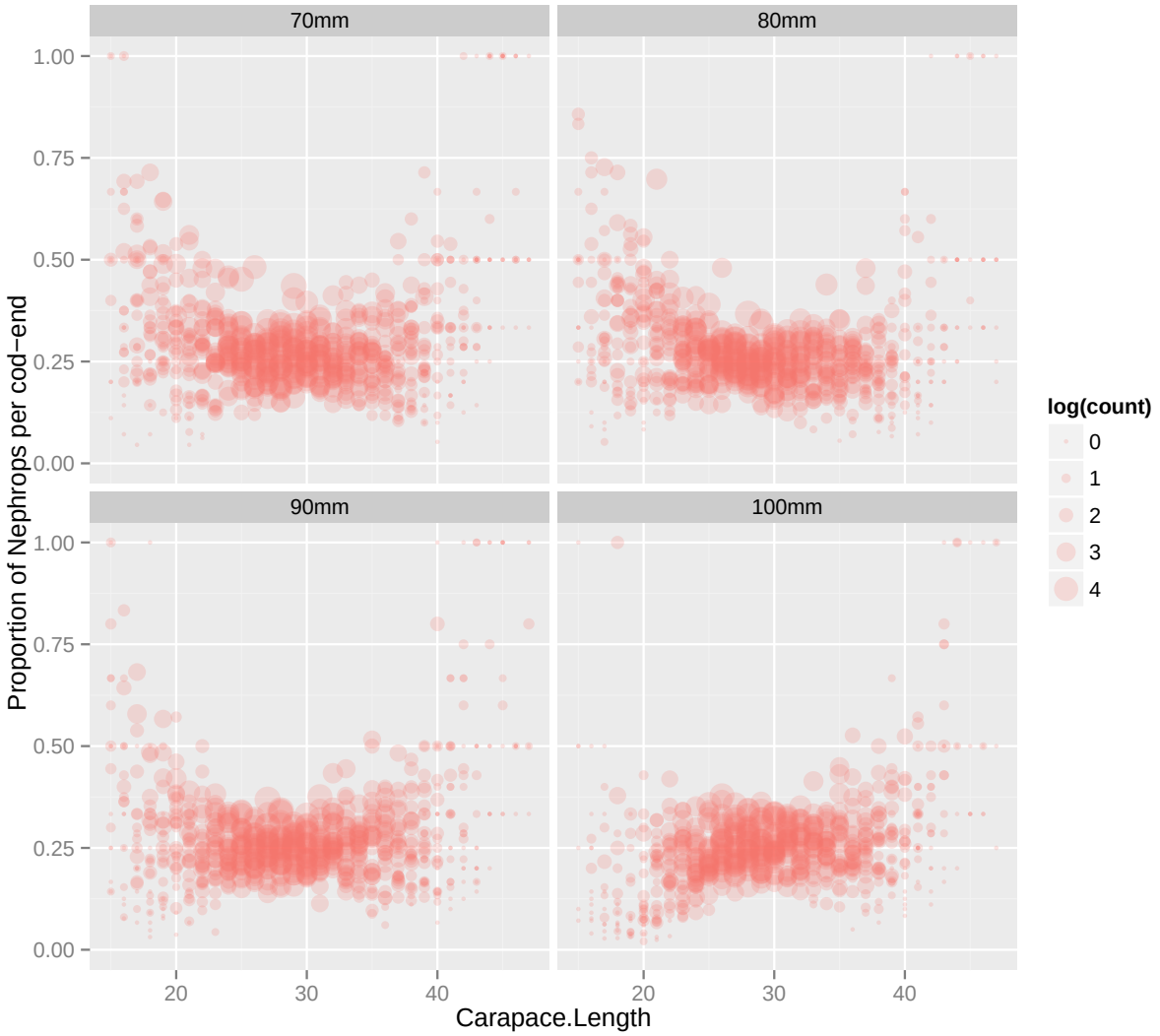


Figure 1: Proportion of Nephrops catch retained per haul. Each point represents the proportion of the Nephrops catch (in number) per haul and length class retained in a given cod-end (70mm, 80mm, 90mm, or 100mm). The size of the point is proportional to the log of the count.

2 Model

The model we focus on is the multinomial, which is a generalization of the binomial to cases with more than two categories (here 4 categories: 70mm, 80mm, 90mm, 100mm). Under the assumption that each net fishes the same, we would expect 25% of the catch to be retained in each net. We can test that hypothesis.

```
library(nnet)

## First fit is constant proportions
## not accounting for length

mnom0 <- multinom(neph.count.mat ~ 1 + offset(offset.mat))


## # weights:  24 (3 variable)
## initial  value 65833.303791
## final  value 65796.538161
## converged


## include carapace length
## first scale it to range between zero and one
max.length <- max(neph.cast$Carapace.Length)
neph.cast$prop.Carapace.Length <- neph.cast$Carapace.Length/max.length

## Extend to third order polynomial (based on AIC and BIC)
neph.cast$prop.Carapace.Length2 <- neph.cast$prop.Carapace.Length^2
neph.cast$prop.Carapace.Length3 <- neph.cast$prop.Carapace.Length^3

##
mnom.length <- multinom(neph.count.mat ~
                        prop.Carapace.Length + prop.Carapace.Length2 + prop.Carapace.Length
                        offset(offset.mat), data = neph.cast)


## # weights:  36 (12 variable)
## initial  value 65833.303791
## iter  10 value 65574.013563
## iter  20 value 65540.974075
## final  value 65538.628411
## converged


AIC(mnom0, mnom.length)


##             df      AIC
## mnom0        3 131599.1
## mnom.length 12 131101.3
```

