EPIF project end of year research report: 2015

Cóilín Minto, Brian Burke (GMIT), Ronán Cosgrove (BIM)

This report contains a brief summary of research activity under the project "Implications of Alternative Technical Measures on the Economic Performance of the Irish Fishing Fleet (EPIF)", from April 7th – December 31st 2015. The purpose is to document the analyses conducted with reference to the original and updated project proposal. Plans for research and analytical support activities for 2016 are also outlined.

Table of Contents

Work package developments 2015

WP1a – Meta-analysis

WP1a proposed a meta-analytical framework for identifying gaps in gear-trial design (mainly for whitefish). Gaps analysis here refers to identification of gaps in the experimental design rather than gaps in the statistical methods (WP1b). The method used was straightforward gridding of parameter estimates from previously conducted trials (Figure 1). Discussion of these trials and present industry needs highlighted that there was a gap in the design for the 70mm cod-end mesh (Figure 1). No formal meta-analyses were conducted here though these are possible depending on needs.

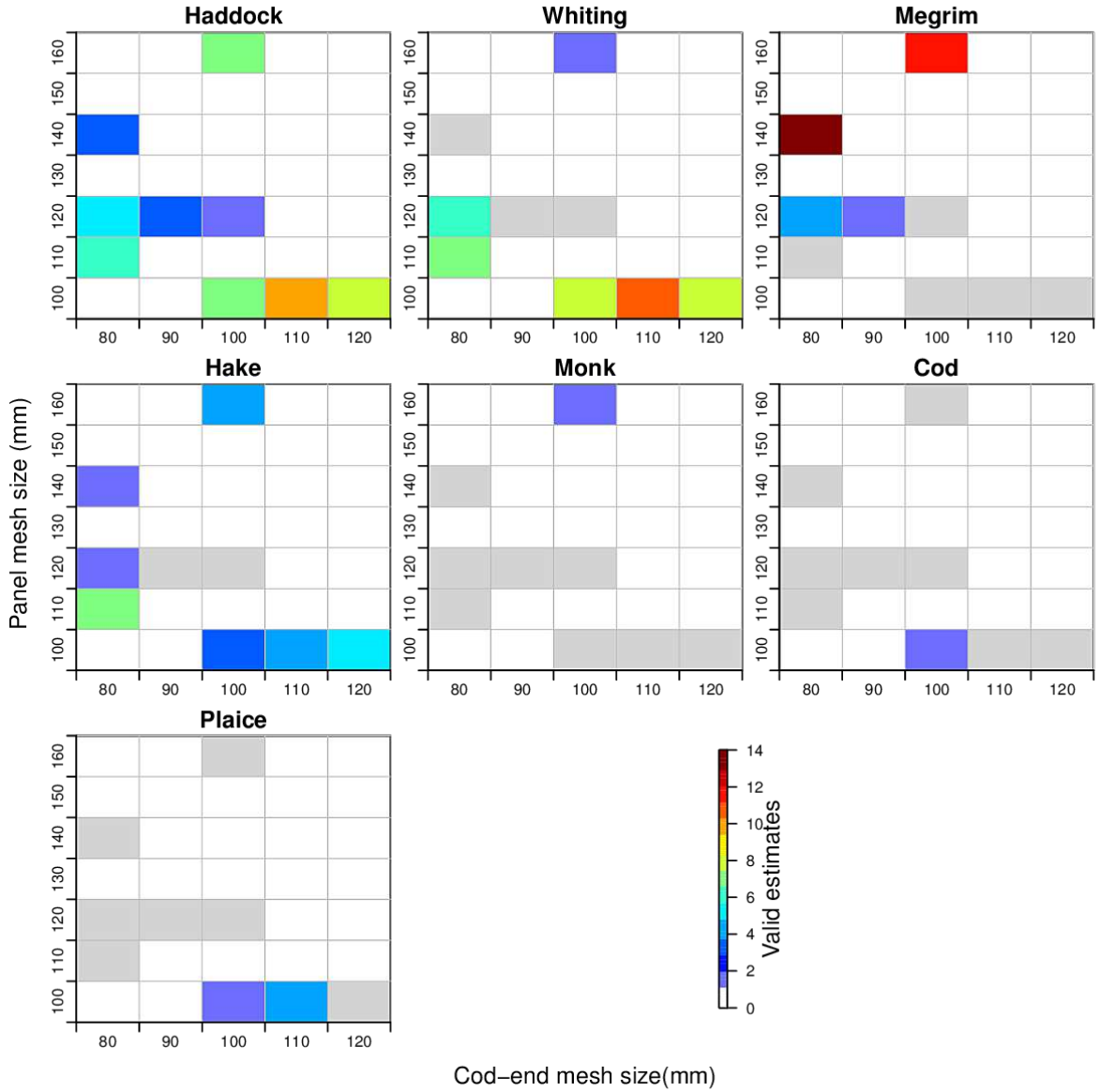


Figure 1. Gap analysis used for discussion on whitefish trials. Each grid cell represents a combination of a cod-end mesh size and panel mesh size trialled in selectivity experiments. The colour indicates the number of L50 parameters estimated for that combination. Grey indicates a trial conducted but insufficient data to estimate the parameters.

WP1b – Technical Measure Experimental design and data analysis

Technical Measure Experimental design

A simulation model for abundance-at-length and fishing of *Nephrops* in the Western Irish Sea was developed for the specific purpose of assisting BIM gear technologists in planning and executing gear trial experiments. As requested by BIM gear technologists, the first simulation focussed on what influence net position may have on retention of *Nephrops*. This is an important question as strong position effects may mask design variable effects. Simulation code was developed to address this question and comprised three main elements

1) Abundance simulation
    - Length distributions on the grounds, based on previous trials
    - Burrow density using underwater camera survey density estimates
2) Gear contacts
    - Based on the swept area of a tow
    - Assumed probability of emergence
3) Fishing Gear Simulation
    - Simplified estimation of how the gear may affect catch composition
    - Estimate the number of *Nephrops* contacting the net per hour
    - Simulating the Relative fishing intensity of each of the gears with realistic difficulties such as door and net effects included
    - Simulation of retention selectivity

A report on this investigation (Appendix I) highlighted the importance of net rotation to mitigate for position effects. This resulted in a development in trial design whereby the nets are rotated each night. This enhanced understanding was brought about by rounds of detailed collaboration between the gear technologists and the modelling team.

It is envisaged that further development of the power analysis simulation framework will assist in the design and implementation of additional trials during the course of the EPIF project.

Data analysis

Work Package 1b focuses on the development of modelling frameworks for the analysis of gear trials carried out by an Bord Iascaigh Mhara (BIM).

Gear trials conducted by BIM during the year focused on the quad-rig trawl, which is being utilised in the *Nephrops* fisheries - one of the most commercially important fisheries in Ireland. Usage of this new gear along with the implementation of the landing obligation for this fishery necessitated evaluation of the catch of this gear so that improvements in management strategy could be made.

Discussions with gear technologists at BIM indicated a lack of suitable methods for analysing catch composition of this gear. We addressed this gap by extending the method of Holst and Revill (2009) for twin-rig trawls to two or more cod-end comparisons. Initial model developments helped to

identify necessary explanatory variables and model type (GLMM as opposed to GAMM). Also from the initial models variability which had not been taken into account was identified.

By working closely with gear technologists and through the completion of preliminary analysis of trials (Appendix II, III, IV) a number of important factors were identified for inclusion into the analysis such as; the inclusion of net rotation to account for net position effects and the importance of bulk weights in the cod-end per haul. Catch comparison analysis of the quad rig data using the multinomial mixed effects model provided an understanding of how much variability they accounted for in the measured counts at length. The culmination of this work has provided a new statistical framework for analysing gear with 2 or more alternatives and has also helped in the identification of variables of importance for best fitting the model for these types of gears.

This work resulted in the writing and submission of a paper to the *ICES Journal of Marine Science*:

Browne, D., Minto, C., Cosgrove, R., Burke, B., McDonald, D., Officer, R. and Keatinge, M (*in review*). A general catch comparison method for multi-gear trials: application to a quad-rig trawling fishery for *Nephrops*.

We also assisted with data analysis for the following BIM report which has formed the basis of a proposed increase in minimum cod-end mesh size from to 70 to 80mm in Ireland's demersal fisheries:

Cosgrove, R., Browne, D., McDonald, D., Curtin, R., and Keatinge, M., 2015. Assessment of an increase in cod-end mesh size in the Irish Sea Nephrops fishery. Irish Sea Fisheries Board (BIM), Marine Technial report, 16 pp.

Contributions to this report are provided in Appendix V.

Capacity building

BIM staff attended an applied course in R and statistical modelling at GMIT September 28th – October 2nd and October 12th – 16th 2015.

Work plan 2016

To reflect and adapt to updated requirements, the project plan for 2016 departs from the original project proposal. Work plan 2016 is prioritised here with additional details given below. The priorities for 2016 are:

1. Quota optimisation
2. Gear technology
3. Albacore indices contributing to ICCAT assessments
4. Price analysis

Quota optimisation

In place of packages 3 and 4, GMIT will assist in the development of an alternative fishery based quota allocation system. The system will seek to optimise quota shares to groups of vessels based on the main species they target ensuring the continued viability of fisheries which are heavily impacted by the landing obligation such as the whitefish mixed demersal fishery.

Gear technology

GMIT will continue to work on the design and analysis of gear trials with BIM gear technologists as required during 2016. Work in 2016 will continue to focus on methods to reduce unwanted catches in *Nephrops* and whitefish fisheries. Building on work carried out in 2015, a bioeconomic analysis of the effects of sorting grids on catches of *Nephrops* and whitefish will be carried out. GMIT will also assist in designing and analysing data from a major assessment of the catch composition in single, twin and quad-rigged trawls in a *Nephrops* fishery.

Albacore tuna analyses

This work will be a collaboration between experts at BIM and analysts at GMIT to assist in the production of indices for ICCAT assessment of North Atlantic albacore tuna. Methods developed in previous collaborations will be extended temporally and new research on fleet dynamics contributing to the index will be explored by a PhD student at GMIT.

Price Analysis

An imperative to understanding the implications of gear changes on the economic performance of fleet segments is to understand price dynamics. We will focus on developing a framework for price analysis at multiple levels.

Prices of fish may vary based on a multitude of drivers, including: supply (local and international), season, size/grade, condition, species, product differentiation, processing, etc. The goal of the EPIF price analysis component of WP 4 is to investigate the influence of these factors on the price obtained at first point of sale by select segments in the Irish fleet, as identified by BIM. All methods will be trialled and discussed in close collaboration with BIM staff.

Univariate (single species) price analysis

Here we can focus on explaining variability in observed prices per species. Methods which can be explored for analysing univariate data would include regression approaches such as price elasticity models and time series methods such as autoregressive models (AR), moving average models (MA) or some combination of these (such as ARMA, or ARIMA models). These can be used for analysing fish price dynamics and trends, taking the temporal aspect into account when fitting.

Regression Model:

- For example regressing price on quantity, quality, location, etc. This model can be used to derive varying price elasticities.

Autoregressive Model:

- Used to describe a random process, the model assumes that the output variable depends, linearly, on the previous values of the output variable and a stochastic error term.

Moving Average Model:

- Also used to describe a random process, the model assumes that the output variable depends, linearly, on current and previous values of the stochastic error term.

Autoregressive Moving Average Models (ARMA):

- Comprised of both autoregressive and moving average components.

Multivariate (multiple species) price analysis

These analyses will focus on inter-relationships of prices among species and possibly locations (e.g., ports). Methods to be investigated include:

- Multivariate versions of the time series analysis highlighted above.
- Seemingly Unrelated Regression:
  This model will be used in identifying relationship between different fish prices, identification of substitutes or inferior goods within fish markets for given fish.
  It could also be used to analysis the price at port level, is there a coastal bias (additional transportation costs?) or port size bias (larger ports give lower prices?) when landing catches.
  The Seemingly Unrelated Regression (SUR) model
  - Runs a number of individual equations, each having a dependent variable and either the same or different predictor variables
  - These equations are then generalized into a linear regression model
  - The covariance structure of the residuals of these equations are then able to be taken into account providing efficient estimates which is then analysed to determine if a relationship between the equations exist.
  - Determination of the exact estimators (such as using FGLS, GLS or OLS) for the best estimates will require investigation of the data in question.

Appendices

Please note appendix page numbers are appendix-specific.

Appendix I

Simulations to assist with *Nephrops* quad-rig trial designs

For discussion only

1 Summary

A simulation model for abundance-at-length and fishing of *Nephrops* in the Western Irish Sea is developed for the specific purpose of assisting in survey design. The abundance simulation is relatively straightforward, the fishing gear effects are more involved and should only be considered indicative at present. Given time constraints, issues may have arisen with the code so extreme caution is requested in interpreting these results. It is envisaged that the framework will form the basis of further trial designs.

2 Simulation setup

We follow the general model of Millar and Fryer (1999) to generate catches from three processes:

1. Abundance;
2. Relative fishing intensity;
3. Contact-selection.

2.1 Abundance-at-length of *Nephrops*

The purpose of this section is to simulate a true underlying population of *Nephrops* in terms of abundance-at-length.
As the ageing of *Nephrops* is unclear at the moment we assume that there are four components (possibly age classes) present. This can be changed subsequently. First set the population lengths for each component

```r
## median carapace lengths for 4 components
mu.l <- c(10, 15, 20, 25)

## cv of carapace length for each component
cv.l <- 0.3

## proportion of the population composed of given component
p <- c(0.4, 0.3, 0.2, 0.1)
```

Plot the distribution of simulated lengths on the grounds (Figure 1).

```r
## vector of lengths
length.vec <- 0:60

## densities of each component
d1 <- p[1] * dlnorm(length.vec, log(mu.l[1]), sdlog = cv.l)
d2 <- p[2] * dlnorm(length.vec, log(mu.l[2]), sdlog = cv.l)
d3 <- p[3] * dlnorm(length.vec, log(mu.l[3]), sdlog = cv.l)
d4 <- p[4] * dlnorm(length.vec, log(mu.l[4]), sdlog = cv.l)

d <- rbind(d1, d2, d3, d4)

dsum <- apply(d, 2, sum)

## plot
plot(length.vec, dsum, type = "n", xlab = "Carapace length (mm)",
     ylab = "Distribution density", bty = "L")

matlines(length.vec, t(d), col = "purple", lty = 1)

lines(length.vec, apply(d, 2, sum), col = "forestgreen", lwd = 1.5)

legend("topright", legend = c("Component (age) distribution", "Overall distribution"),
       lty = 1, lwd = c(1, 1.5), col = c("purple", "forestgreen"), bty = "n")
```

We first assume that the local length distribution (on the grounds) is the same as the population length distribution (Western Irish Sea area). To convert the length distribution to the numbers at length n_l we approximate the overall distribution (Figure 1) by one mm bins, the proportion of the total area occupied by each bin is then used to generate the population counts by length bin. First the proportion of the population per length bin is simulated

```r
## approximate the area under the overall distribution
library(flux)
midpoints <- (length.vec[-length(length.vec)] + length.vec[-1])/2

nm <- length(midpoints)

## proportion of population per length class
prop.l <- rep(NA, nm)
```

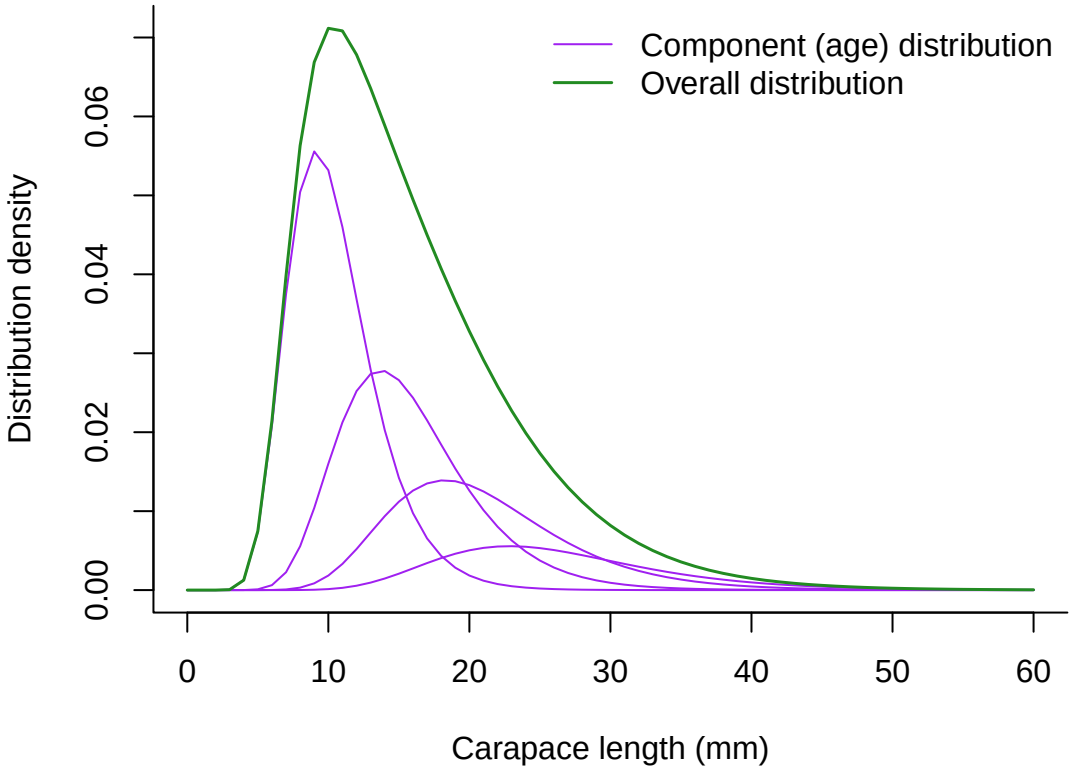


Figure 1: Simulated length distribution of the population of *Nephrops* on the grounds.

```
for(i in 1:(length(length.vec) - 1)){
  prop.l[i] <- auc(x = length.vec[i:(i + 1)], y = dsum[i:(i + 1)])
}

sum(prop.l)


## [1] 0.9997978


## close to 1, difference due to approximation
```

The proportion of the population per length bin is then multiplied by the estimated abundance of 4.6×10^9 burrows (Ligas et al., 2014) to obtain the distribution of numbers at length (Figure 2).

```
n <- 4.6e9

nl <- prop.l * n

barplot(nl, space = 0, col = "lightgrey", ylab = "Abundance (number of burrows)",
        xlab = "Length-class (cm)")

axis(side = 1, at = seq(0, 60, by = 10))
```

It may be necessary to alter the length distribution and abundance spatially to truly reflect the population on the grounds.

2.2 Contacts with gear

The footline length on the Celtic Warrior II trials on the Smalls was 38m - assuming the footline takes on a semi-circle during fishing (revise to catenary or similar - think about how this influences the face of each rig also) implies $2\pi r = 2 \times 38m$ and so the radius $r = 38m/\pi$ and the diameter (fishing contact width) is $2 \times 38m/\pi \approx 24.2m$. Based on a towing speed of 3 knots ($\approx$ 5.6km.h^{-1}), the area covered per hour is assumed to be 5.6km.h^{-1} $\times$ 0.0242km $\approx$ 0.14km^2.h^{-1}.

```
fishing.width <- 0.0242 ## km

speed <- 5.6 ## km/h

area.ph <- round(fishing.width * speed, 2) ## km^2/hour
```

The adjusted mean density is estimated as 0.83 burrows.m^{-2} (Ligas et al., 2014). Therefore, the simulated number of burrows covered per hour is 0.14km^2.h^{-1} $\times$ 8.3×10^5km^{-2} $= 1.162 \times 10^5$.h^{-1}.

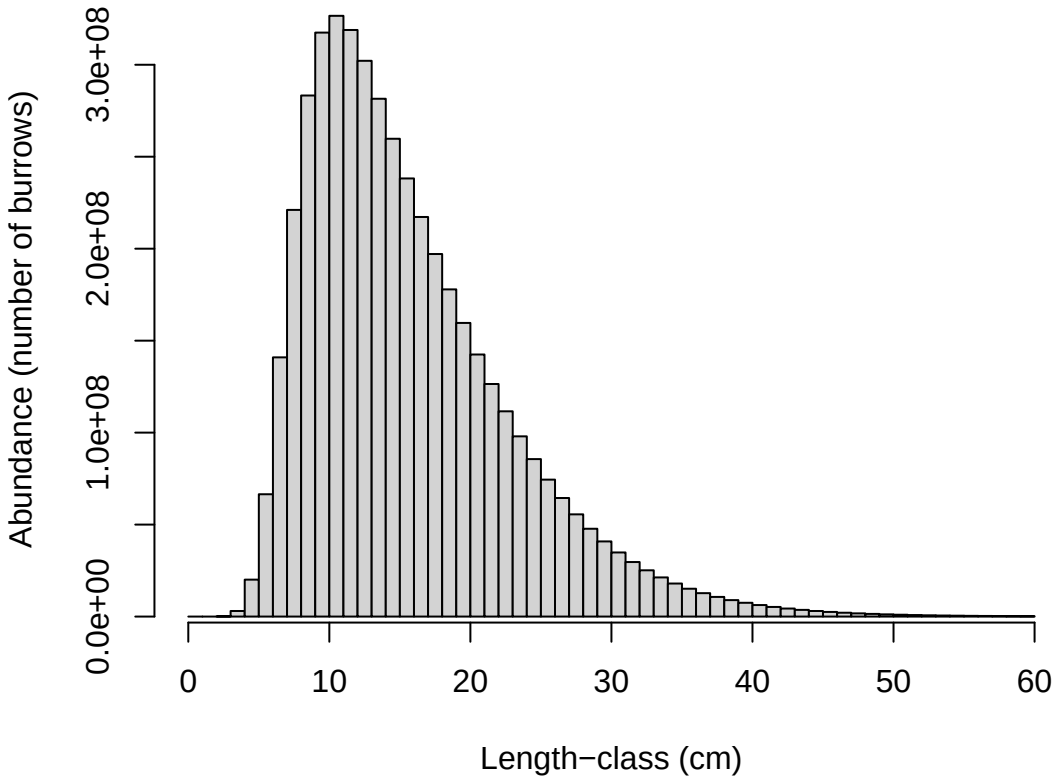


Figure 2: Simulated abundance-at-length of *Nephrops* on the grounds.

```
burrow.dens <- 0.83 * 1e6

(burrows.ph <- area.ph * burrow.dens)

## [1] 116200
```

Assuming a proportion of the *Nephrops* in the burrows covered will come out of the burrows and contact the net (revise), we can simulate the numbers contacting the net per hour.

```
contact.prop <- 0.2

hours.towed <- 1

(n.contact <- contact.prop * hours.towed * burrows.ph)

## [1] 23240
```

This can then be converted to the numbers at length contacting the net per hour, assuming the probability of contact is equal across length classes.

```
nl.contact <- n.contact/n * nl
```

We now have a very basic model linear model for generating *Nephrops* contacts with the net on the basis of the numbers of hours towed and density of burrows on the grounds. This could be considerably improved with further discussion and refinement but is assumed correct in the simulations below.

2.3 Relative fishing intensity

As in (Millar and Fryer, 1999), the relative fishing intensity $p_j(l)$ is the probability that a fish of length l contacts gear j given that it has contacted the entire net. In a quad-rig, this is the probability that a *Nephrops* entering the entire net contacts each of the rigs. The simplest assumption is that the relative fishing intensity is length-independent and equal across the gears $p_j(l) = 1/4$ (for quad-rig). But it is possible that asymmetries (e.g., door functioning), hydrodynamics, etc. make the probabilities of contact unequal across the nets. We simulate three relative fishing intensity scenarios, assuming length-independence of each:

1. Equal relative fishing intensity $p_j(l) = 1/4$;
2. Single net effect: relative fishing intensity of the second net (second in from port side) fishing twice as well as the others ($p_2(l) = 2/5, p_{1,3,4} = 1/5$).
3. Simulated door effect: relative fishing intensity increasing from port to starboard with ($p_1(l) = 1/10, p_2 = 2/10, p_3 = 3/10, p_4 = 4/10$). Note that the relative fishing intensity is drastic

with the outermost starboard having four times the fishing intensity of the outermost port. This is unlikely in reality but is used in the simulations to exemplify the effects.

```
## relative fishing intensity
## scenario 1
p.rfi.1 <- rep(1/4, 4)

## scenario 2
p.rfi.2 <- c(1/5, 2/5, rep(1/5, 2))

## scenario 3
p.rfi.3 <- c(1/10, 2/10, 3/10, 4/10)
```

2.4 Simulated contact selection $r_j(l)$

We focus on a much-simplified analysis of the retention of the proportion of *Nephrops* above and below the MLS retained in each cod-end. Given the priority of the cod-end we focus on that here but will add in grid selectivity on (Monday).

2.4.1 Cod-end retention

From trials conducted in the Danish fishery over two years, using commercial 90mm diamond mesh codend with 5mm double twine, Frandsen et al. (2010) estimated $L_{50,2006} = 16.71$, $L_{50,2007} = 18.35$ and $SR_{2006} = 14.71, SR_{2007} = 11.73$. We take an average of these parameters for the selectivity of the 90mm cod-end.

```
l50.90mm <- (16.71 + 18.35) / 2

SR.90mm <- (14.71 + 11.73) / 2
```

For the other cod-ends we could not readily find information on the selectivity curves, we therefore assume that each 10mm change in the cod-end mesh size shifts the L_{50} by 5mm, keeping the selectivity range constant (Figure 3).

```
delta <- 5 ## difference per 10mm

l50.100mm <- l50.90mm + delta
l50.80mm <- l50.90mm - delta
l50.70mm <- l50.90mm - 2 * delta

## all
l50 <- c(l50.70mm, l50.80mm, l50.90mm, l50.100mm)
names(l50) <- c("70mm", "80mm", "90mm", "100mm")
```

```
## selectivity range
SR.100mm <- SR.80mm <- SR.70mm <- SR.90mm

## all
SR <- c(SR.70mm, SR.80mm, SR.90mm, SR.100mm)
names(SR) <- c("70mm", "80mm", "90mm", "100mm")
```

```
logist.sel <- function(x, l50, sr){
  plogis(2 * log(3) / sr * (x - l50))
}

curve(logist.sel(x = x, l50 = l50.70mm, sr = SR.70mm),
      from = 0, to = 60, xlab = "Carapace length (mm)",
      ylab = "Proportion retained in cod-end",
      bty = "L", ylim = c(0, 1))
curve(logist.sel(x = x, l50 = l50.80mm, sr = SR.80mm),
      lty = 2, add = TRUE)
curve(logist.sel(x = x, l50 = l50.90mm, sr = SR.90mm),
      lty = 3, add = TRUE)
curve(logist.sel(x = x, l50 = l50.100mm, sr = SR.100mm),
      lty = 4, add = TRUE)
legend("bottomright", legend = c("70mm", "80mm", "90mm", "100mm"),
       lty = 1:4, bty = "n")
```

```
## matrix of cod-end retention at length by gear
rl.mat <- cbind(
            logist.sel(x = midpoints, l50 = l50.70mm, sr = SR.70mm),
            logist.sel(x = midpoints, l50 = l50.80mm, sr = SR.80mm),
            logist.sel(x = midpoints, l50 = l50.90mm, sr = SR.90mm),
            logist.sel(x = midpoints, l50 = l50.100mm, sr = SR.100mm)
            )
colnames(rl.mat) <- c("70mm", "80mm", "90mm", "100mm")
```

We do not include weight effects in the simulation here although these could be included by making the parameters of the selectivity curves a function of bulk weight.

2.5 Net positions

In a quad-rig with with 4 cod-end mesh sizes, there are 24 possible configurations of the positioning of the nets. To see what configuration gives the maximal difference of mesh sizes between adjacent nets, we can use the permuations.

```
library(combinat)

perm.list <- permn(c(70, 80, 90, 100))
```

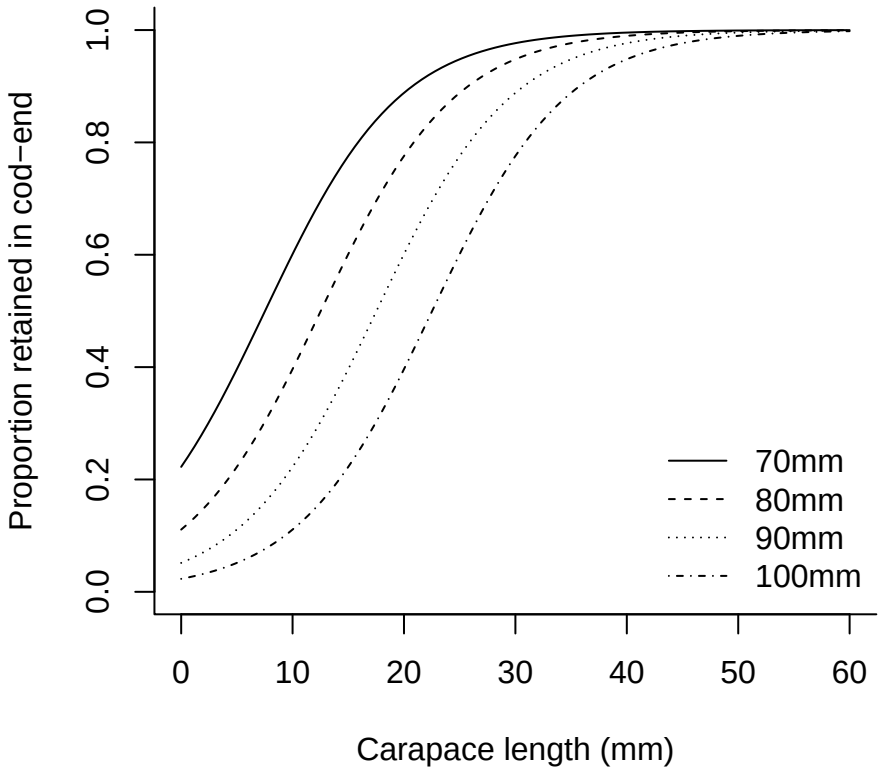


Figure 3: Simulated cod-end selectivity curves. Based on diamond-mesh sizes as indicated in the legend.

```
## calulate the differences between the mesh size
perm.diff <- unlist(lapply(perm.list, FUN = function(z){sum(abs(diff(z)))}))

perm.list[which.max(perm.diff)] ## reverse is also maximal difference


## [[1]]
## [1]  90  70 100  80
```

We use three net configurations in the simulations:

1. Adjacent: 70, 80, 90, 100;
2. Random: each haul random (completely impractical but useful for comparison);
3. Non-adjacent: 70, 90, 80, 100.

```
net.position.adj <- c("70mm", "80mm", "90mm", "100mm")

net.position.nadj <- c("70mm", "90mm", "80mm", "100mm")
```

3 Simulation

The numbers retained in the cod-end per length class l per haul h per gear j are simulated as (Millar and Fryer, 1999):

$$n_{l,j,h} \sim \text{Pois}(p_j \lambda_l r_j(l) e^{\varepsilon_h}) \tag{1}$$

where ε_j is a haul effect representing unmeasured variables influence on the counts (e.g., some hauls have greater or lower numbers). We assume that this haul effect has a CV of 20% (refine), corresponding to $\varepsilon_h \sim \text{N}(0, 0.2^2)$. Equation (1) gives us the ability to simulate catches per net by haul. We can write a function that produces the expected data per haul

```
sim.neph.data <- function(p.rfi, net.position, subs.ratio, CV.haul, hours.towed){
  ##----------------------------
  ## p.rfi is the relative fishing intensity with names "70mm","80mm","90mm","100mm"
  ## net.position is the order of the nets, e.g., c("70mm","80mm","90mm","100mm")
  ## subs.ratio is the proportion of the catch sampled
  ## CV.haul is the coefficient of variation of the haul effect
  ## hours.towed: number of hourse towed
  ##----------------------------
  ## numbers contacting the net
  (n.contact <- contact.prop * hours.towed * burrows.ph)
```

```
  ## numbers contacting by length
  lambda.l <- n.contact/n * nl
  ## numbers contacting each of the gears
  nlj <- matrix(lambda.l) %*% t(matrix(p.rfi))
  colnames(nlj) <- net.position
  ## cod-end retention in order of net position
  r.l <- rl.mat[, net.position]
  ## haul effect
  epsilon <- rnorm(1, 0, CV.haul)
  ##
  pois.mean <- nlj * r.l * exp(epsilon)
  ## numbers per net per length class
  nlh <- apply(pois.mean, 2, FUN = function(z){rpois(nm, lambda = z)})
  ## subsample
  for(i in 1:4){
    nlh.all <- rep(midpoints, times = nlh[,i])
    ## randomly choose from the total catch
    n.neph <- length(nlh.all)
    sample.idx <- sample(1:n.neph, size = round(n.neph/2), replace = FALSE)
    neph.subsample <- nlh.all[sample.idx]
    assign(paste("neph.subsample", colnames(nlh)[i], sep = "."),
           neph.subsample)
    rm(neph.subsample)
  }
  ##
  binned.70mm <- table(cut(neph.subsample.70mm, breaks = length.vec))
  binned.80mm <- table(cut(neph.subsample.80mm, breaks = length.vec))
  binned.90mm <- table(cut(neph.subsample.90mm, breaks = length.vec))
  binned.100mm <- table(cut(neph.subsample.100mm, breaks = length.vec))
  ##
  haul.df <- data.frame(length = midpoints,
                        "count.70mm" = as.numeric(binned.70mm),
                        "count.80mm" = as.numeric(binned.80mm),
                        "count.90mm" = as.numeric(binned.90mm),
                        "count.100mm" = as.numeric(binned.100mm)
                        )
  return(haul.df)
}
```

Test the function (Figure 4).

```
sim.haul <- sim.neph.data(p.rfi = p.rfi.1,
                          net.position = c("70mm", "80mm", "90mm", "100mm"),
                          subs.ratio = 0.2,
                          CV.haul = 0.2,
                          hours.towed = 2)

matplot(sim.haul[,1], sim.haul[, -1], pch = 1, xlab = "Carapace length (mm)",
        ylab = "Measured number", bty = "L")
legend("topright", legend = c("70mm", "80mm", "90mm", "100mm"),
       pch = 1, col = 1:4, bty = "n")
```

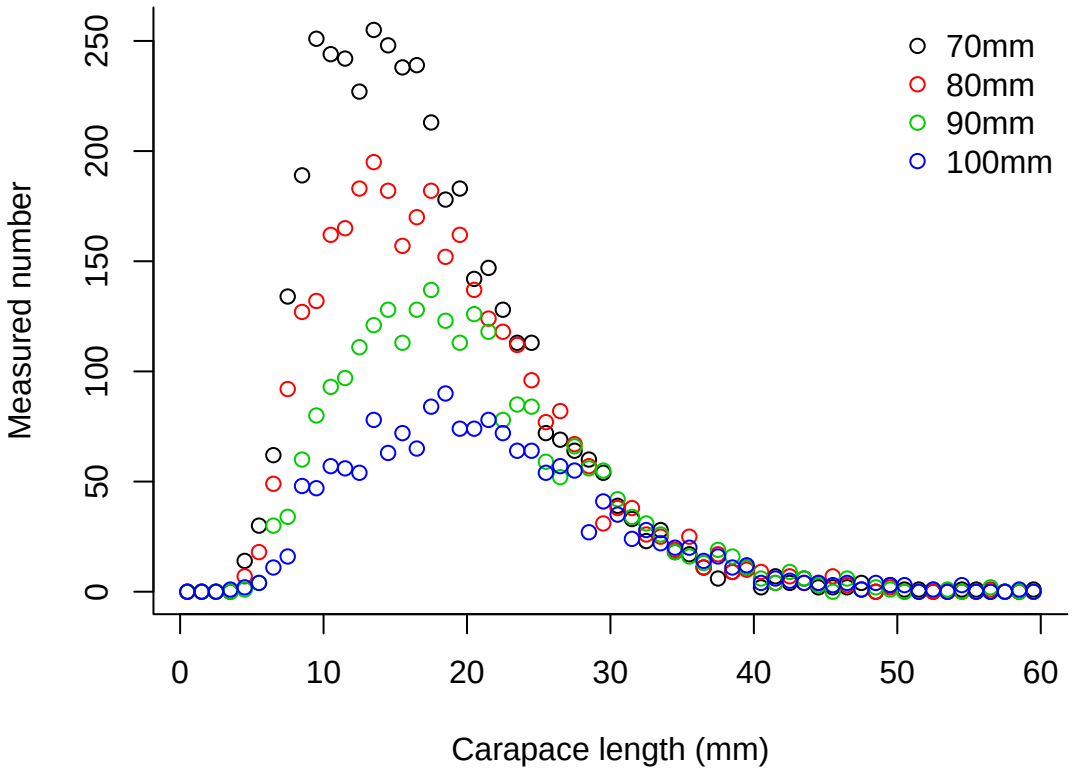


Figure 4: Simulated sample numbers per length class by mesh size assuming equal fishing intensity.

3.1 Simulation scenarios

For the simulation scenarios we define the power as the power to detect a significant ($\alpha = 0.05$) difference in the number caught below the minimum landing size (20mm) by gear with the direction of the estimated counts being $70mm > 80mm > 90mm > 100mm$.

Generate the combinations of simulation scenarios

```
sim.comb <- expand.grid(p.rfi = c("p.rfi.1","p.rfi.2","p.rfi.3"),
                        net.position = c("adjacent","random","non-adjacent"),
                        switch = c("no", "yes"),
                        stringsAsFactors = FALSE)

sim.comb$power <- NA
```

Run through the scenarios with 50 simulations per scenario

```
nsim <- 50

for(i in 1:dim(sim.comb)[1]){
  print(i)
  sim.correct <- rep(NA, nsim)
  ##
  for(j in 1:nsim){
    ## get the net position
    if(sim.comb$net.position[i] == "adjacent"){
      net.position <- net.position.adj
    }
      if(sim.comb$net.position[i] == "non-adjacent"){
        net.position <- net.position.nadj
      }
    if(sim.comb$net.position[i] == "random"){
      net.position <- sample(c("70mm", "80mm", "90mm", "100mm"), 4)
    }
    ##
    nhauls <- 10
    haul.list <- lapply(1:nhauls, function(x){
      if(sim.comb$switch[i] == "yes" & x > 5){
        net.position <- rev(net.position) ## rotate the nets
      }
      ##print(net.position)
      dat <- sim.neph.data(p.rfi = get(sim.comb$p.rfi[i]),
                           net.position = net.position,
                           subs.ratio = 0.2,
                           CV.haul = 0.2,
                           hours.towed = 2)
      dat$HAULID <- x
      return(dat)
    })
    haul.data <- do.call(rbind, haul.list)
    ## subset for those under MLS
    mls.data <- haul.data[haul.data$length < 20, ]
    ## sum the counts
    sum.mls.data <- aggregate(mls.data[, -1], by = list(mls.data$HAULID), sum)
    ## CHECK!! WHEN CHANGES
    sum.mls.data$HAULID <- sum.mls.data$HAULID/20
    ##
    sum.mls.data.long <- reshape(sum.mls.data, direction = "long",
                                 drop = c("Group.1"), varying = 2:5, timevar = "Mesh")
    ## net position
    sum.mls.data.long$net.pos <- match(sum.mls.data.long$Mesh, net.position)
    ##
    if(sim.comb$switch[i] == "yes"){
      sum.mls.data.long$net.pos[sum.mls.data.long$HAULID > 5] <-
        match(sum.mls.data.long$Mesh[sum.mls.data.long$HAULID > 5], rev(net.position))
      ##
      sum.mls.data.long$net.pos <- factor(paste("pos", sum.mls.data.long$net.pos,
```

```
                                                sep =""))
      ##
      ## run the GLMER with position effects
      mls.glmer <- glmer(count ~ -1 +  Mesh + (1|net.pos),
                         data = sum.mls.data.long, family = poisson)
      ## plot(unlist(ranef(mls.glmer)))
      null.glmer <- glmer(count ~ (1|net.pos), data = sum.mls.data.long,
                          family = poisson)
      ## significance
      (D <- - 2 * logLik(null.glmer) + 2 * logLik(mls.glmer))
      sig <- pchisq(D, lower.tail = FALSE, df = 3)
      ## coefficient values
      coef.hat <- fixef(mls.glmer)[c("Mesh70mm", "Mesh80mm", "Mesh90mm", "Mesh100mm")]
    }else{
      ## run the GLM without position effects
      mls.glm <- glm(count ~ -1 +  Mesh, data = sum.mls.data.long, family = poisson)
      ##
      null.glm <- glm(count ~ 1, data = sum.mls.data.long, family = poisson)
      ## significance
      (D <- - 2 * logLik(null.glm) + 2 * logLik(mls.glm))
      sig <- pchisq(D, lower.tail = FALSE, df = 3)
      ## coefficient values
      coef.hat <- coef(mls.glm)[c("Mesh70mm", "Mesh80mm", "Mesh90mm", "Mesh100mm")]
    }
    if(all(diff(coef.hat) < 0) & sig < 0.05){
      sim.correct[j] <- 1
    }else{
      sim.correct[j] <- 0
    }
  }
  ##
  sim.comb$power[i] <- sum(sim.correct)/nsim
}

save(list = c("sim.comb"), file = "sim_comb.RData")
```

```
load("sim_comb.RData")

sim.comb$gear.effect <- ifelse(sim.comb$p.rfi == "p.rfi.1",
                               "Equal fishing intensity",
                               ifelse(sim.comb$p.rfi == "p.rfi.2",
                                      "Single net effect",
                                      "Door effect")
                               )

sim.comb <- sim.comb[order(sim.comb$p.rfi, sim.comb$net.position), ]

library(xtable)
```

```
sim.comb.xtab <-
  xtable(sim.comb[, c("net.position", "gear.effect", "switch", "power")])

print(sim.comb.xtab, include.rownames = FALSE)
```

net.position	gear.effect	switch	power
adjacent	Equal fishing intensity	no	1.00
adjacent	Equal fishing intensity	yes	1.00
non-adjacent	Equal fishing intensity	no	1.00
non-adjacent	Equal fishing intensity	yes	1.00
random	Equal fishing intensity	no	1.00
random	Equal fishing intensity	yes	1.00
adjacent	Single net effect	no	0.00
adjacent	Single net effect	yes	0.02
non-adjacent	Single net effect	no	0.00
non-adjacent	Single net effect	yes	0.04
random	Single net effect	no	0.24
random	Single net effect	yes	0.74
adjacent	Door effect	no	0.00
adjacent	Door effect	yes	1.00
non-adjacent	Door effect	no	0.00
non-adjacent	Door effect	yes	1.00
random	Door effect	no	0.08
random	Door effect	yes	1.00

From these results we can see that:

- If the fishing intensity is constant across all the nets (all fishing the same), the power to detect meaningful differences in the numbers caught below the MLS is very high;
- If there is a door effect such that the fishing intensity/fishing power decreases or increases across the nets, the power to detect differences will be low without a switch; performing a switch and including the net position as a factor in the analysis results in much improved power to detect;
- The most difficult scenario to rectify with the current setup is a single net fishing considerably better than others. Switching does improve this in the random net location by haul but for practical switches it is difficult to isolate what is a gear effect and what is a net effect with a single switch (random effects are assumed symmetric). Additional switches would allow the net position to be included as a fixed effect in the analysis (here they've been necessarily included as a random effect), which may alleviate the issue. This has not been tested here.