Get predictions for the fitted model (note this is long-winded here but will be better coded for more

than the preliminary example).

```
## get predictions manually
## CIs not defined in multinomial context but let's try

## fit coefficients
beta.mu <- c(t(coef(mnom.length)))

## fit coefficient variance covariance matrix
Sigma <- vcov(mnom.length)

## number of lengths to predict for
nlength <- 100
pred.prop.length <- seq(min(neph.cast$prop.Carapace.Length),
                        max(neph.cast$prop.Carapace.Length), length = 100)

pred.length <- seq(min(neph.cast$Carapace.Length),
                        max(neph.cast$Carapace.Length), length = 100)

## model matrix
X <- cbind(1, pred.prop.length, pred.prop.length^2, pred.prop.length^3)

## number of times to resample predictions to get CIs
nresamp <- 100
pred.array <- array(NA, dim = c(nlength, 4, nresamp))

## package to draw from multivariate normal
library(mvtnorm)

for(i in 1:nresamp){
  ## print(i)
  beta <- matrix(rmvnorm(1, mean = beta.mu, sigma = Sigma),
                 nrow = 3, byrow = TRUE)
  p80 <- exp(X %*% matrix(beta[1,]))/(1 + rowSums(exp(X %*% t(beta))))
  p90 <- exp(X %*% matrix(beta[2,]))/(1 + rowSums(exp(X %*% t(beta))))
  p100 <- exp(X %*% matrix(beta[3,]))/(1 + rowSums(exp(X %*% t(beta))))
  p70 <- 1 - p80 - p90 - p100
  pred.p <- cbind(p70, p80, p90, p100)
  pred.array[ , , i] <- pred.p
  rm(pred.p)
}

## mean across samples
pred.mu <- apply(pred.array, c(1, 2), mean)

## upper across samples
pred.upper <- apply(pred.array, c(1, 2), quantile, p = 0.975)

## lower across samples
pred.lower <- apply(pred.array, c(1, 2), quantile, p = 0.025)

## bring all together in a data frame for ggplot
```

```
m <- dim(pred.mu)[1]

pred.ci.df <- data.frame(
                Mesh.Size = factor(rep(c("70mm", "80mm", "90mm", "100mm"),
                 each = m), levels = c("70mm", "80mm", "90mm", "100mm")),
                Carapace.Length = rep(pred.length, times = 4),
                proportion = c(pred.mu),
                lower = c(pred.lower),
                upper = c(pred.upper))
```

Finally overlay the fit on the sample proportions

```
p <- ggplot(prop.mesh.df, aes(x = Carapace.Length, y = proportion)) +
  geom_point(colour = "#F8766D", alpha = 0.2, aes(size = log(count))) +
facet_wrap(~ Mesh.Size) + ylab("Proportion of Nephrops per cod-end")

p + geom_ribbon(data=pred.ci.df, aes(ymin = lower, ymax = upper),
                alpha=0.3, fill = "blue") +
  geom_line(data = pred.ci.df, aes(x = Carapace.Length, y = proportion),
            col = "navy", size = 0.5) +
  geom_hline(aes(yintercept = 0.25), linetype = "dashed")
```

2.1 Including weight as a covariate

Make a row per observation and merge with the weight data

```
## get a row per length measurement (raise them also)
n <- nrow(neph.dat)

##neph.dat2 <- neph.dat[rep(1:n, times =
##                          round(neph.dat$COUNT/neph.dat$SUBSRATIO, 0)), ]

## Note: no raising here
##neph.dat2 <- neph.dat[rep(1:n, times = neph.dat$COUNT), ]

weight.dat <- read.xls("../data/Celtic Warrior Diamond mesh July 2014 Celtic Sea.xls",
                       sheet = "Weights",
                       stringsAsFactors = FALSE)

## Show the first 2 rows
head(weight.dat, 2)


##         Date Haul.. Compartment Mesh.Size Species Total.weight..kg.
## 1 2014-07-19      1       TEST1      90mm    Bulk             26.28
## 2 2014-07-19      1       TEST1      90mm Haddock              0.38
##   Sbsample.weight..kg.
## 1
## 2
```

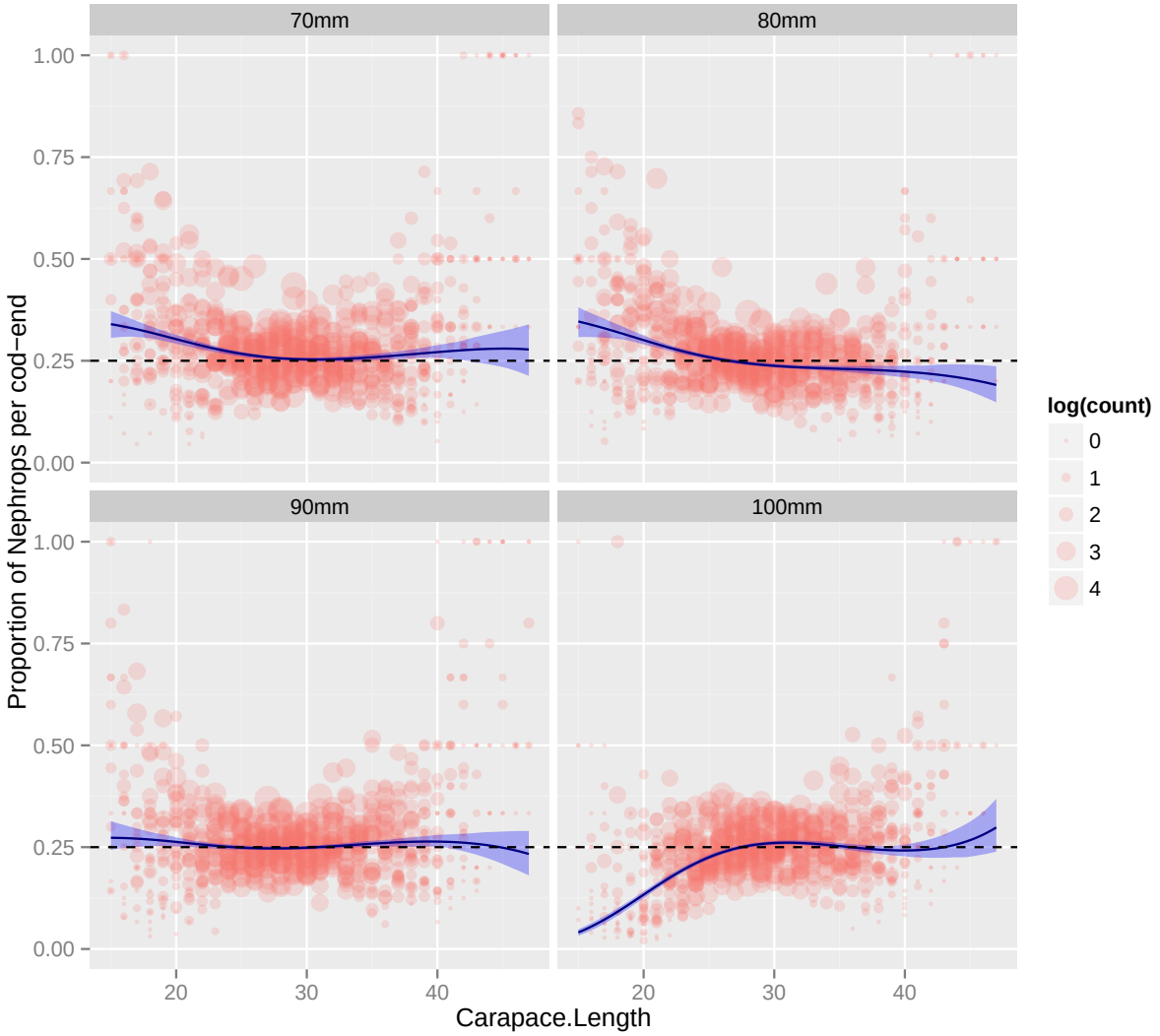


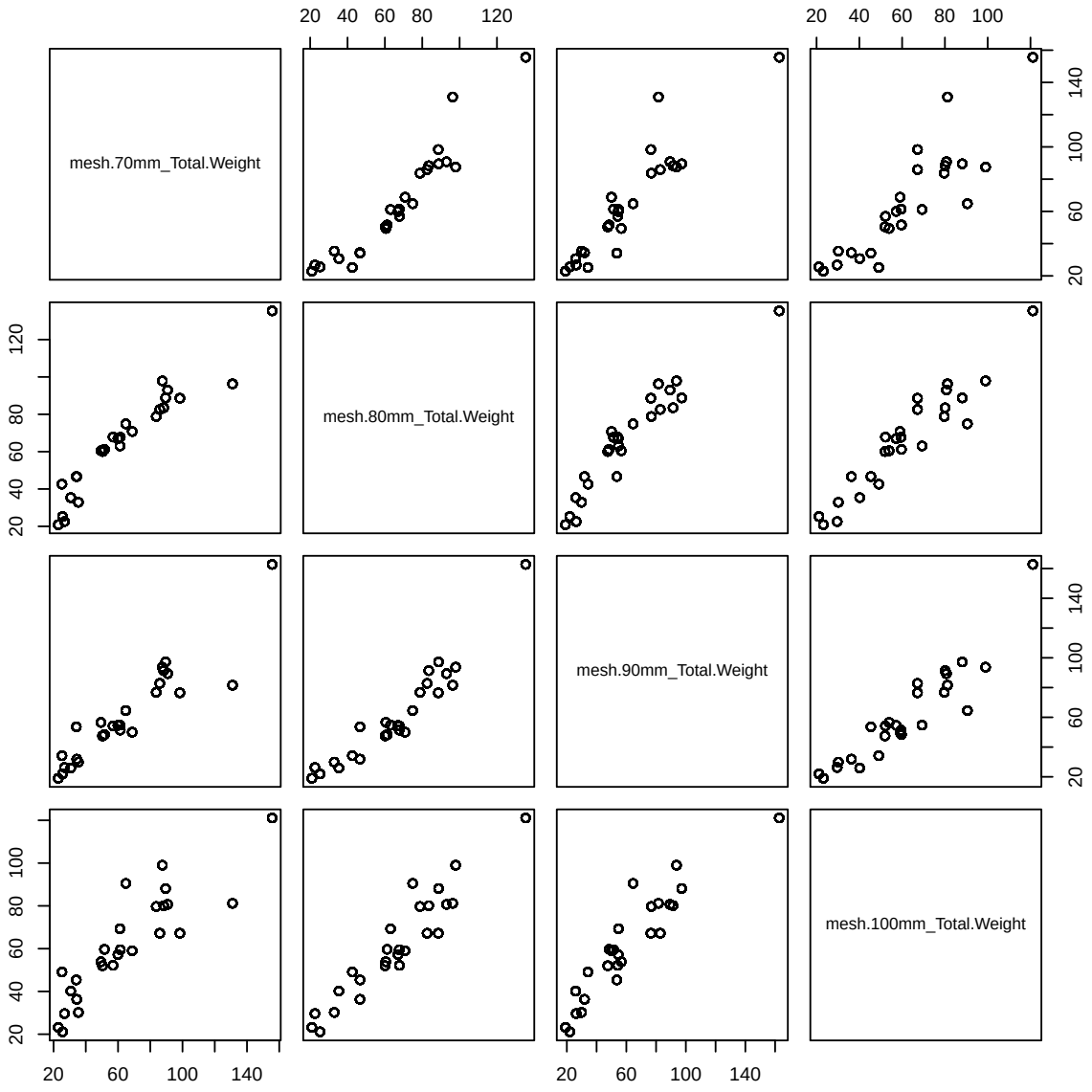
Figure 2: Proportion of Nephrops catch retained per haul with fitted multinomial model and associated re-sampled intervals. Null hypothesis of equal retention is displayed as the dashed line at 0.25.

```
## create a new "HAUL" variable for the merge
weight.dat$HAUL <- paste("H", weight.dat$Haul.., sep ="")

## re-name total weight column
names(weight.dat)[names(weight.dat) == "Total.weight..kg."] <- "Total.Weight"

weight.dat <- subset(weight.dat, Species == "Bulk")


## melt the data frame
weight.melt <- melt(weight.dat[, c("HAUL", "Mesh.Size", "Total.Weight")],
                    id = c("Mesh.Size", "HAUL"))

## re-form the dataframe in required format
weight.cast <- cast(weight.melt, HAUL ~ Mesh.Size  + variable)
weight.cast <- weight.cast[order(weight.cast$HAUL), ]
weight.cast[is.na(weight.cast)] <- 0

## show the first few rows
head(weight.cast, 2)


##   HAUL 100mm_Total.Weight 70mm_Total.Weight 80mm_Total.Weight
## 1   H1              29.64             26.76             22.56
## 2  H10              52.00             50.40             60.10
##   90mm_Total.Weight
## 1             26.28
## 2             47.50


weight.cast$fHAUL <- factor(weight.cast$HAUL)

names(weight.cast)[grep("^[0-9]", names(weight.cast))] <- paste("mesh.", names(weight.cast)

## merge nephrops length and total bulk weight data
neph.dat3 <- merge(neph.cast,
                   weight.cast,
                   by = c("fHAUL"),
                   sort = FALSE
                   )

max.length <- max(neph.dat3$Carapace.Length)
neph.dat3$prop.Carapace.Length <- neph.dat3$Carapace.Length/max.length
neph.dat3$prop.Carapace.Length2 <- neph.dat3$prop.Carapace.Length^2
neph.dat3$prop.Carapace.Length3 <- neph.dat3$prop.Carapace.Length^3
neph.dat3$prop.Carapace.Length4 <- neph.dat3$prop.Carapace.Length^4
```