References

Frandsen, R. P., Madsen, N. and Krag, L. A. 2010. Selectivity and escapement behaviour of five commercial fishery species in standard square- and diamond-mesh codends. *ICES Journal of Marine Science: Journal du Conseil* **67**(8): 1721–1731. doi:10.1093/icesjms/fsq050

Ligas, A., Doyle, J., Lordan, C., Brown, V., Doran, S., McArdle, J., McCausland, I., McCorriston, P., Simpson, S., and Schön, P.-J. 2014. Western Irish Sea Nephrops Grounds (FU15) 2014 UWTV Survey Report and catch options for 2015. *AFBI and Marine Institute UWTV Survey report* 21pp

Millar, R. B. and Fryer, R. J. 1999. Estimating the size-selection curves of towed gears, traps, nets and hooks. *Reviews in Fish Biology and Fisheries* **9**(1): 89–116

Appendix II

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")
```

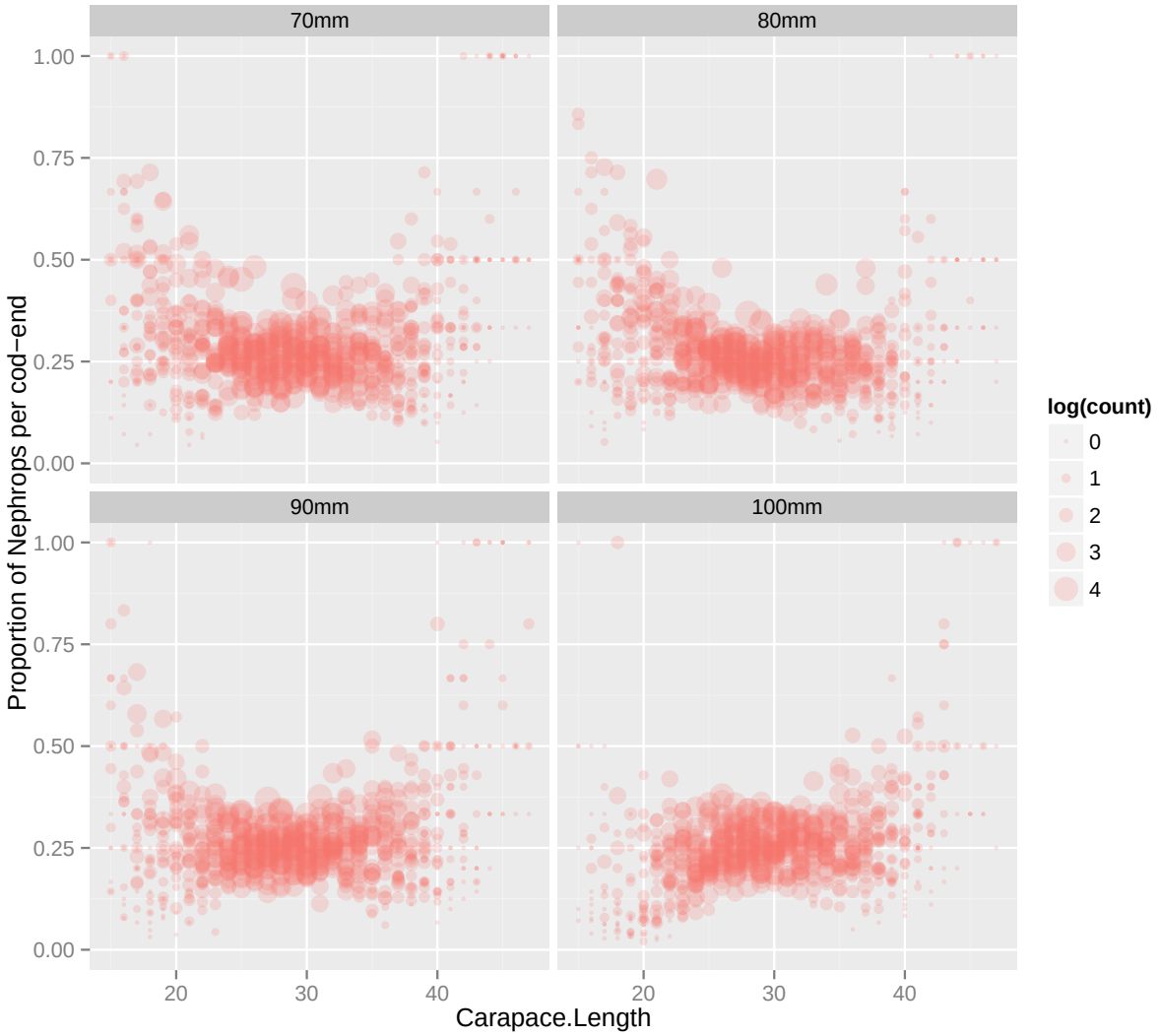


Figure 1: Proportion of Nephrops catch retained per haul. Each point represents the proportion of the Nephrops catch (in number) per haul and length class retained in a given cod-end (70mm, 80mm, 90mm, or 100mm). The size of the point is proportional to the log of the count.

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
```

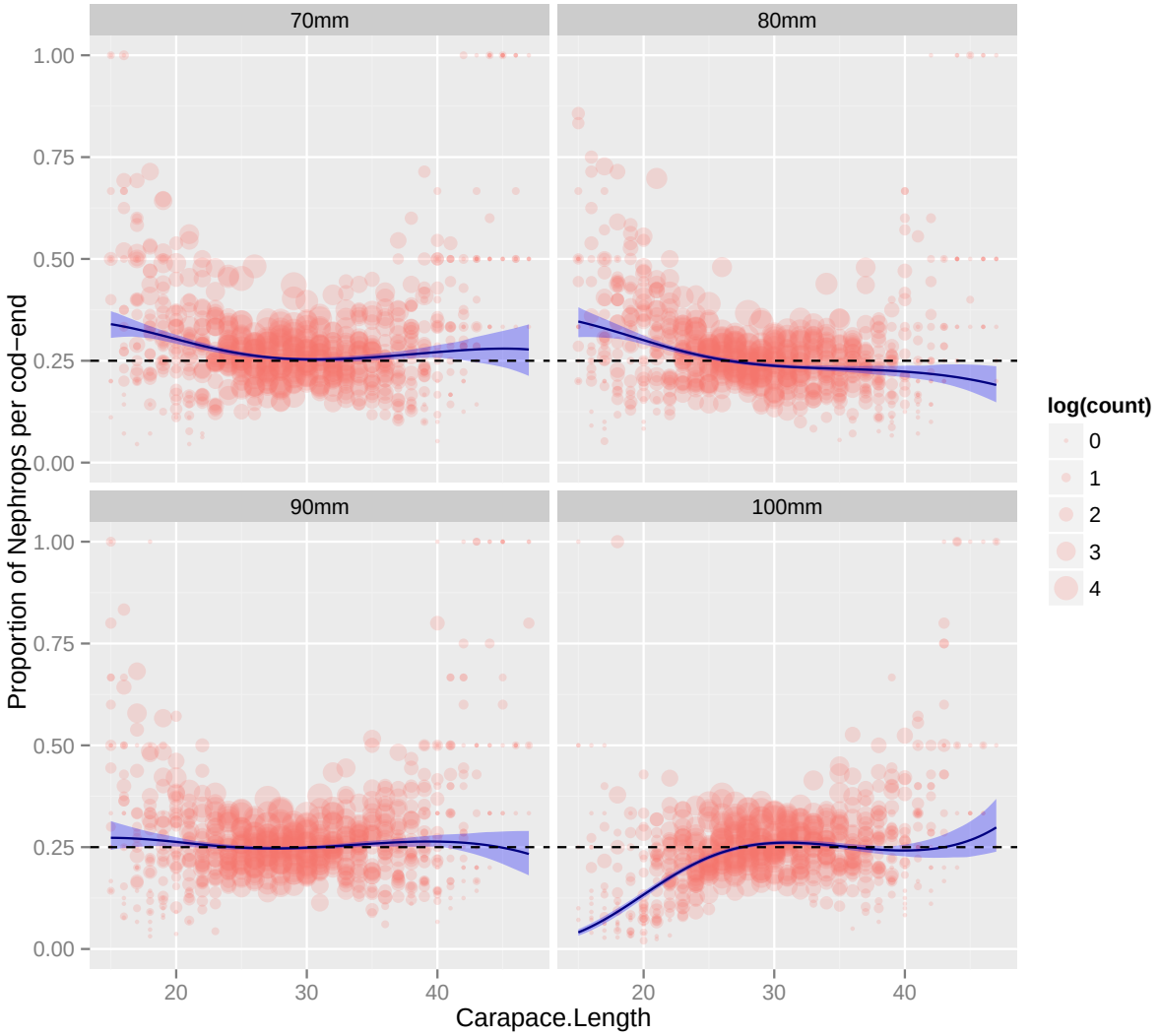


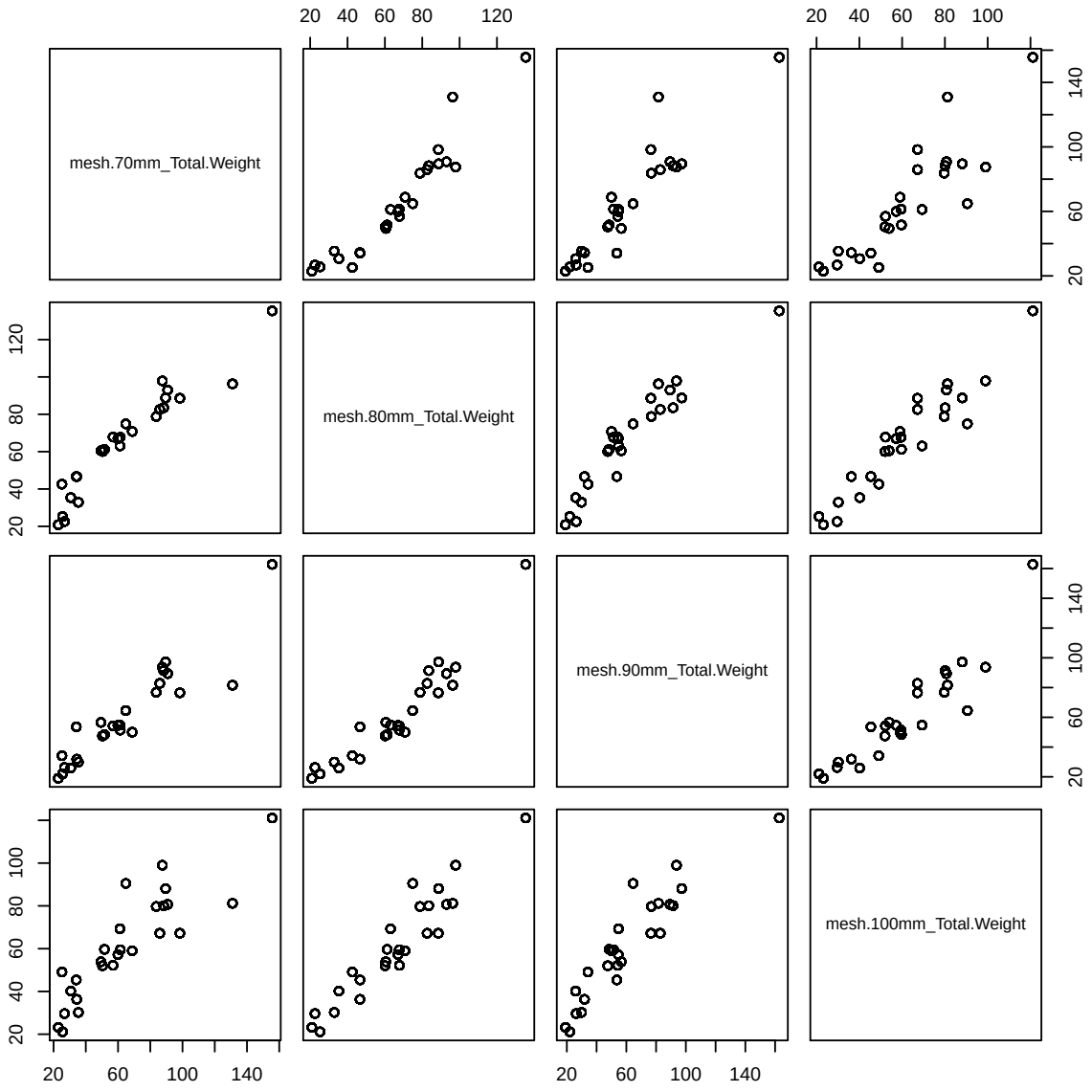
Figure 2: Proportion of Nephrops catch retained per haul with fitted multinomial model and associated re-sampled intervals. Null hypothesis of equal retention is displayed as the dashed line at 0.25.

```
## 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

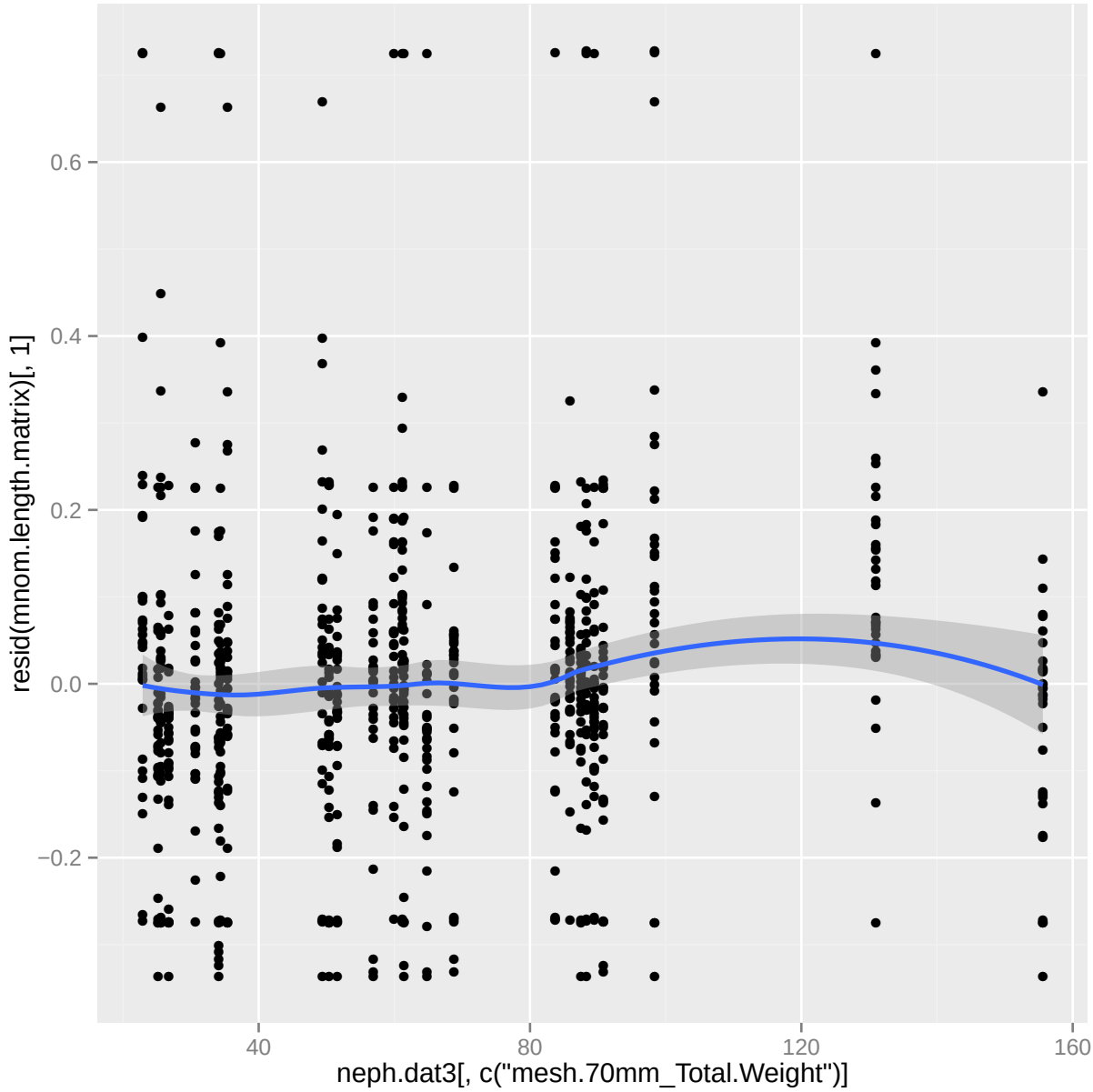
```
## compare two ways of writing same model
```