Include weight in the fit

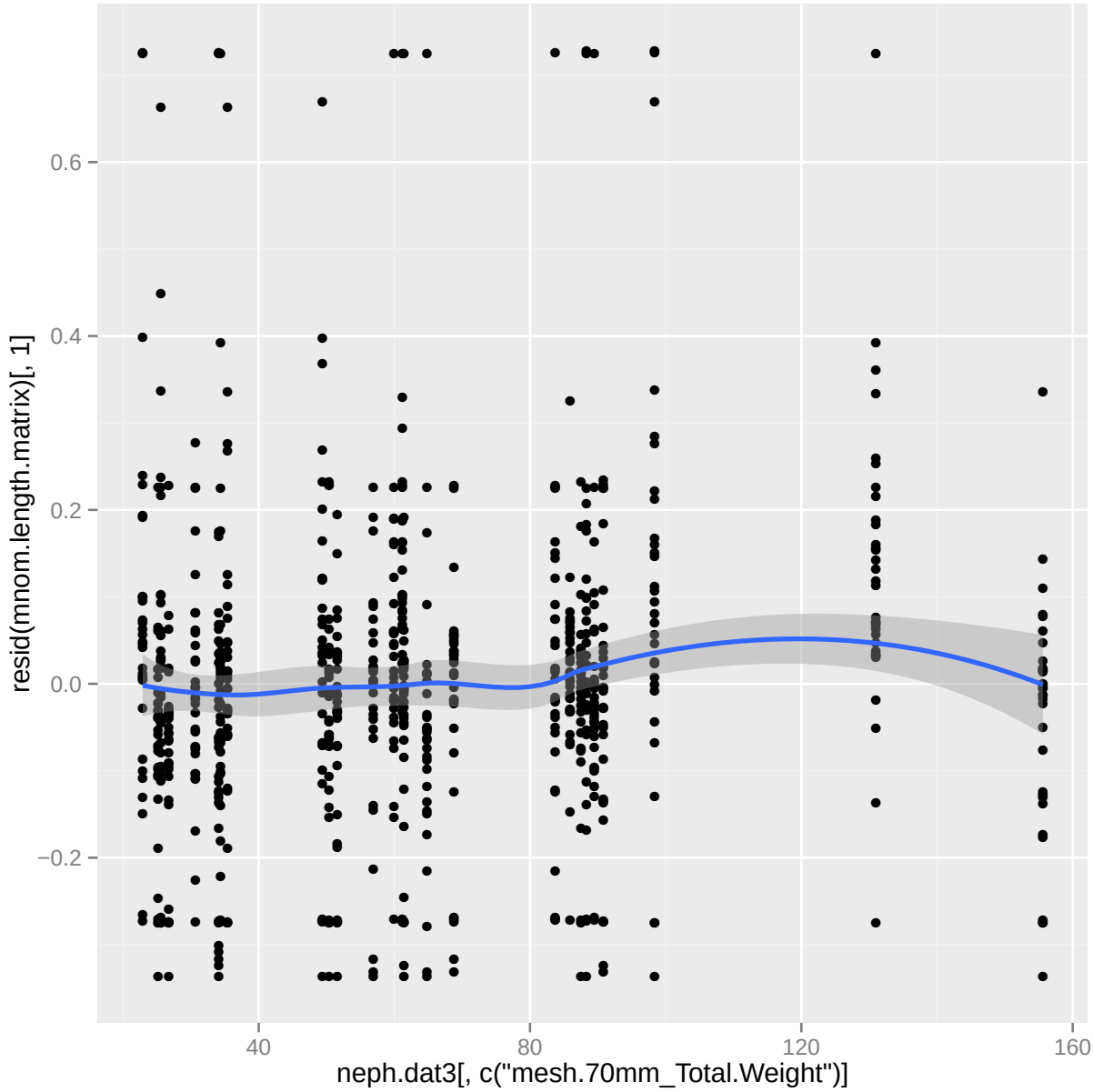
```
## compare two ways of writing same model
```

```r
## in matrix format, as before except without offset for the moment
mnom.length.matrix <- multinom(as.matrix(neph.dat3[, c("70mm_COUNT", "80mm_COUNT", "90mm_C
                               prop.Carapace.Length +
                               prop.Carapace.Length2 +
                               prop.Carapace.Length3,
                               ##offset(offset.mat),
                               data = neph.dat3)


## # weights:  20 (12 variable)
## initial  value 65846.209564
## iter  10 value 65594.036234
## iter  20 value 65560.638668
## final  value 65558.314009
## converged

## take a look at the residuals by bulk weight
plot(neph.dat3[, c("mesh.70mm_Total.Weight", "mesh.80mm_Total.Weight", "mesh.90mm_Total.We
```

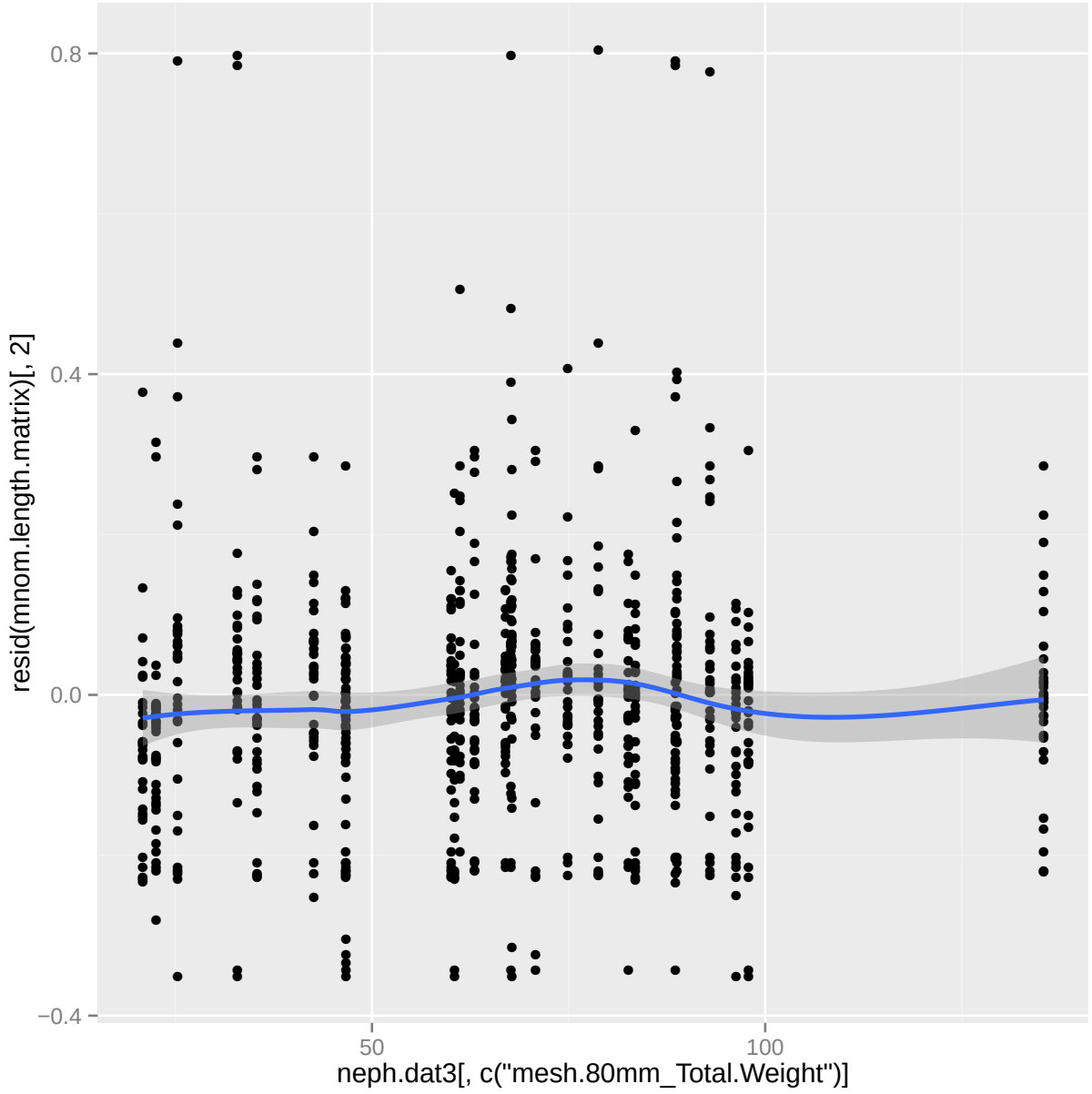



```
## very strong correlation in the counts
round(cor(neph.dat3[, c("mesh.70mm_Total.Weight", "mesh.80mm_Total.Weight", "mesh.90mm_Tota


##                          mesh.70mm_Total.Weight mesh.80mm_Total.Weight
## mesh.70mm_Total.Weight                   1.000                  0.953
## mesh.80mm_Total.Weight                   0.953                  1.000
## mesh.90mm_Total.Weight                   0.927                  0.951
## mesh.100mm_Total.Weight                  0.882                  0.948
##                          mesh.90mm_Total.Weight mesh.100mm_Total.Weight
## mesh.70mm_Total.Weight                   0.927                   0.882
## mesh.80mm_Total.Weight                   0.951                   0.948
```