```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
```

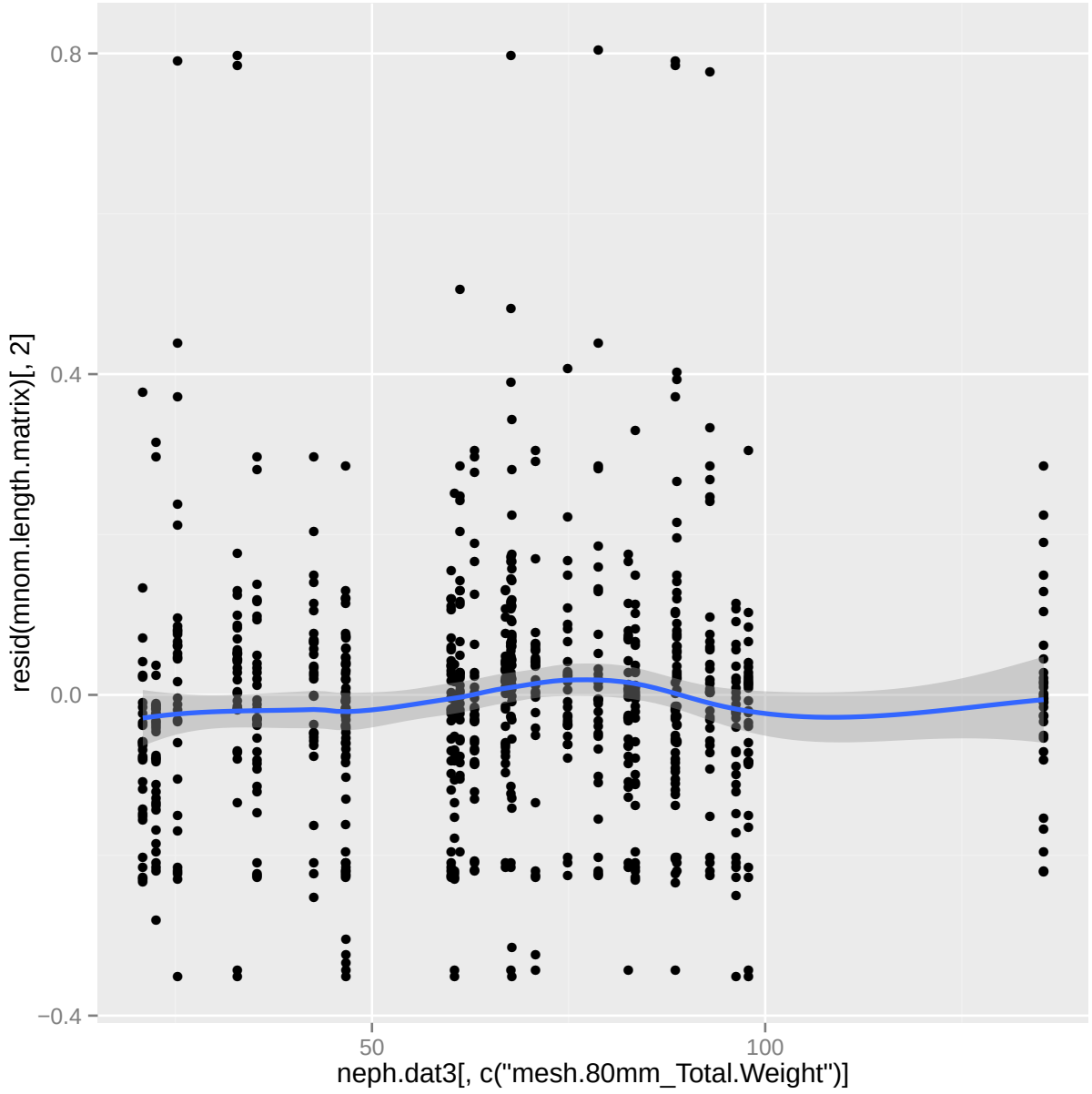



```
## very strong correlation in the counts
round(cor(neph.dat3[, c("mesh.70mm_Total.Weight", "mesh.80mm_Total.Weight", "mesh.90mm_Tota


##                          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
```

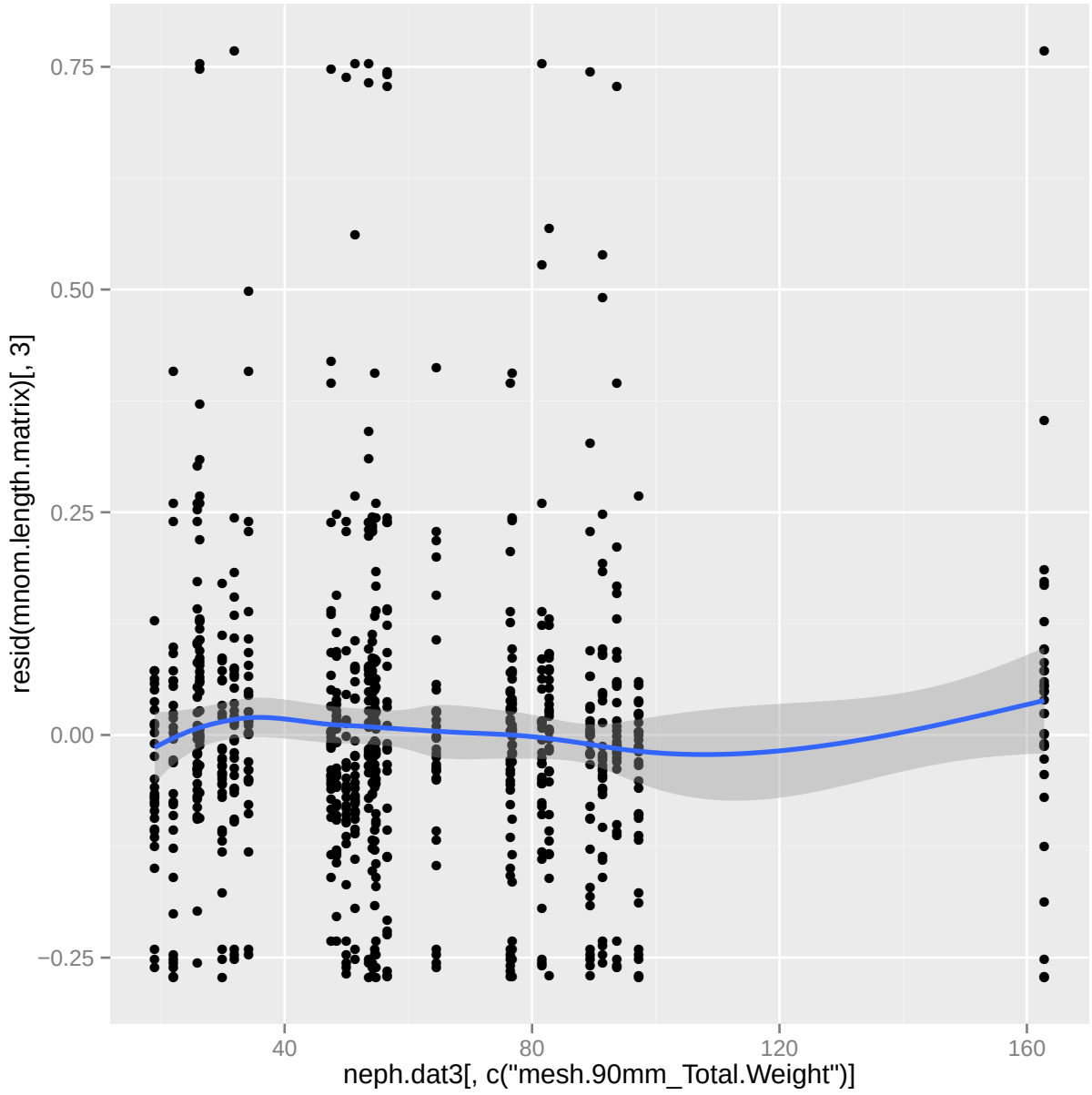
```
## mesh.90mm_Total.Weight                  1.000                  0.927
## mesh.100mm_Total.Weight                 0.927                  1.000


## 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.
```



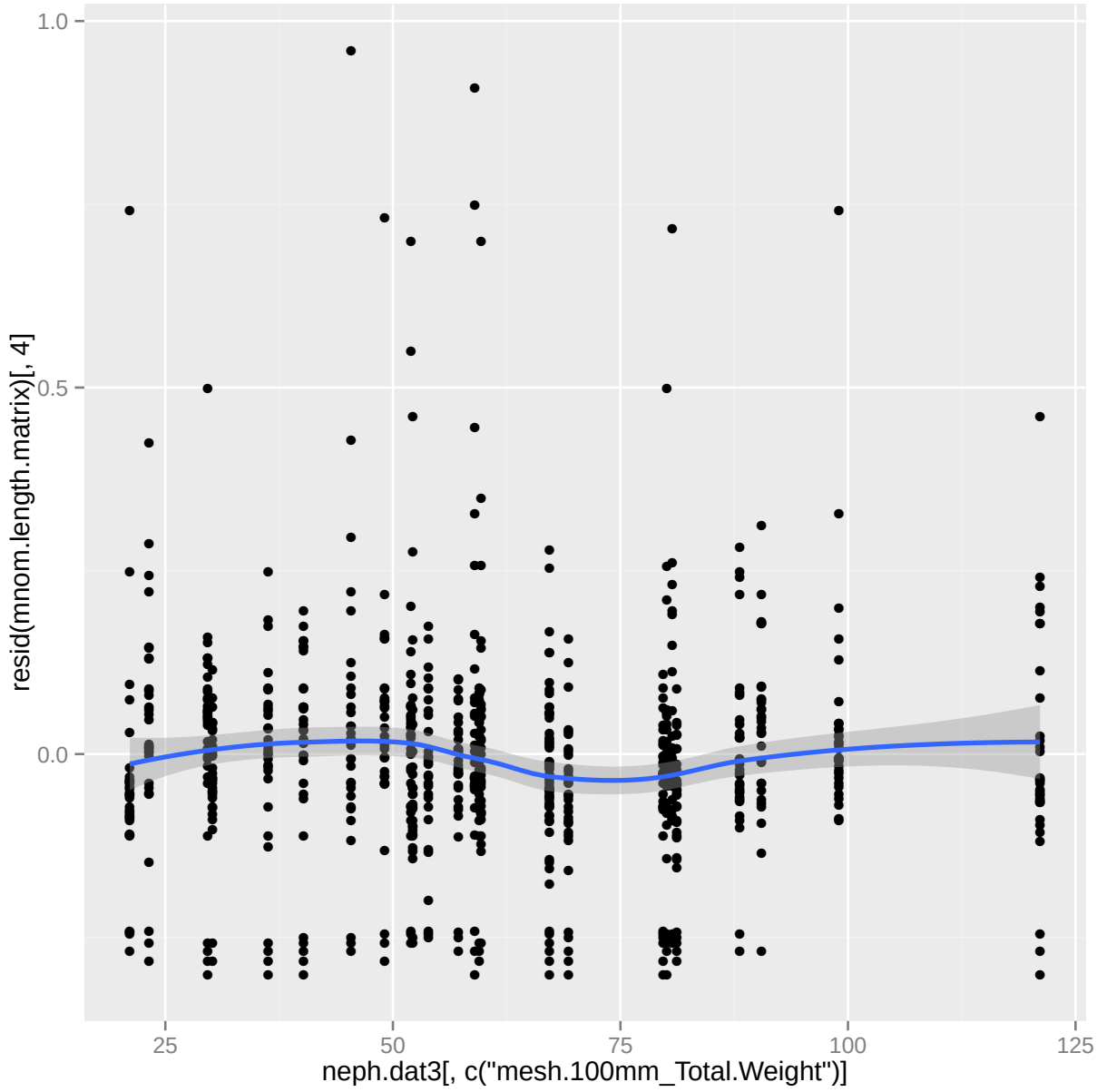
```
qplot(neph.dat3[, c("mesh.80mm_Total.Weight")], resid(mnom.length.matrix)[,2]) + geom_smoot

## 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

Appendix III

Preliminary multinomial analysis of *Arkh Angell* data

For discussion

1 Data

```
library(gdata)

arkh.neph.dat <- read.xls("../data/2015 BIM Nephrops quad rig trials/Arkh Angell 70 V 100mm
                    sheet = "Nephrops length",
                    stringsAsFactors = FALSE)

## remove Haul 11 and 12 currently as these compare 80mm
arkh.neph.dat <- subset(arkh.neph.dat, !Haul %in% c(11, 12))

## replace spaces in Compartment names
arkh.neph.dat$Compartment <- gsub(" ", "", arkh.neph.dat$Compartment)

## remove any redundant levels
arkh.neph.dat <- droplevels(arkh.neph.dat)

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

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

## make some factor variables used in the analyses
arkh.neph.dat$fHAUL <- factor(arkh.neph.dat$HAUL, levels = unique(arkh.neph.dat$HAUL))
arkh.neph.dat$Compartment <- factor(paste("c", arkh.neph.dat$Compartment, sep = ""))

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

Prepare the data for a multinomial fit.

```r
## 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("Compartment", "Carapace.length", "fHAUL", "Count")

## melt the data frame
arkh.neph.melt <- melt(arkh.neph.dat[, vars2keep],
                  id = c("Compartment", "Carapace.length", "fHAUL"))

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

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

## format the subsampling ratio similarly

## multiple raising factors per haul
rf.count <- with(arkh.neph.dat, table(fHAUL, Raising.factor, Compartment))

apply(rf.count, 1, FUN = function(x){sum(x>0)})
## hauls 4 and 9
## rf.count["H4",,] ## 6.8(28obs) and 6.9(2obs) for 100mm Port
## rf.count["H9",,] ## 15.4(27obs) and 15.5(2obs) for 100mm Starboard
## re-set
arkh.neph.dat$Raising.factor[arkh.neph.dat$fHAUL == "H4" &
                             arkh.neph.dat$Compartment == "c100mmPort"] <- 6.8
arkh.neph.dat$Raising.factor[arkh.neph.dat$fHAUL == "H9" &
                             arkh.neph.dat$Compartment == "c100mmStarboard"] <- 15.4

## convert to sub-sampling ratio as in Celtic Warrior
arkh.neph.dat$SUBSRATIO <- 1/arkh.neph.dat$Raising.factor

vars2keep <- c("Compartment", "fHAUL", "SUBSRATIO")

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

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

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

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

## Extract the matrix of counts
```

```
count.vars <- c("c100mmPort_Count", "c70mmPort_Count",
                "c100mmStarboard_Count", "c70mmStarboard_Count")

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

colnames(neph.count.mat) <- c("c100mmPort_Count", "c70mmPort_Count",
                              "c100mmStarboard_Count", "c70mmStarboard_Count")

## Extract the matrix of subsampling ratios
subsratio.vars <- c("c100mmPort_SUBSRATIO", "c70mmPort_SUBSRATIO",
                    "c100mmStarboard_SUBSRATIO", "c70mmStarboard_SUBSRATIO")

subsratio.mat <- as.matrix(arkh.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(arkh.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("100mmPort", "70mmPort", "100mmStarboard", "70mmStarboard"),
                    each = m),
                    levels = c("100mmPort", "70mmPort", "100mmStarboard", "70mmStarboard"))
                  Carapace.length = rep(arkh.neph.cast$Carapace.length, times = 4),
                  fHAUL = rep(arkh.neph.cast$fHAUL, 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")
```

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: 100mm Port, 70mm Port, 100mm Starboard, 70mm

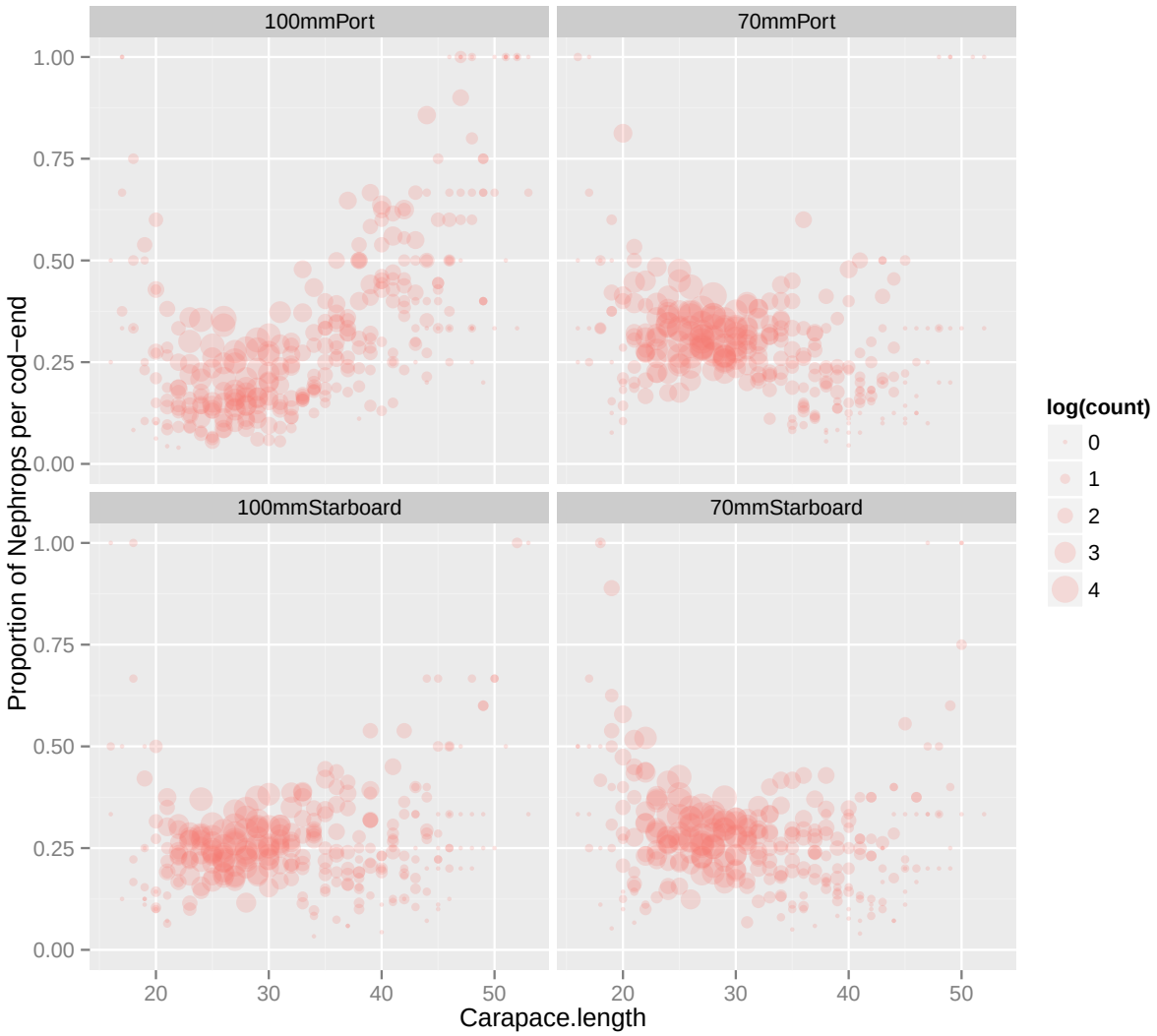


Figure 1: Arkh Angell quad-rig trial. Proportion of Nephrops catch retained per haul. Each point represents the proportion of the Nephrops catch (in number) per haul and length class retained in a given cod-end (70mm, 80mm, 90mm, or 100mm). The size of the point is proportional to the log of the count.

Starboard). 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)

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

## Extend to third order polynomial (based on AIC and BIC)
arkh.neph.cast$prop.Carapace.length2 <- arkh.neph.cast$prop.Carapace.length^2
arkh.neph.cast$prop.Carapace.length3 <- arkh.neph.cast$prop.Carapace.length^3

##
mnom.length <- multinom(neph.count.mat ~
                        prop.Carapace.length + prop.Carapace.length2 +
                        prop.Carapace.length3, data = arkh.neph.cast)

AIC(mnom0, mnom.length)

## try a VGAM fit for comparison
## particularly for 100mm Port small lengths
library(VGAM)

## find out best degrees of freedom
df.vec <- seq(1,10)
aic.vec <- rep(NA, 10)

for(i in 1:10){
  mnom.length.vgam <- vgam(neph.count.mat ~
                           s(prop.Carapace.length, df = df.vec[i]),
                           family = multinomial, data = arkh.neph.cast)
  aic.vec[i] <- AIC(mnom.length.vgam)
}

df.best <- df.vec[which.min(aic.vec)]
mnom.length.vgam <- vgam(neph.count.mat ~
                         s(prop.Carapace.length, df = df.best),
                         family = multinomial, data = arkh.neph.cast)

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

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