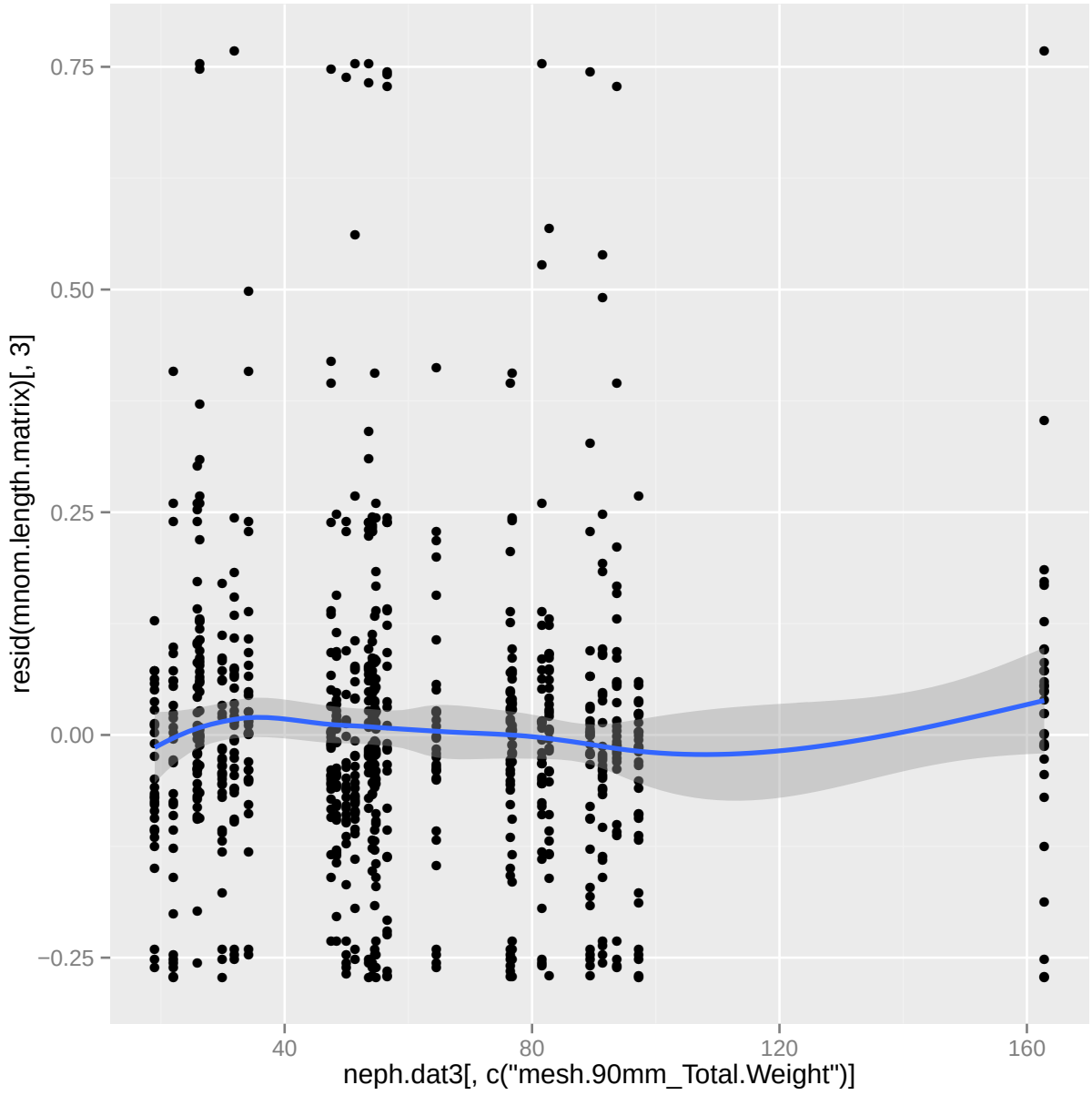
```
## mesh.90mm_Total.Weight                  1.000                  0.927
## mesh.100mm_Total.Weight                 0.927                  1.000


## resids versus weights
qplot(neph.dat3[, c("mesh.70mm_Total.Weight")], resid(mnom.length.matrix)[,1]) + geom_smoot

## geom_smooth:  method="auto" and size of largest group is <1000, so using loess.
Use 'method = x' to change the smoothing method.
```



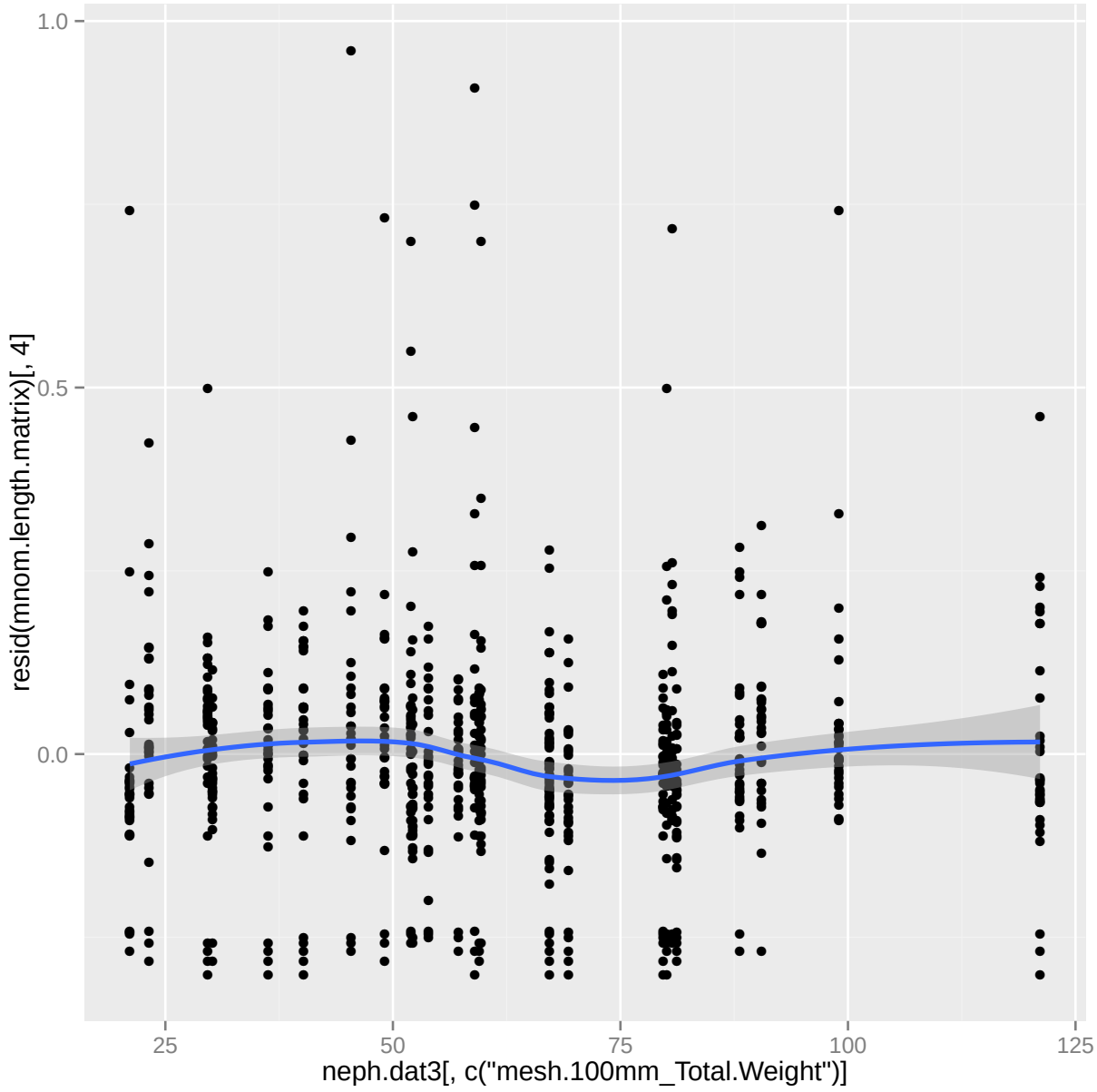
```
qplot(neph.dat3[, c("mesh.80mm_Total.Weight")], resid(mnom.length.matrix)[,2]) + geom_smoot

## geom_smooth:  method="auto" and size of largest group is <1000, so using loess.
Use 'method = x' to change the smoothing method.
```




```
qplot(neph.dat3[, c("mesh.90mm_Total.Weight")], resid(mnom.length.matrix)[,3]) + geom_smoot

## geom_smooth:  method="auto" and size of largest group is <1000, so using loess.
Use 'method = x' to change the smoothing method.
```




```
qplot(neph.dat3[, c("mesh.100mm_Total.Weight")], resid(mnom.length.matrix)[,4]) + geom_smoo

## geom_smooth:  method="auto" and size of largest group is <1000, so using loess.
Use 'method = x' to change the smoothing method.
```




```
## include bulk weight in wide format model, note only including one of the
## bulk weights as the explanatory

neph.dat3$av.Total.Weight <- apply(neph.dat3[, c("mesh.70mm_Total.Weight", "mesh.80mm_Total

mean.wt <- mean(neph.dat3$av.Total.Weight)

neph.dat3$av.Total.Weight.scaled <- neph.dat3$av.Total.Weight - mean.wt

mnom.length.bulk <- multinom(as.matrix(neph.dat3[, c("70mm_COUNT", "80mm_COUNT", "90mm_COU
                             (poly(av.Total.Weight.scaled, 2) + prop.Carapace.Length)^2 +
```

```
                                (poly(av.Total.Weight.scaled, 2) + prop.Carapace.Length2)^2 +
                                (poly(av.Total.Weight.scaled, 2) + prop.Carapace.Length3)^2,
                                ##offset(offset.mat),
                                data = neph.dat3)

## # weights:  52 (36 variable)
## initial  value 65846.209564
## iter  10 value 65578.319726
## iter  20 value 65497.714497
## iter  30 value 65463.674596
## iter  40 value 65455.587103
## iter  50 value 65454.225846
## iter  60 value 65453.755984
## final  value 65453.493336
## converged

## cut the weights here also
wt.brks <- c(0, 44, 90, Inf)

neph.dat3$cmesh.70mm_Total.Weight <- cut(neph.dat3$mesh.70mm_Total.Weight, breaks = wt.brks
neph.dat3$cmesh.80mm_Total.Weight <- cut(neph.dat3$mesh.80mm_Total.Weight, breaks = wt.brks
neph.dat3$cmesh.90mm_Total.Weight <- cut(neph.dat3$mesh.90mm_Total.Weight, breaks = wt.brks
neph.dat3$cmesh.100mm_Total.Weight <- cut(neph.dat3$mesh.100mm_Total.Weight, breaks = wt.br

mnom.length.bulkc <- multinom(as.matrix(neph.dat3[, c("70mm_COUNT", "80mm_COUNT", "90mm_COU
                                prop.Carapace.Length +
                                prop.Carapace.Length2 +
                                prop.Carapace.Length3 +
                                ##
                                cmesh.70mm_Total.Weight +
                                cmesh.80mm_Total.Weight +
                                cmesh.90mm_Total.Weight +
                                cmesh.100mm_Total.Weight,
                                ##offset(offset.mat),
                                data = neph.dat3)

## # weights:  52 (36 variable)
## initial  value 65846.209564
## iter  10 value 65654.262307
## iter  20 value 65557.651822
## iter  30 value 65464.946065
## iter  40 value 65424.314033
## iter  50 value 65416.861080
## final  value 65416.844867
## converged

AIC(mnom.length.matrix, mnom.length.bulk, mnom.length.bulk.order1, mnom.length.bulk.order2,

## Error in lapply(list(object, ...), ll):  object 'mnom.length.bulk.order1' not
found

## INCLUDE HIGHER-ORDER CARAPACE LENGTH
```