```
pred.vgam <- predict(mnom.length.vgam,
                     newdata = data.frame(prop.Carapace.length = pred.prop.length),
                     type = "response")

pred.vgam.df <- data.frame(
                   Mesh.Size =
                   factor(rep(c("100mmPort", "70mmPort",
                                "100mmStarboard", "70mmStarboard"),
                     each = m),
                          levels = c("100mmPort", "70mmPort",
                            "100mmStarboard", "70mmStarboard")),
                   Carapace.length = rep(pred.length, times = 4),
                   proportion = c(pred.vgam))
```

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(arkh.neph.cast$prop.Carapace.length),
                        max(arkh.neph.cast$prop.Carapace.length), length = 100)

pred.length <- seq(min(arkh.neph.cast$Carapace.length),
                        max(arkh.neph.cast$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)
  p70mmPort <- exp(X %*% matrix(beta[1,]))/(1 + rowSums(exp(X %*% t(beta))))
  p100mmStarboard <- exp(X %*% matrix(beta[2,]))/(1 + rowSums(exp(X %*% t(beta))))
```

```
  p70mmStarboard <- exp(X %*% matrix(beta[3,]))/(1 + rowSums(exp(X %*% t(beta))))
  p100mmPort <- 1 - p70mmPort - p100mmStarboard - p70mmStarboard
  pred.p <- cbind(p100mmPort, p70mmPort, p100mmStarboard, p70mmStarboard)
  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("100mmPort", "70mmPort",
                  "100mmStarboard", "70mmStarboard"),
                 each = m), levels = c("100mmPort", "70mmPort",
                              "100mmStarboard", "70mmStarboard")),
                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") +
  geom_line(data = pred.vgam.df, aes(x = Carapace.length, y = proportion),
            col = "green", size = 0.5)
```

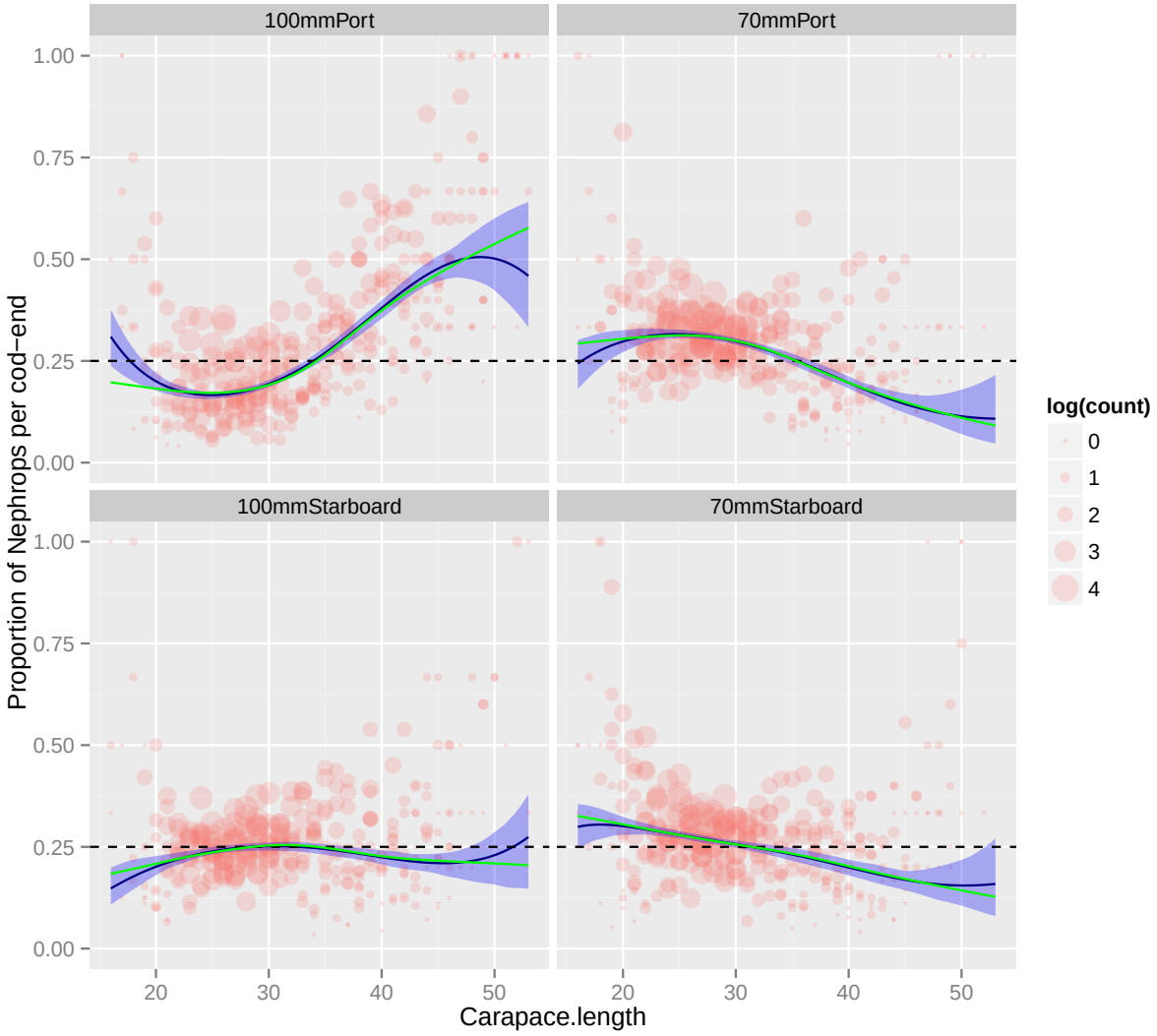


Figure 2: Arkh Angell quad-rig trials. Proportion of Nephrops catch retained per haul with fitted multinomial model and associated re-sampled intervals (blue). Null hypothesis of equal retention is displayed as the dashed line at 0.25. A multinomial additive model is shown in green for comparison.

Appendix IV

Preliminary multinomial analysis of *Our Lass* data

For discussion

1 Data

```
library(gdata)

our.lass.neph.dat <- read.xls("../data//2015 BIM Nephrops quad rig trials/Our Lass 2 70_80_
                      sheet = "All hauls",
                      stringsAsFactors = FALSE)

## Each mesh size has a unique codend number
with(our.lass.neph.dat, table(Mesh.Size, Codend.No))

## position switched
with(our.lass.neph.dat, table(Mesh.Size, Net.position))

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

## Make the "HAUL" variable character
our.lass.neph.dat$HAUL <- paste("H", our.lass.neph.dat$Haul.No, sep ="")

## make some factor variables used in the analyses
our.lass.neph.dat$fHAUL <- factor(our.lass.neph.dat$HAUL, levels = unique(our.lass.neph.dat
our.lass.neph.dat$Mesh.Size <- factor(paste("c", our.lass.neph.dat$Mesh.Size, sep = ""))

## remove observations above 99th and below 1th length percentile
## these can be highly influential on the fits
our.lass.neph.dat <- subset(our.lass.neph.dat, Carapace.length < quantile(Carapace.length,
                     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
```

```r
vars2keep <- c("Mesh.Size", "Carapace.length", "fHAUL", "Count")

## melt the data frame
our.lass.neph.melt <- melt(our.lass.neph.dat[, vars2keep],
                  id = c("Mesh.Size", "Carapace.length", "fHAUL"))

## re-form the dataframe in required format
our.lass.neph.cast <- cast(our.lass.neph.melt, Carapace.length + fHAUL ~ Mesh.Size  + varia
our.lass.neph.cast <- our.lass.neph.cast[order(our.lass.neph.cast$fHAUL, our.lass.neph.cast
our.lass.neph.cast[is.na(our.lass.neph.cast)] <- 0

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

```
##    Carapace.length fHAUL c100mm_Count c70mm_Count c80mm_Count c90mm_Count
## 10              22    H1            1           1           0           1
## 23              23    H1            1           4           2           1
```

```r
## format the subsampling ratio similarly

## unique raising factors per haul
rf.count <- with(our.lass.neph.dat, table(fHAUL, Overall.raising.factor, Mesh.Size))

apply(rf.count, 1, FUN = function(x){sum(x>0)})
```

```
##  H1  H2  H3  H4  H5  H6  H7  H8  H9 H10 H11 H12 H14
##   4   4   4   4   4   4   4   4   4   4   4   4   4
```

```r
## convert to sub-sampling ratio as in Celtic Warrior
our.lass.neph.dat$SUBSRATIO <- 1/our.lass.neph.dat$Overall.raising.factor
vars2keep <- c("Mesh.Size", "fHAUL", "SUBSRATIO")
subs.melt <- melt(unique(our.lass.neph.dat[, vars2keep]), id = c("Mesh.Size", "fHAUL"))
subs.cast <- cast(subs.melt, fHAUL  ~ Mesh.Size + variable)

## get net position of each
vars2keep <- c("Mesh.Size", "fHAUL", "Net.position")
netpos.melt <- melt(unique(our.lass.neph.dat[, vars2keep]), id = c("Mesh.Size", "fHAUL"))
netpos.cast <- cast(netpos.melt, fHAUL  ~ Mesh.Size + variable)

## merge counts and subsampling ratio back together
our.lass.neph.cast0 <- merge(our.lass.neph.cast, subs.cast, by = "fHAUL", all.x = TRUE)

our.lass.neph.cast <- merge(our.lass.neph.cast0, netpos.cast, by = "fHAUL", all.x = TRUE)

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

```
##   fHAUL Carapace.length c100mm_Count c70mm_Count c80mm_Count c90mm_Count
## 1    H1              22            1           1           0           1
```

```
## 2    H1               23            1           4           2           1
##   c100mm_SUBSRATIO c70mm_SUBSRATIO c80mm_SUBSRATIO c90mm_SUBSRATIO
## 1       0.02904138      0.03969299      0.03840203      0.06179486
## 2       0.02904138      0.03969299      0.03840203      0.06179486
##   c100mm_Net.position c70mm_Net.position c80mm_Net.position
## 1                   4                  1                  3
## 2                   4                  1                  3
##   c90mm_Net.position
## 1                  2
## 2                  2
```

```
## Extract the matrix of counts
count.vars <- c("c100mm_Count", "c70mm_Count", "c80mm_Count", "c90mm_Count")

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

## Extract the matrix of subsampling ratios
subsratio.vars <- c("c100mm_SUBSRATIO", "c70mm_SUBSRATIO", "c80mm_SUBSRATIO", "c90mm_SUBSRA

subsratio.mat <- as.matrix(our.lass.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(our.lass.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
netpos.vars <- c("c100mm_Net.position", "c70mm_Net.position", "c80mm_Net.position", "c90mm_

prop.mesh.df <- data.frame(
                  Mesh.Size = factor(rep(count.vars, each = m)),
                  Carapace.length = rep(our.lass.neph.cast$Carapace.length, times = 4),
                  fHAUL = rep(our.lass.neph.cast$fHAUL, times = 4),
                  Net.position = unlist(our.lass.neph.cast[, netpos.vars]),
                  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")
```

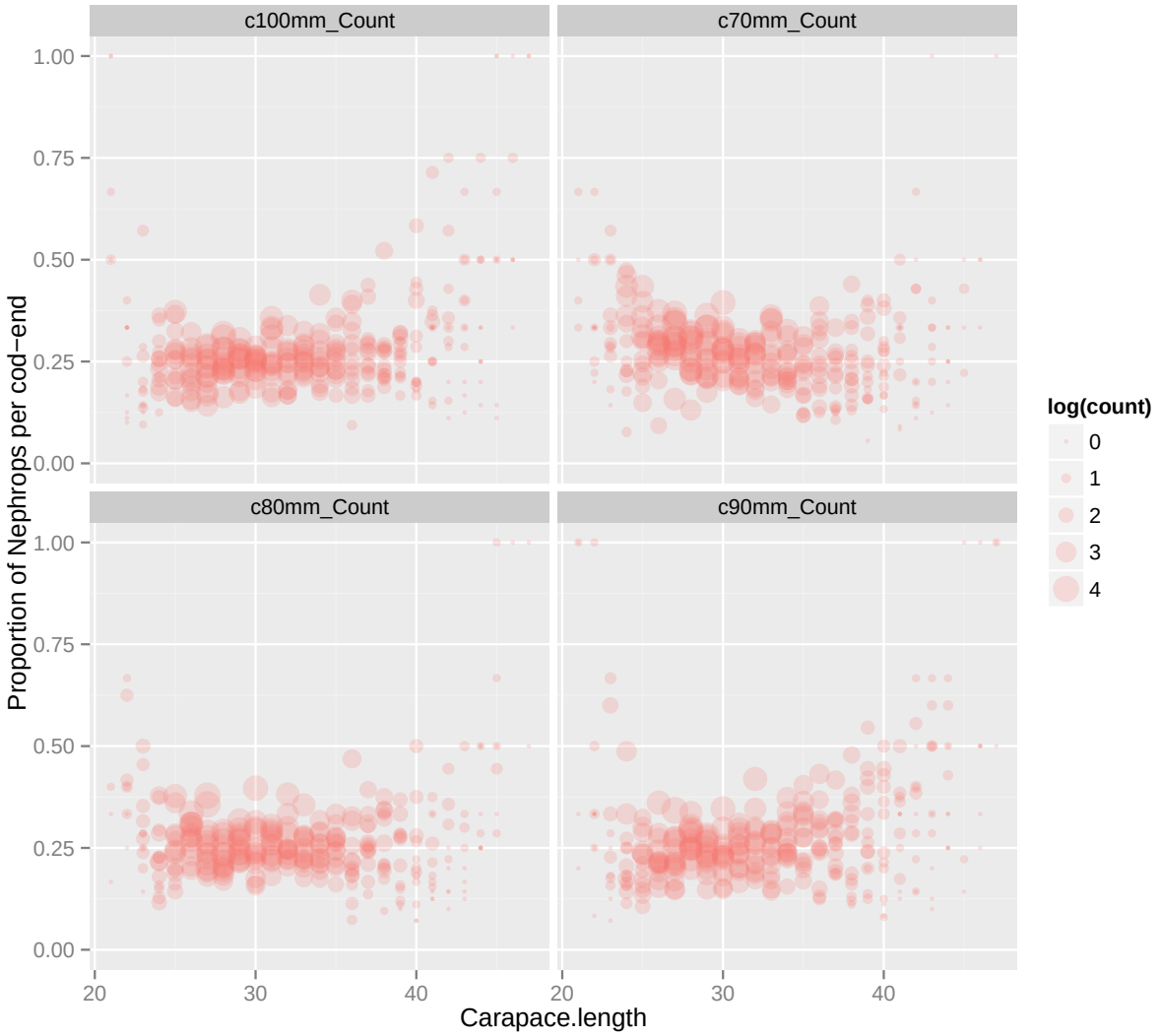


Figure 1: Proportion of Nephrops catch retained per haul. Each point represents the proportion of the Nephrops catch (in number) per haul and length class retained in a given cod-end (70mm, 80mm, 90mm, or 100mm). The size of the point is proportional to the log of the count.

```
##scale_colour_gradientn(colours=rainbow(4))
```

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: 70, 80, 90 and 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)

## # weights:  8 (3 variable)
## initial  value 23374.309223
## final  value 23367.514160
## converged

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

## Extend to third order polynomial (based on AIC and BIC)
our.lass.neph.cast$prop.Carapace.length2 <- our.lass.neph.cast$prop.Carapace.length^2
our.lass.neph.cast$prop.Carapace.length3 <- our.lass.neph.cast$prop.Carapace.length^3

##
mnom.length <- multinom(neph.count.mat ~
                        prop.Carapace.length + prop.Carapace.length2 + prop.Carapace.lengt

## # weights:  20 (12 variable)
## initial  value 23374.309223
## iter  10 value 23336.622678
## iter  20 value 23336.444786
## final  value 23336.392881
## converged

AIC(mnom0, mnom.length)

##             df      AIC
## mnom0        3 46741.03
## mnom.length 12 46696.79
```

```r
## try a VGAM fit for comparison
library(VGAM)

df.vec <- seq(1,10)
aic.vec <- rep(NA, 10)

for(i in 1:10){
  mnom.length.vgam <- vgam(neph.count.mat ~ s(prop.Carapace.length, df = df.vec[i]), family
  aic.vec[i] <- AIC(mnom.length.vgam)
}

df.best <- df.vec[which.min(aic.vec)]
mnom.length.vgam <- vgam(neph.count.mat ~ s(prop.Carapace.length, df = df.best), family = m

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

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

pred.vgam <- predict(mnom.length.vgam, newdata = data.frame(prop.Carapace.length = pred.pro

m <- length(pred.length)

pred.vgam.df <- data.frame(
                  Mesh.Size = factor(rep(count.vars, each = m)),
                  Carapace.length = rep(pred.length, times = 4),
                  proportion = c(pred.vgam))
```

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

```r
## 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(our.lass.neph.cast$prop.Carapace.length),
                        max(our.lass.neph.cast$prop.Carapace.length), length = 100)

pred.length <- seq(min(our.lass.neph.cast$Carapace.length),
                        max(our.lass.neph.cast$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)
  p70mmPort <- exp(X %*% matrix(beta[1,]))/(1 + rowSums(exp(X %*% t(beta))))
  p100mmStarboard <- exp(X %*% matrix(beta[2,]))/(1 + rowSums(exp(X %*% t(beta))))
  p70mmStarboard <- exp(X %*% matrix(beta[3,]))/(1 + rowSums(exp(X %*% t(beta))))
  p100mmPort <- 1 - p70mmPort - p100mmStarboard - p70mmStarboard
  pred.p <- cbind(p100mmPort, p70mmPort, p100mmStarboard, p70mmStarboard)
  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(count.vars, each = m)),
                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") +
  geom_line(data = pred.vgam.df, aes(x = Carapace.length, y = proportion),
            col = "green", size = 0.5)
```

2.1 Including net position as a covariate

```
## create factor variables
our.lass.neph.cast$f100mm_Net.position <- paste("pos", our.lass.neph.cast$c100mm_Net.positi
our.lass.neph.cast$f90mm_Net.position <- paste("pos", our.lass.neph.cast$c90mm_Net.position
our.lass.neph.cast$f80mm_Net.position <- paste("pos", our.lass.neph.cast$c80mm_Net.position
our.lass.neph.cast$f70mm_Net.position <- paste("pos", our.lass.neph.cast$c70mm_Net.position

mnom.length.netpos <- multinom(neph.count.mat ~
                               prop.Carapace.length +
                               prop.Carapace.length2 +
                               prop.Carapace.length3 +
                               f100mm_Net.position +
                               f70mm_Net.position +
                               f80mm_Net.position +
                               f90mm_Net.position,
                               data = our.lass.neph.cast)


## # weights:  68 (48 variable)
## initial  value 23374.309223
## iter  10 value 23326.650060
## iter  20 value 23319.754756
## iter  30 value 23318.476676
## final  value 23318.304129
## converged

AIC(mnom.length, mnom.length.netpos)


##                     df      AIC
## mnom.length         12 46696.79
## mnom.length.netpos  24 46684.61
```

Plot the net position predictions