Get predictions for low, medium and high catch weights

```
## In the model we include weights by compartment
## in the predictions we fix the weights to be equal so we can see the mesh effects
## 'standardize for weight effects so we can investigate mesh effects'

## low.med.high.bulk <- quantile(weight.dat[weight.dat$Species == "Bulk",]$Total.Weight,
##                               p = c(0.1, 0.5, 0.9))
## low.med.high.bulk
## ## set high to 120kg
## low.med.high.bulk[3] <- 120

## get mean total weight per bin used for empirical
low.med.high.bulk <- with(neph.dat.bulk, tapply(Total.Weight, cTotal.Weight, mean))

## Error in with(neph.dat.bulk, tapply(Total.Weight, cTotal.Weight, mean)):  object
'neph.dat.bulk' not found

pred.df <- expand.grid(Carapace.Length = pred.length,
                       av.Total.Weight = low.med.high.bulk)

## Error in expand.grid(Carapace.Length = pred.length, av.Total.Weight = low.med.high.bulk)
object 'low.med.high.bulk' not found

pred.df$av.Total.Weight.scaled <- pred.df$av.Total.Weight- mean.wt

## Error in `$<-.data.frame`(`*tmp*`, "av.Total.Weight.scaled", value = numeric(0)):
replacement has 0 rows, data has 400

## pred.df$mesh.80mm_Total.Weight <- pred.df$mesh.70mm_Total.Weight
## pred.df$mesh.90mm_Total.Weight <- pred.df$mesh.70mm_Total.Weight
## pred.df$mesh.100mm_Total.Weight <- pred.df$mesh.70mm_Total.Weight

## pred.df$cmesh.70mm_Total.Weight <- cut(pred.df$mesh.70mm_Total.Weight, breaks = wt.brks,
## pred.df$cmesh.80mm_Total.Weight <- cut(pred.df$mesh.80mm_Total.Weight, breaks = wt.brks,
## pred.df$cmesh.90mm_Total.Weight <- cut(pred.df$mesh.90mm_Total.Weight, breaks = wt.brks,
## pred.df$cmesh.100mm_Total.Weight <- cut(pred.df$mesh.100mm_Total.Weight, breaks = wt.bri

pred.df$prop.Carapace.Length <- pred.df$Carapace.Length/max.length

## Error in `$<-.data.frame`(`*tmp*`, "prop.Carapace.Length", value = numeric(0)):
replacement has 0 rows, data has 400

pred.df$prop.Carapace.Length2 <- pred.df$prop.Carapace.Length^2

## Error in `$<-.data.frame`(`*tmp*`, "prop.Carapace.Length2", value = numeric(0)):
replacement has 0 rows, data has 400

pred.df$prop.Carapace.Length3 <- pred.df$prop.Carapace.Length^3
```

```
## Error in `$<-.data.frame`(`*tmp*`, "prop.Carapace.Length3", value = numeric(0)):
replacement has 0 rows, data has 400

## Need to re-define offset here!!
## Better to set offset as cbind in data frames
##offset.mat <- matrix(0, ncol = 4, nrow = dim(pred.df)[1])

##mnom.pred <- predict(mnom.length.bulk.order3, newdata = pred.df, type = "prob")
mnom.pred <- predict(mnom.length.bulk, newdata = pred.df, type = "prob")

## Error in poly(av.Total.Weight.scaled, 2, coefs = structure(list(alpha = c(2.599200942798
:  object 'av.Total.Weight.scaled' not found

m <- dim(mnom.pred)[1]

## Error in eval(expr, envir, enclos):  object 'mnom.pred' not found

mnom.pred.df <- data.frame(
                  Mesh.Size = factor(rep(c("70mm", "80mm", "90mm", "100mm"),
                    each = m), levels = c("70mm", "80mm", "90mm", "100mm")),
                  Carapace.Length = rep(pred.df$Carapace.Length, times = 4),
                  prop.Carapace.Length = rep(pred.df$prop.Carapace.Length, times = 4),
                  av.Total.Weight = rep(pred.df$av.Total.Weight, times = 4),
                  proportion = c(mnom.pred))

## Error in data.frame(Mesh.Size = factor(rep(c("70mm", "80mm", "90mm", "100mm"),
:  object 'mnom.pred' not found

mnom.pred.df$Bulk.Weight <-
  ifelse(mnom.pred.df$av.Total.Weight == low.med.high.bulk[1], "Low (26kg)",
         ifelse(mnom.pred.df$av.Total.Weight == low.med.high.bulk[2], "Medium (60.5kg)", "H
         )

## Error in ifelse(mnom.pred.df$av.Total.Weight == low.med.high.bulk[1], :  object
'mnom.pred.df' not found

mnom.pred.df$Bulk.Weight <- factor(mnom.pred.df$Bulk.Weight,
                                   levels = c("Low (26kg)", "Medium (60.5kg)", "High (120kg

## Error in factor(mnom.pred.df$Bulk.Weight, levels = c("Low (26kg)", "Medium (60.5kg)",
:  object 'mnom.pred.df' not found

p + geom_line(data = mnom.pred.df,
              aes(x = Carapace.Length, y = proportion,
                  group = Bulk.Weight,
                  colour = Bulk.Weight)) +
  scale_colour_manual(values=c("#619CFF", "#00BA38", "#FF0000")) +
##scale_colour_manual(values = c("blue", "green", "red")) +
geom_hline(aes(yintercept = 0.25), linetype = "dashed") +
  geom_vline(aes(xintercept = 25), linetype = "dashed")

## Error in do.call("layer", list(mapping = mapping, data = data, stat = stat, :
object 'mnom.pred.df' not found
```

```
## draw empirical lines
## bin the weights
neph.dat.bulk <-  merge(neph.dat, weight.dat)

neph.dat.bulk$cTotal.Weight <- cut(neph.dat.bulk$Total.Weight, breaks = wt.brks)
##neph.dat.bulk$cTotal.Weight <- 1

## RAISING NOT INCLUDED UNTIL OFFSET INCLUDED
##neph.dat.bulk$rCOUNT <- with(neph.dat.bulk, COUNT/SUBSRATIO)
neph.dat.bulk$rCOUNT <- with(neph.dat.bulk, COUNT)

count.array <- with(neph.dat.bulk, tapply(rCOUNT, list(cTotal.Weight, Carapace.Length, Mesh

count.array[is.na(count.array)] <- 0

prop.array <- prop.table(count.array, c(1,2))

prop.df <- expand.grid(Total.Weight = dimnames(prop.array)[[1]],
                       Carapace.Length = dimnames(prop.array)[[2]],
                       Mesh.Size = dimnames(prop.array)[[3]],
                       stringsAsFactors = FALSE)

prop.df$empirical.prop <- NA
prop.df$rCOUNT <- NA

for(i in 1:dim(prop.df)[1]){
  ## proportion
  prop.df$empirical.prop[i] <- prop.array[prop.df$Total.Weight[i],
                                          prop.df$Carapace.Length[i],
                                          prop.df$Mesh.Size[i]
                                          ]
  ## count
  prop.df$rCOUNT[i] <- count.array[prop.df$Total.Weight[i],
                                   prop.df$Carapace.Length[i],
                                   prop.df$Mesh.Size[i]
                                   ]
}

prop.df$Bulk.Weight <- ifelse(prop.df$Total.Weight == "(0,44]", "Low (26kg)",
                              ifelse(prop.df$Total.Weight == "(44,90]", "Medium (60.5kg)",
                              )

prop.df$Carapace.Length <- as.numeric(prop.df$Carapace.Length)

prop.df$Mesh.Size <- factor(prop.df$Mesh.Size, levels = c("70mm", "80mm", "90mm","100mm"))

## empirical versus modelled

##pdf("../tex/figures/")
ggplot(prop.df, aes(x = Carapace.Length, y = empirical.prop,
                group = Bulk.Weight,
```

```
                colour = Bulk.Weight)) +
  geom_point(aes(size = log(rCOUNT)), linetype = "dashed", alpha = 0.5) +
  facet_wrap(~ Mesh.Size) +
  geom_line(data = mnom.pred.df, aes(x = Carapace.Length, y = proportion,
              group = Bulk.Weight,
              colour = Bulk.Weight)) +
  geom_hline(aes(yintercept = 0.25), linetype = "dashed") +
  geom_vline(aes(xintercept = 25), linetype = "dashed")
```

```
## Error in do.call("layer", list(mapping = mapping, data = data, stat = stat, :
object 'mnom.pred.df' not found
```

References

Holst, R. and Revill, A. 2009. A simple statistical method for catch comparison studies. *Fisheries Research* **95**(2–3): 254 – 259