```
## get predictions by net position

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

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

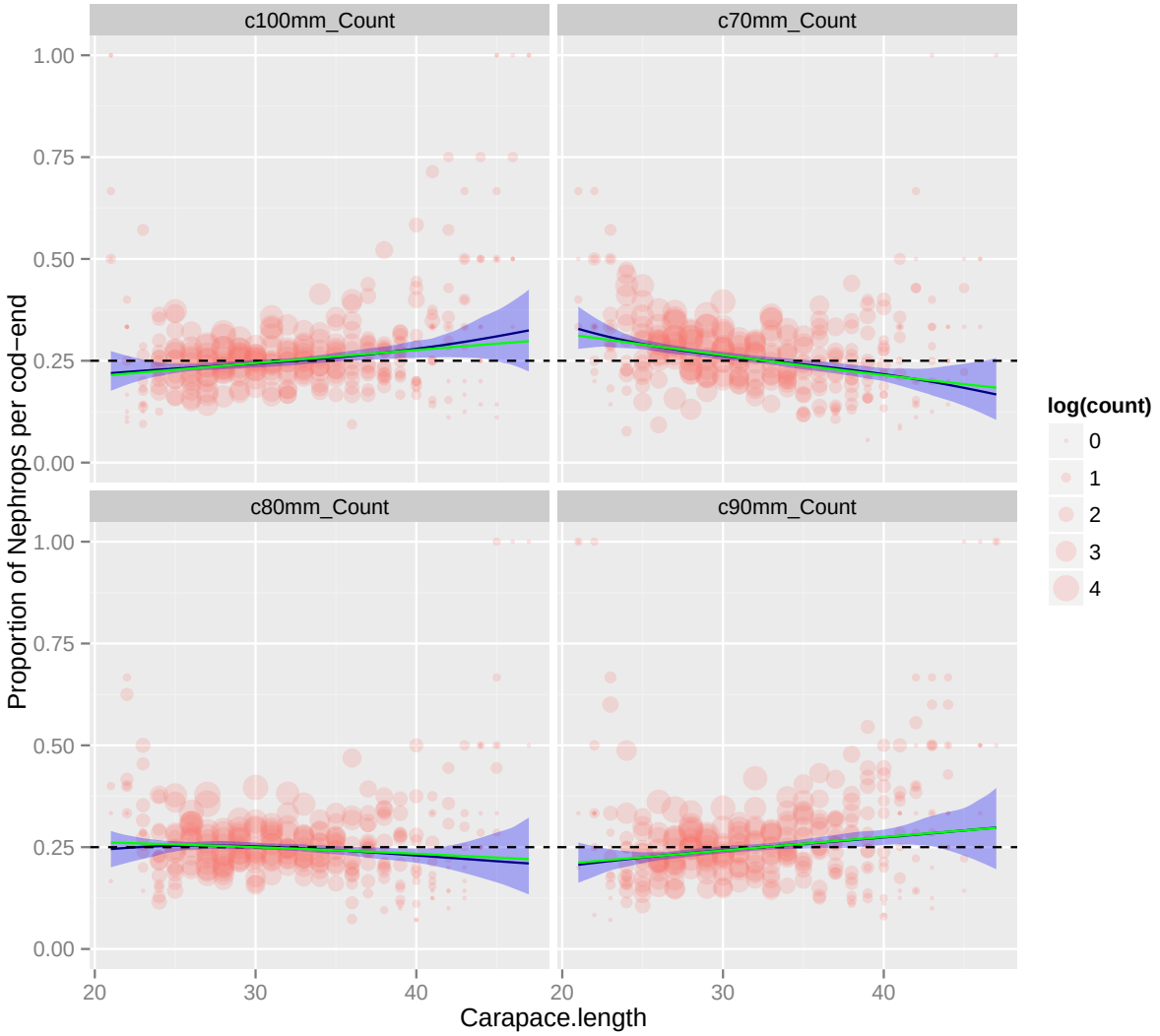
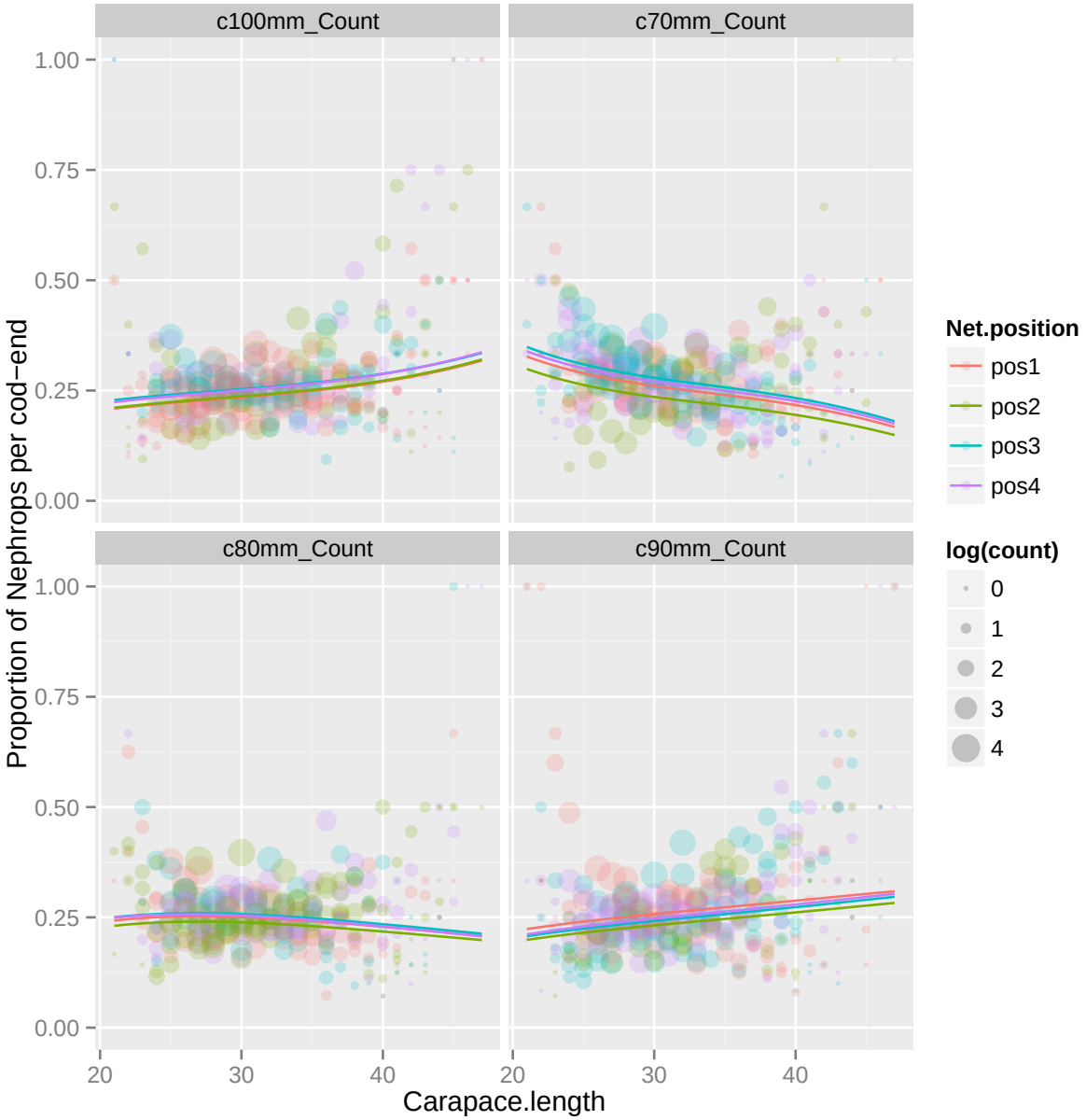


Figure 2: Proportion of Nephrops catch retained per haul with fitted multinomial model and associated re-sampled intervals. Null hypothesis of equal retention is displayed as the dashed line at 0.25.

```r
pos.vec <- paste("pos", 1:4, sep = "")

## removing fifth row for now
netpos.unique <- as.data.frame(apply(unique(netpos.cast[,2:5])[-5, ], 2, FUN = function(x){

names(netpos.unique) <- names(netpos.cast[,2:5])

names(netpos.unique) <- gsub("c", "f", names(netpos.unique))

m <- length(pred.prop.length)

pred.df <- netpos.unique[rep(1:4, each = m), ]
pred.df$prop.Carapace.length <- rep(pred.prop.length, times = 4)
pred.df$prop.Carapace.length2 <- pred.df$prop.Carapace.length^2
pred.df$prop.Carapace.length3 <- pred.df$prop.Carapace.length^3
pred.df$Carapace.length <- rep(pred.length, times = 4)

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

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

pred.netpos.df <- data.frame(
                    Mesh.Size = factor(rep(count.vars, each = m)),
                    Carapace.length = rep(pred.df$Carapace.length, times = 4),
                    Net.position = unlist(pred.df[, 1:4]),
                    proportion = c(pred.netpos))

ggplot(prop.mesh.df2, aes(x = Carapace.length, y = proportion, colour = Net.position)) +
  geom_point(alpha = 0.2, aes(size = log(count))) +
facet_wrap(~ Mesh.Size) + ylab("Proportion of Nephrops per cod-end") +
  ##scale_colour_manual(values=rainbow(4)) +
  geom_line(data = pred.netpos.df, aes(x = Carapace.length, y = proportion))
```


c100mm_Count
c70mm_Count
c80mm_Count
c90mm_Count
Proportion of Nephrops per cod−end
1.00
0.75
0.50
0.25
0.00
20
30
40
Carapace.length
Net.position
pos1
pos2
pos3
pos4
log(count)
0
1
2
3
4

Appendix V

Our Lass 70mm versus 80mm analysis

For discussion

1 Data

Read the data in and make a few factor variables

```
library(gdata)
library(reshape)

neph.dat <- read.xls("../data/2015 BIM Nephrops quad rig trials/Our Lass 2 70_80_90_100mm c
                     sheet = "All hauls",
                     stringsAsFactors = FALSE)

## subset containing the 70mm and 80mm data
neph.7080 <- subset(neph.dat, Mesh.Size %in% c("70mm", "80mm"))

## make factor haul variable
neph.7080$fHAUL <- factor(paste("H", neph.7080$Haul.No, sep =""))

## make a factor mesh size variable
neph.7080$fMesh.Size <- factor(paste("mesh", neph.7080$Mesh.Size, sep = ""))

## remove trailing dot from raised count column name
names(neph.7080)[names(neph.7080) == "Raised.count."] <- "Raised.count"
```

Re-shape the data to wide format (columns for 70mm, 80mm variables).

```
## 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("fHAUL", "fMesh.Size", "Net.position", "Carapace.length",
               "Count", "Raised.count", "Total.catch", "Overall.Sampling.Ratio")

## melt the data frame
neph.7080.melt <- melt(neph.7080[, vars2keep],
                           id = c("fHAUL", "fMesh.Size", "Carapace.length"))

## re-form the dataframe in required format
```

```
neph.7080.cast <- cast(neph.7080.melt, Carapace.length + fHAUL ~ fMesh.Size  + variable)

## first couple of lines
head(neph.7080.cast, 2)

##   Carapace.length fHAUL mesh70mm_Net.position mesh70mm_Count
## 1              17   H14                    4             1
## 2              17    H3                   NA            NA
##   mesh70mm_Raised.count mesh70mm_Total.catch
## 1                  32.1                467.7
## 2                    NA                   NA
##   mesh70mm_Overall.Sampling.Ratio mesh80mm_Net.position mesh80mm_Count
## 1                      0.03112036                   NA             NA
## 2                              NA                    3              1
##   mesh80mm_Raised.count mesh80mm_Total.catch
## 1                    NA                   NA
## 2                  32.1               420.05
##   mesh80mm_Overall.Sampling.Ratio
## 1                              NA
## 2                      0.03118295

summary(neph.7080.cast) ## note lots of NAs

##  Carapace.length     fHAUL     mesh70mm_Net.position mesh70mm_Count
##  Min.   :17.00   H11    : 31   Min.   :1.000         Min.   : 1.00
##  1st Qu.:27.00   H14    : 28   1st Qu.:2.000         1st Qu.: 3.00
##  Median :33.00   H4     : 27   Median :3.000         Median :10.00
##  Mean   :33.18   H7     : 27   Mean   :2.657         Mean   :14.76
##  3rd Qu.:40.00   H6     : 26   3rd Qu.:4.000         3rd Qu.:24.00
##  Max.   :54.00   H8     : 26   Max.   :4.000         Max.   :66.00
##                  (Other):172   NA's   :37            NA's   :37
##  mesh70mm_Raised.count mesh70mm_Total.catch
##  Min.   :  10.8        Min.   :203.1
##  1st Qu.:  71.6        1st Qu.:306.0
##  Median : 205.8        Median :411.3
##  Mean   : 374.9        Mean   :416.3
##  3rd Qu.: 580.4        3rd Qu.:490.1
##  Max.   :2251.3        Max.   :618.0
##  NA's   :37            NA's   :37
##  mesh70mm_Overall.Sampling.Ratio mesh80mm_Net.position mesh80mm_Count
##  Min.   :0.02310                 Min.   :1.000         Min.   : 1.00
##  1st Qu.:0.03112                 1st Qu.:2.000         1st Qu.: 2.25
##  Median :0.04072                 Median :2.000         Median :10.00
##  Mean   :0.04926                 Mean   :2.441         Mean   :13.79
##  3rd Qu.:0.05830                 3rd Qu.:3.000         3rd Qu.:21.75
##  Max.   :0.09226                 Max.   :4.000         Max.   :61.00
##  NA's   :37                      NA's   :31            NA's   :31
##  mesh80mm_Raised.count mesh80mm_Total.catch
##  Min.   :   9.4        Min.   :165.8
##  1st Qu.:  52.5        1st Qu.:264.8
```

```
##  Median : 193.4        Median :407.2
##  Mean   : 305.8        Mean   :388.1
##  3rd Qu.: 441.5        3rd Qu.:458.8
##  Max.   :1621.2        Max.   :635.3
##  NA's   :31            NA's   :31
##  mesh80mm_Overall.Sampling.Ratio
##  Min.   :0.02309
##  1st Qu.:0.03163
##  Median :0.04821
##  Mean   :0.05499
##  3rd Qu.:0.06868
##  Max.   :0.10591
##  NA's   :31
```

```r
## fill in missing values
## these occur if there is a count for e.g. 20mm CL in 70mm but not in 80mm
neph.7080.cast$mesh70mm_Count[is.na(neph.7080.cast$mesh70mm_Count)] <- 0
neph.7080.cast$mesh70mm_Raised.count[is.na(neph.7080.cast$mesh70mm_Raised.count)] <- 0
neph.7080.cast$mesh80mm_Count[is.na(neph.7080.cast$mesh80mm_Count)] <- 0
neph.7080.cast$mesh80mm_Raised.count[is.na(neph.7080.cast$mesh80mm_Raised.count)] <- 0

for(i in 1:dim(neph.7080.cast)[1]){
  haul.dat <- subset(neph.7080.cast, fHAUL == neph.7080.cast$fHAUL[i])
  ## 70mm net position
  if(is.na(neph.7080.cast$mesh70mm_Net.position[i])){
    neph.7080.cast$mesh70mm_Net.position[i] <-
      unique(na.omit(haul.dat$mesh70mm_Net.position))
  }
  ## 80mm net position
  if(is.na(neph.7080.cast$mesh80mm_Net.position[i])){
    neph.7080.cast$mesh80mm_Net.position[i] <-
      unique(na.omit(haul.dat$mesh80mm_Net.position))
  }
  ## 70mm total catch
  if(is.na(neph.7080.cast$mesh70mm_Total.catch[i])){
    neph.7080.cast$mesh70mm_Total.catch[i] <-
      unique(na.omit(haul.dat$mesh70mm_Total.catch))
  }
  ## 80mm total catch
  if(is.na(neph.7080.cast$mesh80mm_Total.catch[i])){
    neph.7080.cast$mesh80mm_Total.catch[i] <-
      unique(na.omit(haul.dat$mesh80mm_Total.catch))
  }
  ## Sampling ratio
  ## 70mm total catch
  if(is.na(neph.7080.cast$mesh70mm_Overall.Sampling.Ratio[i])){
    neph.7080.cast$mesh70mm_Overall.Sampling.Ratio[i] <-
      unique(na.omit(haul.dat$mesh70mm_Overall.Sampling.Ratio))
  }
  ## 80mm total catch
  if(is.na(neph.7080.cast$mesh80mm_Overall.Sampling.Ratio[i])){
```

```
    neph.7080.cast$mesh80mm_Overall.Sampling.Ratio[i] <-
      unique(na.omit(haul.dat$mesh80mm_Overall.Sampling.Ratio))
  }
}

summary(neph.7080.cast) ## no missing


##  Carapace.length     fHAUL     mesh70mm_Net.position mesh70mm_Count
##  Min.   :17.00   H11    : 31   Min.   :1.000         Min.   : 0.00
##  1st Qu.:27.00   H14    : 28   1st Qu.:2.000         1st Qu.: 2.00
##  Median :33.00   H4     : 27   Median :3.000         Median : 8.00
##  Mean   :33.18   H7     : 27   Mean   :2.656         Mean   :13.14
##  3rd Qu.:40.00   H6     : 26   3rd Qu.:4.000         3rd Qu.:23.00
##  Max.   :54.00   H8     : 26   Max.   :4.000         Max.   :66.00
##                  (Other):172
##  mesh70mm_Raised.count mesh70mm_Total.catch
##  Min.   :   0.0        Min.   :203.1
##  1st Qu.:  34.3        1st Qu.:306.0
##  Median : 160.7        Median :411.3
##  Mean   : 333.7        Mean   :416.9
##  3rd Qu.: 520.6        3rd Qu.:490.1
##  Max.   :2251.3        Max.   :618.0
##
##  mesh70mm_Overall.Sampling.Ratio mesh80mm_Net.position mesh80mm_Count
##  Min.   :0.02310                 Min.   :1.000         Min.   : 0.00
##  1st Qu.:0.03112                 1st Qu.:2.000         1st Qu.: 2.00
##  Median :0.04072                 Median :2.000         Median : 7.00
##  Mean   :0.04960                 Mean   :2.442         Mean   :12.52
##  3rd Qu.:0.05830                 3rd Qu.:3.000         3rd Qu.:20.00
##  Max.   :0.09226                 Max.   :4.000         Max.   :61.00
##
##  mesh80mm_Raised.count mesh80mm_Total.catch
##  Min.   :   0.0        Min.   :165.8
##  1st Qu.:  32.1        1st Qu.:264.8
##  Median : 158.1        Median :407.2
##  Mean   : 277.7        Mean   :387.6
##  3rd Qu.: 406.3        3rd Qu.:458.8
##  Max.   :1621.2        Max.   :635.3
##
##  mesh80mm_Overall.Sampling.Ratio
##  Min.   :0.02309
##  1st Qu.:0.03163
##  Median :0.04821
##  Mean   :0.05452
##  3rd Qu.:0.06868
##  Max.   :0.10591
##
```

Get the empirical proportion 80/(70 + 80) at length. Note that the length-specific CIs do not reflect the non-independence of the observations across lengths at the haul level are therefore not plotted.

```
## vector of carapace lengths
cl.vec <- unique(neph.7080.cast$Carapace.length)
cl.vec <- cl.vec[order(cl.vec)]

## including number of observations contributing to count here
count.df <- data.frame(Carapace.length = cl.vec, N = NA, prop.80 = NA)

for(i in 1:dim(count.df)[1]){
  sub.dat <- subset(neph.7080.cast, Carapace.length == count.df$Carapace.length[i])
  ##
  if(dim(sub.dat)[1] > 1){
    ## raised number
    count.df$N[i] <- with(sub.dat,
                          round(sum(mesh80mm_Raised.count)) + round(sum(mesh70mm_Raised.cou
    ##
    count.df$prop.80[i] <- with(sub.dat, round(sum(mesh80mm_Raised.count)) /
                                (round(sum(mesh80mm_Raised.count)) + round(sum(mesh70mm_Rai
    rm(list = c("sub.dat"))
  }
}
```

Plot the data (Figure 1)

```
with(count.df, plot(Carapace.length, prop.80, ylim = c(0, 1), pch = 19,
                    xlab = "Carapace length (mm)",
                    ylab = "Proportion (N80mm/(N70mm + N80mm))",
                    bty = "L",
                    cex = 1/5 * log(N)))
abline(h = 0.5, lty = 2)
```

2 Models

A catch comparison binomial Generalized Additive/Linear Mixed Model is suitable choice for these count data where we are interested in estimating how the proportion changes with carapace length. We first try a model with only carapace length as an explanatory variable with haul random effects.

```
library(mgcv)

neph.7080.cast$dum <- 1

## no length effect
gamm.null <- gam(cbind(mesh80mm_Count, mesh70mm_Count) ~ 1 +
                   s(fHAUL, bs="re", by = dum),
                   offset =
                   log(mesh80mm_Overall.Sampling.Ratio /
```

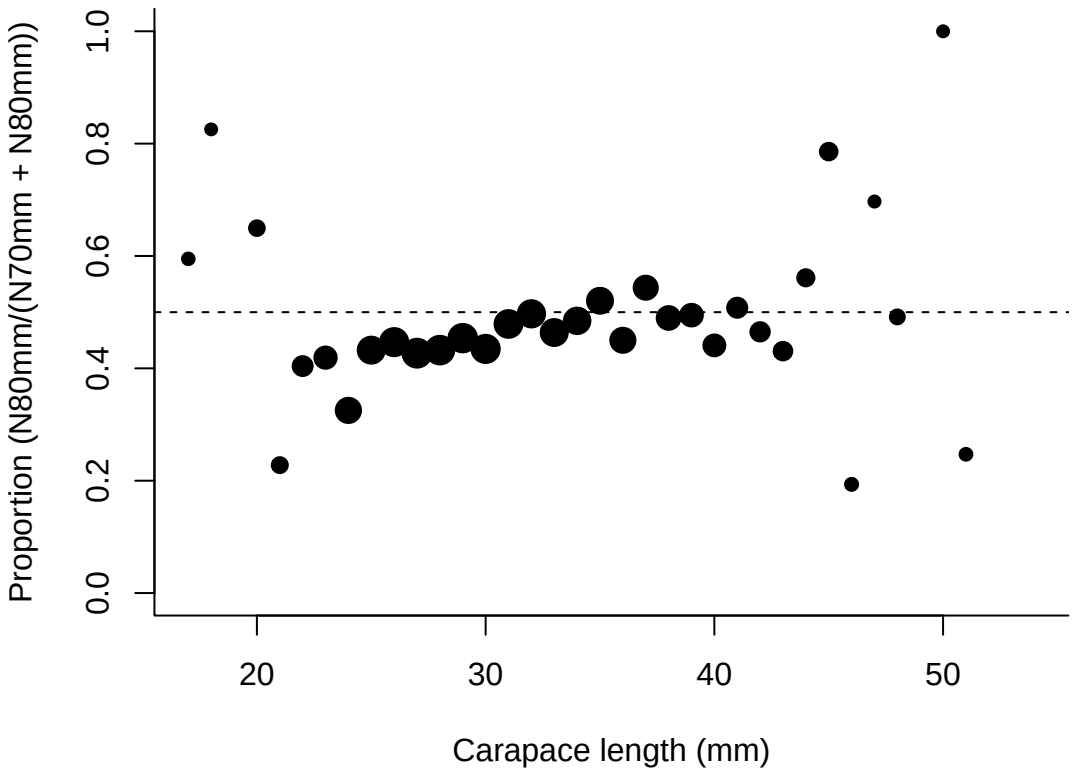


Figure 1: Proportion of Nephrops raised numbers retained in the 80mm over the sum of the 80mm and 70mm meshes.

```
                    mesh70mm_Overall.Sampling.Ratio),
                family = binomial,
                data = neph.7080.cast)

gamm.alt <- gam(cbind(mesh80mm_Count, mesh70mm_Count) ~
                s(Carapace.length, k = 5) +
                s(fHAUL, bs="re", by = dum),
                offset =
                log(mesh80mm_Overall.Sampling.Ratio /
                    mesh70mm_Overall.Sampling.Ratio),
                family = binomial,
                data = neph.7080.cast)

## likelihood ratio test for the significance of carapace length
anova(gamm.null, gamm.alt, test = "Chisq")


## Analysis of Deviance Table
##
## Model 1: cbind(mesh80mm_Count, mesh70mm_Count) ~ 1 + s(fHAUL, bs = "re",
##     by = dum)
## Model 2: cbind(mesh80mm_Count, mesh70mm_Count) ~ s(Carapace.length, k = 5) +
##     s(fHAUL, bs = "re", by = dum)
##   Resid. Df Resid. Dev      Df Deviance Pr(>Chi)
## 1    324.11     438.09
## 2    323.11     429.68 0.99849   8.4149 0.003711 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Plot the predictions from this model.

```
mean.catch <- mean(c(unique(neph.7080.cast$mesh70mm_Total.catch),
                     unique(neph.7080.cast$mesh80mm_Total.catch)))

## data frame to predictfor
pred.df <- data.frame(Carapace.length = cl.vec,
                      fHAUL = "H1",
                      dum = 0,
                      mesh80mm_Overall.Sampling.Ratio = 1,
                      mesh70mm_Overall.Sampling.Ratio = 1,
                      mesh70mm_Total.catch = mean.catch,
                      mesh80mm_Total.catch = mean.catch
                      )

pred.gamm.alt <- predict(gamm.alt, newdata = pred.df, se.fit = TRUE)

## predicted proportions and confidence intervals
pred.df$pred.prop <- plogis(pred.gamm.alt$fit)
pred.df$lwr.prop <- plogis(pred.gamm.alt$fit - qnorm(0.975) * pred.gamm.alt$se.fit)
pred.df$upr.prop <- plogis(pred.gamm.alt$fit + qnorm(0.975) * pred.gamm.alt$se.fit)
```

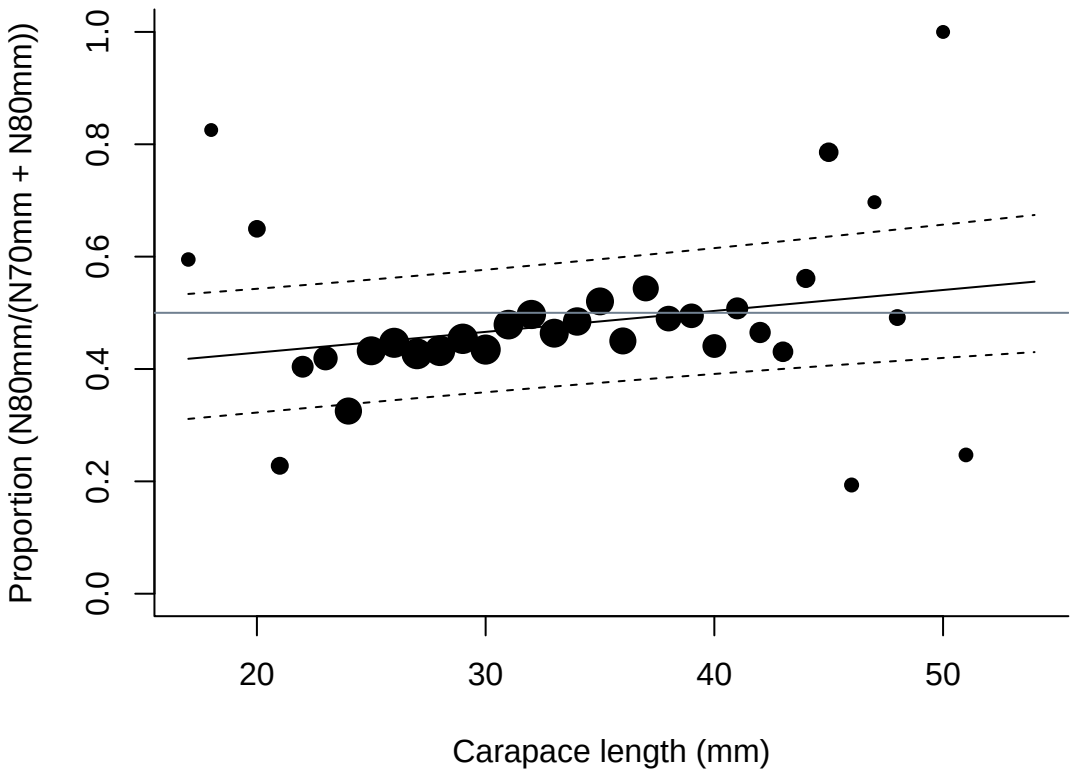


Figure 2: Predicted proportion from binomial GLMM without covariates.

```
with(count.df, plot(Carapace.length, prop.80, ylim = c(0, 1), pch = 19,
                    xlab = "Carapace length (mm)",
                    ylab = "Proportion (N80mm/(N70mm + N80mm))",
                    bty = "L",
                    cex = 1/5 * log(N)))

with(pred.df, lines(Carapace.length, pred.prop))
with(pred.df, lines(Carapace.length, lwr.prop, lty = 2))
with(pred.df, lines(Carapace.length, upr.prop, lty = 2))
abline(h = 0.5, col = "slategrey")
```

The cause of the wide confidence intervals (Figure 2) is the large amount of inter-haul variability in the proportion retained in the 80mm (Figure 3)

```r
## Get the proportion at length by haul
haul.count.df <- expand.grid(Carapace.length = cl.vec,
                             fHAUL = unique(neph.7080.cast$fHAUL))

## order the levels
haul.count.df$fHAUL <- factor(as.character(haul.count.df$fHAUL),
                              levels = c(paste("H", 1:12, sep = ""), "H14"))

haul.count.df$prop.80 <- NA
haul.count.df$N <- NA
haul.count.df$lwr <- NA
haul.count.df$upr <- NA

for(i in 1:dim(haul.count.df)[1]){
  sub.dat <- subset(neph.7080.cast,
                    Carapace.length == haul.count.df$Carapace.length[i] &
                    fHAUL == haul.count.df$fHAUL[i])
  ##
  ##if((sub.dat$mesh80mm_Raised.count + sub.dat$mesh70mm_Raised.count) > 0){
  if(dim(sub.dat)[1] > 0){
    btest <- with(sub.dat,
                  binom.test(x = round(mesh80mm_Raised.count),
                             n = round(mesh80mm_Raised.count + mesh70mm_Raised.count)))
    ##
    haul.count.df$N[i] <- with(sub.dat,
                               round(mesh80mm_Raised.count + mesh70mm_Raised.count))
    haul.count.df$prop.80[i] <- btest$estimate
    haul.count.df$lwr[i] <- btest$conf.int[1]
    haul.count.df$upr[i] <- btest$conf.int[2]
    ##
    rm(list = c("sub.dat", "btest"))
  }
}

## get predictions at the HAUL level from model
haul.count.df$dum <- 1
haul.count.df$pred.prop <- plogis(predict(gamm.alt, newdata = haul.count.df))


library(ggplot2)

blue2red <- colorRampPalette(c("darkblue", "white", "red"))

ggplot(haul.count.df, aes(x = Carapace.length, y = prop.80)) +
  geom_point(aes(colour = fHAUL, size = 1/5 * log(N))) +
  geom_line(data = haul.count.df, aes(x = Carapace.length, y = pred.prop,  colour = fHAUL)) +
  scale_colour_manual(values = blue2red(13)) + ylab("Proportion in 80mm")


## Warning:  Removed 118 rows containing missing values (geom_point).
```

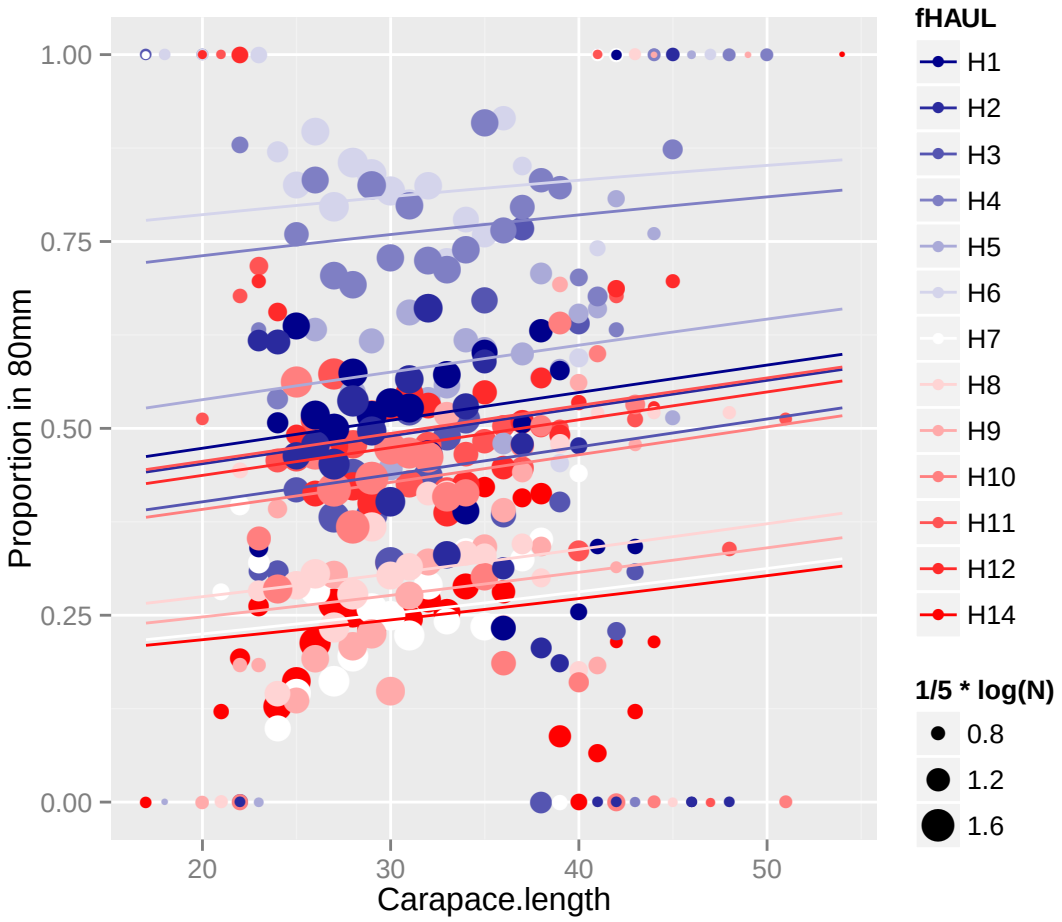


Figure 3: Observed proportion in the 80mm by haul. Note the wide variability of the proportion with some having much higher or lower proportions. Fitted lines come from the GLMM with carapace length only.

We can take a look at additional measured covariates to see if these relate to the haul-level variability (random effects in the model above) (Figure 4). Firstly, read in the haul duration data

```
gear.details <- read.xls("../data/2015 BIM Nephrops quad rig trials/Our Lass 2 70_80_90_100

## merge with the neph data
gear.details$fHAUL <- factor(paste("H", gear.details$Tow.., sep = ""))

tmp <- strsplit(as.character(gear.details$Tow.duration), split = ":")

hr <- as.numeric(unlist(lapply(tmp, "[", 1)))
min <- as.numeric(unlist(lapply(tmp, "[", 2)))

gear.details$dec.hr <- hr + min / 60
```

```
##
ranef.df <- data.frame(fHAUL = levels(neph.7080.cast$fHAUL),
                       ranef = coef(gamm.alt)[-c(1:5)])
##
covar.names <- c("fHAUL", "mesh70mm_Net.position", "mesh70mm_Total.catch",
                 "mesh80mm_Net.position", "mesh80mm_Total.catch")
##
covar.df <- unique(neph.7080.cast[, covar.names])
## include haul duration
covar.df <- merge(covar.df, gear.details[, c("fHAUL", "dec.hr")])
##
ranef.df <- merge(ranef.df, covar.df)
## convert to long format for plotting
ranef.df <- melt(ranef.df, id = c("fHAUL", "ranef"))

##
ggplot(ranef.df, aes(x = value, y = ranef)) +
  geom_point() +
  facet_wrap(~ variable, scales = "free") +
  xlab("Covariate value") +
  ylab("Random effect")
```

There are some strong relationships between the random effects and measured covariates (Figure 4). It is best to include these measured variables in the model as fixed effects.

```
library(lme4)
library(effects)

## including additional covariates
## check identifiability
neph.7080.cast$fmesh80mm_Net.position <-
  factor(paste("pos",
               neph.7080.cast$mesh80mm_Net.position, sep = ""))
```

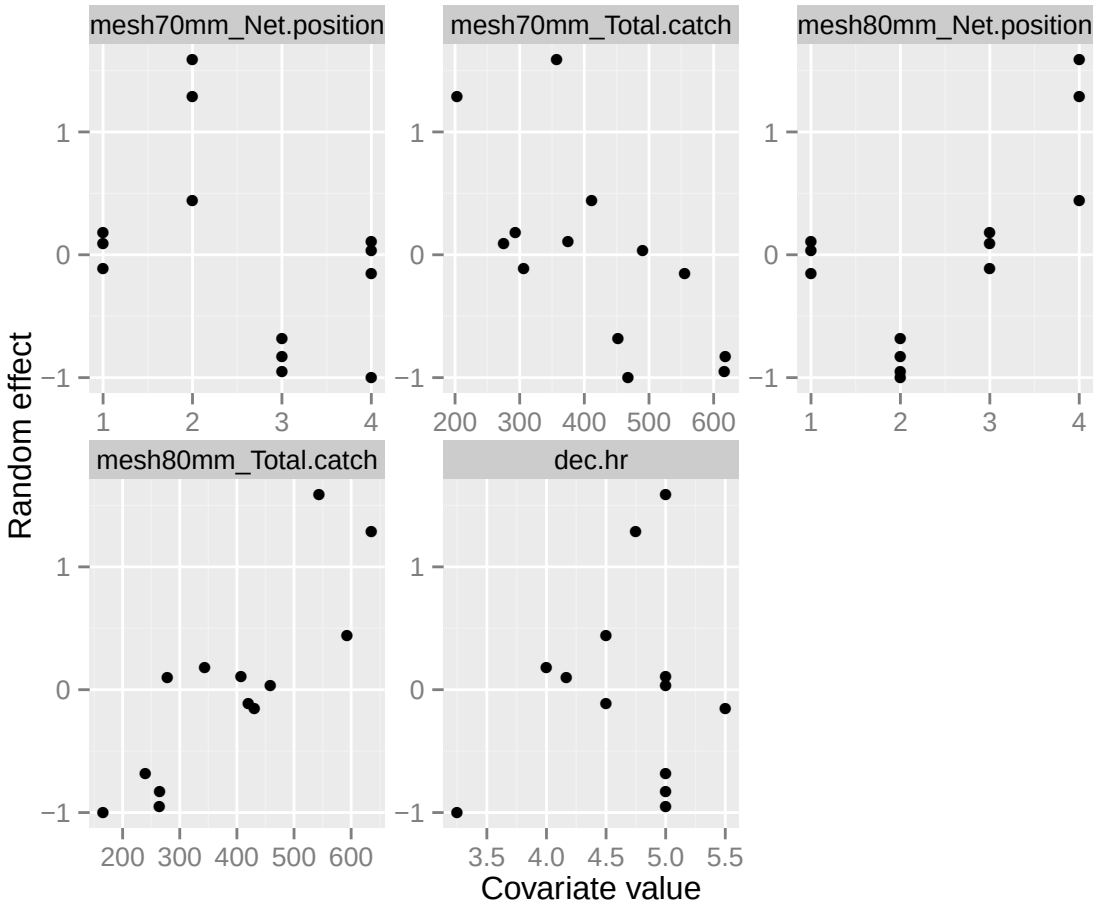


Figure 4: Relationship between the random effects of the carapace length only model and measured covariates .

```
neph.7080.cast <- merge(neph.7080.cast, gear.details[, c("fHAUL", "dec.hr")])

## using log of total catch weights - return to this
neph.7080.cast$log.mesh80mm_Total.catch <- log(neph.7080.cast$mesh80mm_Total.catch)
neph.7080.cast$log.mesh70mm_Total.catch <- log(neph.7080.cast$mesh70mm_Total.catch)

max.Carapace.length <- max(neph.7080.cast$Carapace.length)
neph.7080.cast$prop.Carapace.length <- neph.7080.cast$Carapace.length / max.Carapace.length

## Note should return to this warning later
## fits okay in gam but glmm used for effects package
glmm.alt.covar <- glmer(cbind(mesh80mm_Count, mesh70mm_Count) ~
                        ##I(log(mesh80mm_Total.catch / mesh70mm_Total.catch)) * Carapace.le
                        log.mesh80mm_Total.catch + log.mesh70mm_Total.catch +
                        prop.Carapace.length +
                        fmesh80mm_Net.position +
                        (1 | fHAUL),
                        offset =
                        log(mesh80mm_Overall.Sampling.Ratio /
                            mesh70mm_Overall.Sampling.Ratio),
                        family = binomial,
                        data = neph.7080.cast,
                        control=glmerControl(optimizer="bobyqa"))

glmm.alt.covar.catchdur <- glmer(cbind(mesh80mm_Count, mesh70mm_Count) ~
                                 ##I(log(mesh80mm_Total.catch / mesh70mm_Total.catch)) * Ca
                                 log.mesh80mm_Total.catch + log.mesh70mm_Total.catch +
                                 prop.Carapace.length +
                                 fmesh80mm_Net.position +
                                 poly(dec.hr, 1) +
                                 (1 | fHAUL),
                                 offset =
                                 log(mesh80mm_Overall.Sampling.Ratio /
                                     mesh70mm_Overall.Sampling.Ratio),
                                 family = binomial,
                                 data = neph.7080.cast,
                                 control=glmerControl(optimizer="bobyqa"))

glmm.alt.covar.catchdur.nobulk <- glmer(cbind(mesh80mm_Count, mesh70mm_Count) ~
                                        ##I(log(mesh80mm_Total.catch / mesh70mm_Total.catch
                                        ##log.mesh80mm_Total.catch + log.mesh70mm_Total.ca
                                        prop.Carapace.length +
                                        fmesh80mm_Net.position +
                                        poly(dec.hr, 1) +
                                        (1 | fHAUL),
                                        offset =
                                        log(mesh80mm_Overall.Sampling.Ratio /
                                            mesh70mm_Overall.Sampling.Ratio),
                                        family = binomial,
                                        data = neph.7080.cast,
                                        control=glmerControl(optimizer="bobyqa"))
```

```
## include squared length term
glmm.alt.covar2 <- glmer(cbind(mesh80mm_Count, mesh70mm_Count) ~
                         ##I(log(mesh80mm_Total.catch / mesh70mm_Total.catch)) * Carapace..
                         log.mesh80mm_Total.catch + log.mesh70mm_Total.catch +
                         poly(prop.Carapace.length, 2) +
                         poly(dec.hr, 1) +
                         fmesh80mm_Net.position +
                         (1 | fHAUL),
                         offset =
                         log(mesh80mm_Overall.Sampling.Ratio /
                             mesh70mm_Overall.Sampling.Ratio),
                         family = binomial,
                         data = neph.7080.cast,
                         control=glmerControl(optimizer="bobyqa"))

## in GAM
gamm.alt.covar2 <- gam(cbind(mesh80mm_Count, mesh70mm_Count) ~
                       ##I(log(mesh80mm_Total.catch / mesh70mm_Total.catch)) * Carapace.le
                       log.mesh80mm_Total.catch + log.mesh70mm_Total.catch +
                       s(prop.Carapace.length, k = 5, fx = TRUE) +
                       poly(dec.hr, 1) +
                       fmesh80mm_Net.position +
                       s(fHAUL, bs="re", by = dum),
                       offset =
                       log(mesh80mm_Overall.Sampling.Ratio /
                           mesh70mm_Overall.Sampling.Ratio),
                       family = binomial,
                       data = neph.7080.cast)

## use effects package to get prediction for model with net position
## set predictor variables
xlevels <- list(prop.Carapace.length = cl.vec/max.Carapace.length)

## if we wanted to set the proportions of net positions equivalent
## otherwise set to the proportion observed in the data
##given.values <- c("fmesh80mm_Net.positionpos2" = 1/4,
##                  "fmesh80mm_Net.positionpos3" = 1/4,
##                  "fmesh80mm_Net.positionpos4" = 1/4
##                  )

##cl.effect <- effect("Carapace.length", glmm.alt.covar, xlevels = xlevels, offset = 0, gi
cl.effect <- effect("prop.Carapace.length", glmm.alt.covar.catchdur, xlevels = xlevels, of
```

Plot the effect of carapace length with the other variables set to their mean in the data (Figure 5).

```
with(count.df, plot(Carapace.length, prop.80, ylim = c(0, 1), pch = 19,
                    xlab = "Carapace length (mm)",
                    ylab = "Proportion (N80mm/(N70mm + N80mm))",
                    bty = "L",
                    cex = 1/5 * log(N)))
```

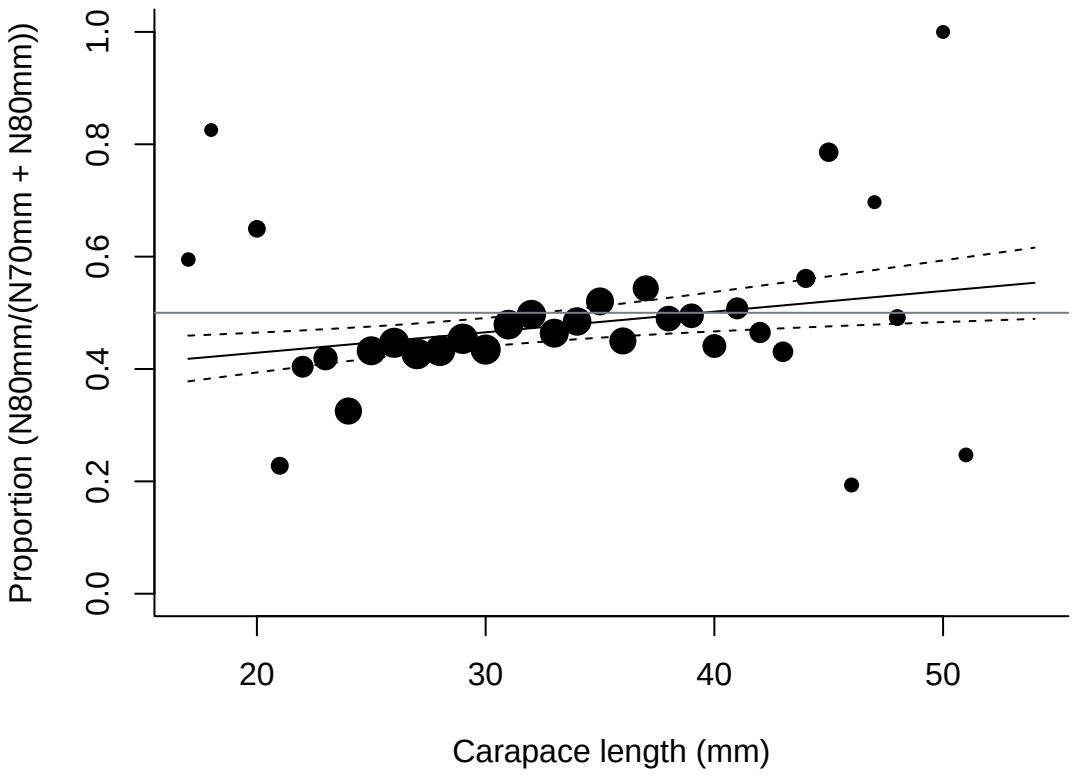


Figure 5: Predicted proportion from binomial GLMM with covariates. Note in the predictions the bulk weights are set to their mean and the net positions to their proportional occurence in the data.

```
## effects prediction
lines(cl.vec, plogis(cl.effect$fit[, 1]))
lines(cl.vec, plogis(cl.effect$lower[, 1]), lty = 2)
lines(cl.vec, plogis(cl.effect$upper[, 1]), lty = 2)
abline(h = 0.5, col = "slategrey")
```

```
## Note the wide confidence intervals
```

Finally test length effect in covariate model

```
glmm.alt.covar.catchdur.nolength <- glmer(cbind(mesh80mm_Count, mesh70mm_Count) ~
                                          log.mesh80mm_Total.catch + log.mesh70mm_Total.cat
                                          fmesh80mm_Net.position +
                                          poly(dec.hr, 1) +
```

```
                                          (1 | fHAUL),
                                          offset =
                                          log(mesh80mm_Overall.Sampling.Ratio /
                                              mesh70mm_Overall.Sampling.Ratio),
                                          family = binomial,
                                          data = neph.7080.cast,
                                          control=glmerControl(optimizer="bobyqa"))

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv,
:  Model failed to converge with max|grad| = 0.0073401 (tol = 0.001, component 2)

## likelihood ratio test
anova(glmm.alt.covar.catchdur.nolength, glmm.alt.covar.catchdur)

## Data: neph.7080.cast
## Models:
## glmm.alt.covar.catchdur.nolength: cbind(mesh80mm_Count, mesh70mm_Count) ~ log.mesh80mm_
## glmm.alt.covar.catchdur.nolength:     log.mesh70mm_Total.catch + fmesh80mm_Net.position
## glmm.alt.covar.catchdur.nolength:     1) + (1 | fHAUL)
## glmm.alt.covar.catchdur: cbind(mesh80mm_Count, mesh70mm_Count) ~ log.mesh80mm_Total.cat
## glmm.alt.covar.catchdur:     log.mesh70mm_Total.catch + prop.Carapace.length + fmesh80m
## glmm.alt.covar.catchdur:     poly(dec.hr, 1) + (1 | fHAUL)
##                                   Df    AIC    BIC  logLik deviance  Chisq
## glmm.alt.covar.catchdur.nolength  8 1417.9 1448.5 -700.97   1401.9
## glmm.alt.covar.catchdur           9 1411.6 1446.0 -696.79   1393.6 8.3654
##                                   Chi Df Pr(>Chisq)
## glmm.alt.covar.catchdur.nolength
## glmm.alt.covar.catchdur               1   0.003824 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## significant effect of carapace length
```

3 Economic section

```
curve(0.00032 * x^3.21, from = 0, to = 60, xlab = "Carapace length (mm)",
      ylab = "Total weight (g)")

npkilo.breaks <- c(1, 11, 16, 21, 31, 41, 50)
wt.breaks <- (1/ npkilo.breaks) * 1e3
abline(h = wt.breaks, col = "grey")

## corresponding length cut-offs
lt.breaks = exp(log(wt.breaks / 0.00032)/3.21)
abline(v = lt.breaks, col = "grey")
```

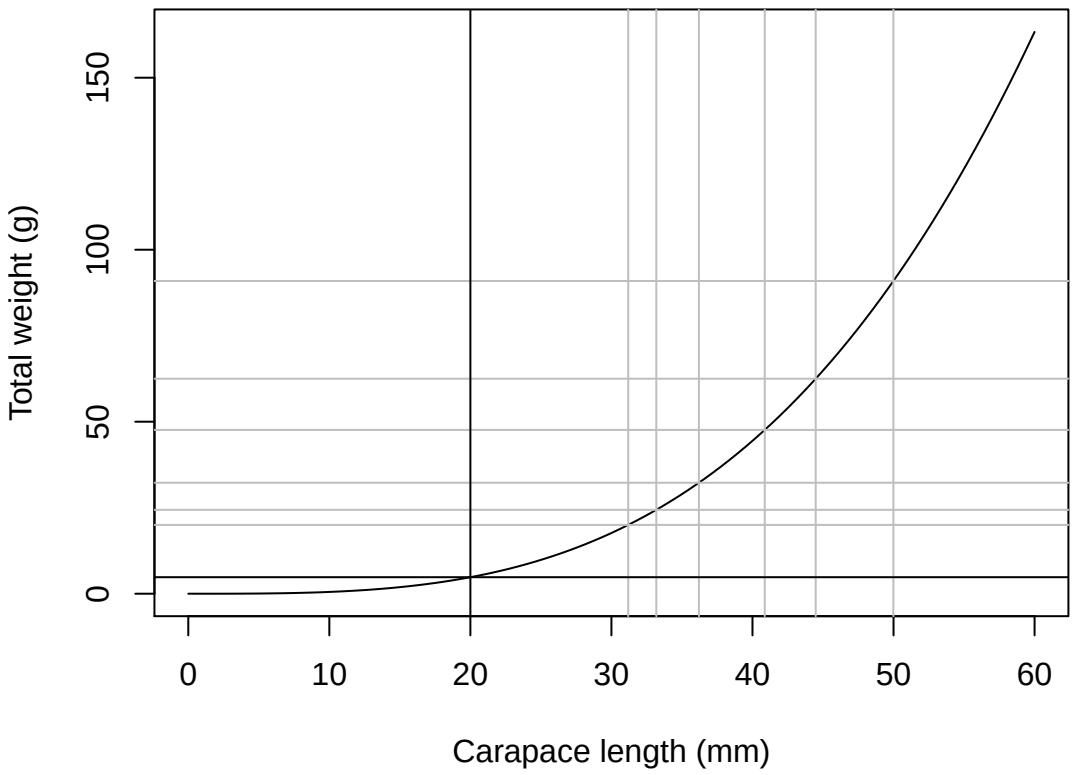


Figure 6: Length - weight relationship with horizontal price bands converted to length-based price bands. Price band in black denotes below Minimum Conservation Reference Size prawns.

```
## below MCRS band
abline(v = 20)
abline(h = 0.00032*20^3.21)
```

Simulate distributions of retained catches with different means

```
sim.means <- c(24, 27, 33)

## get the vector of raised counts for the whole Our Lass trip
neph.lengths <- with(subset(neph.dat, Mesh.Size%in%c("70mm", "80mm")),
                     rep(Carapace.length, times = round(Raised.count.)))

length(neph.lengths) ## raised number of prawns caught


## [1] 206009
```

```r
(mean.cl <- mean(neph.lengths)) ## mean carapace length below


## [1] 30.09305


## differences in the mean observed length and simulated means
## plus 0.5 to account for nearest mm below
sim.diff <- mean.cl - (sim.means + 0.5)
```

```r
## plot the simulated distributions of retained catches
hist.orig <- hist(neph.lengths, breaks = seq(0.5, 60, by = 1), plot = FALSE)
##hist.orig <- hist(neph.lengths, breaks = seq(1, 60, by = 1), plot = FALSE)
cl.mids <- hist.orig$mids
count.sim <- matrix(hist.orig$counts, nrow = 1)

plot(hist.orig$mids, hist.orig$counts, type = "l", xlim = c(10, 50),
     xlab = "Carapace length (mm)", ylab = "Count per 1 mm bin")
##
for(i in 1:length(sim.means)){
  hist.sim <- hist(neph.lengths - sim.diff[i],
                   breaks = seq(0.5, 60, by = 1), plot = FALSE)
                   ##breaks = seq(1, 60, by = 1), plot = FALSE)
  lines(hist.sim$mids, hist.sim$counts, lty = 1 + i, col = 1+i)
  count.sim <- rbind(count.sim, hist.sim$counts)
}

legend("topright", legend =
       c("Original (30.09mm)", "24mm mean",
         "27mm mean", "33mm mean"),
       lty = c(1:4), col = 1:4, bty = "n")
```

```r
##
rownames(count.sim) <- c("original", "mm.24", "mm.27", "mm.33")
```

Per-haul variables

```r
## count per haul (13 hauls)
count.phaul.sim <- count.sim / (13) ## note this is the sum for the 70 and 80

## get predicted prawn weight per length bin in kgs
wt.mids <- (0.00032 * cl.mids^3.21) / 1e3

## get total weight per haul
wt.phaul.sim <- t(apply(count.phaul.sim, 1, FUN = function(z){z * wt.mids}))

## predicted price per length class
## note 20mm CL included here
wt.cuts <- cut(wt.mids, breaks = c(wt.breaks, 0.00032*20^3.21, 0) / 1e3)
```

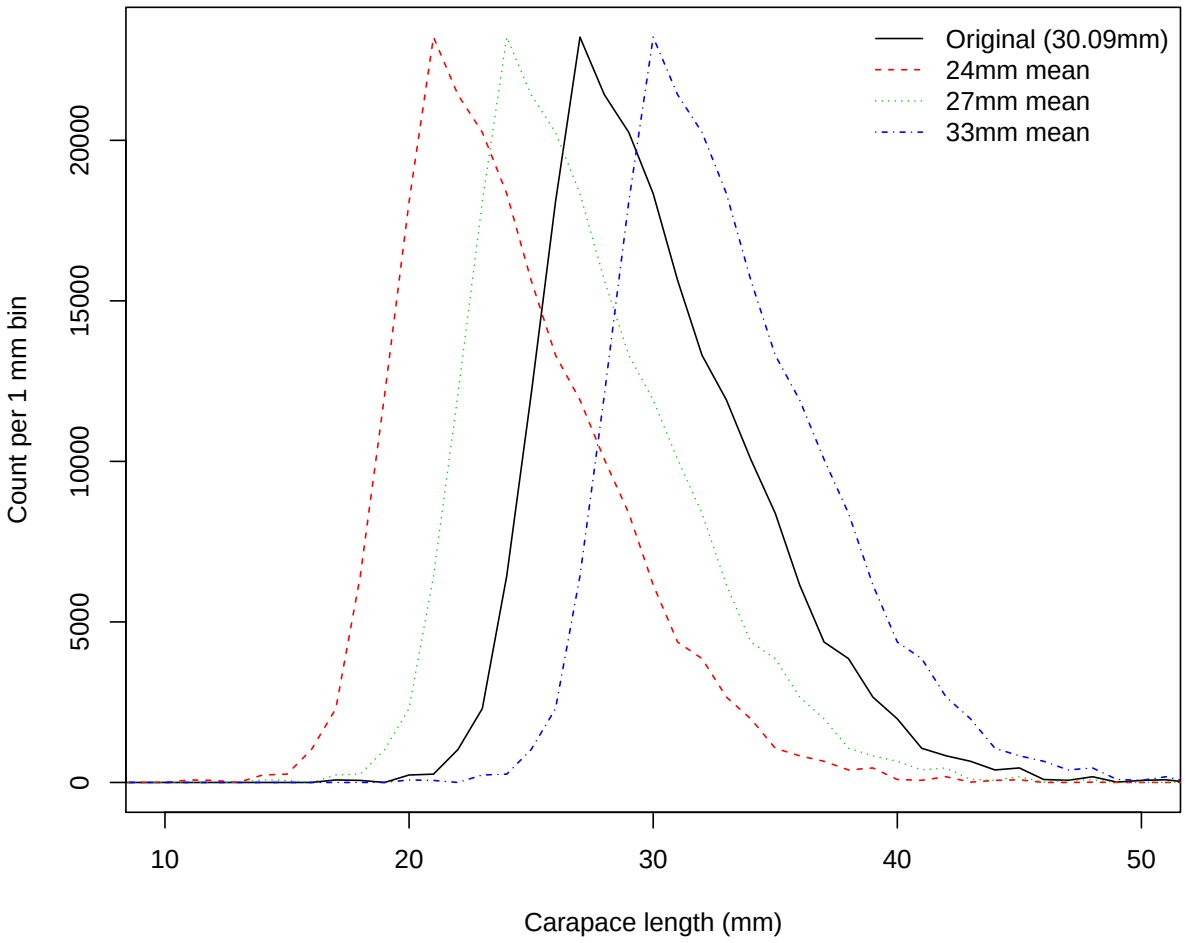


Figure 7: Simulated distributions of retained catches.

```
price.df <-  data.frame(wt.bin = unique(wt.cuts),
                        price = c(-0.2, 1.90, 4.75, 5.35, 7.75, 10.75, 13, 13))

price.df


##            wt.bin price
## 1      (0,0.0048] -0.20
## 2   (0.0048,0.02]  1.90
## 3   (0.02,0.0244]  4.75
## 4 (0.0244,0.0323]  5.35
## 5 (0.0323,0.0476]  7.75
## 6 (0.0476,0.0625] 10.75
## 7 (0.0625,0.0909] 13.00
## 8      (0.0909,1] 13.00


## prices per length class bin
price.bin.df <- merge(data.frame(wt.bin = wt.cuts), price.df)
price.bin.df$Carapace.length <- cl.mids

## value per haul
value.phaul.sim <- t(apply(wt.phaul.sim, 1, FUN = function(z){z * price.bin.df$price}))
```

Finally, split the variables at length between 70 and 80mm.

```
## get predicted proportions in 80mm for given carapace length mid-points
xlevels <- list(prop.Carapace.length = cl.mids/max.Carapace.length)

## set the proportions of net positions equivalent
given.values <- c("fmesh80mm_Net.positionpos2" = 1/4,
                  "fmesh80mm_Net.positionpos3" = 1/4,
                  "fmesh80mm_Net.positionpos4" = 1/4
                  )

cl.effect <- effect("prop.Carapace.length", glmm.alt.covar.catchdur, xlevels = xlevels, of

p80 <- plogis(cl.effect$fit[, 1])

## plot(cl.mids, p80, ylim = c(0, 1))

## split out 70 and 80
## count
count.phaul.sim.80 <- t(apply(count.phaul.sim, 1, FUN = function(z){z * p80}))
count.phaul.sim.70 <- t(apply(count.phaul.sim, 1, FUN = function(z){z * (1 - p80)}))

## weight
wt.phaul.sim.80 <- t(apply(wt.phaul.sim, 1, FUN = function(z){z * p80}))
wt.phaul.sim.70 <- t(apply(wt.phaul.sim, 1, FUN = function(z){z * (1 - p80)}))

## value
```

```
value.phaul.sim.80 <- t(apply(value.phaul.sim, 1, FUN = function(z){z * p80}))
value.phaul.sim.70 <- t(apply(value.phaul.sim, 1, FUN = function(z){z * (1 - p80)}))
```

Plot the counts, weights and value per length bin split by 70 and 80mm (Figure 8).

```
par(mfrow = c(2, 2), mar = c(2, 3, 1, 1), oma = c(2, 2, 1, 1))
## Count
matplot(cl.mids, t(count.phaul.sim.80),
        type = "l", col = "darkblue",
        xlim = c(10, 50), ylim = c(0, 1e3))
matlines(cl.mids, t(count.phaul.sim.70), type = "l", col = "red1")
mtext(side = 2, line = 2, text = "Count per 1mm bin")
## to demonstrate same retention across scenarios
## use xlim = c(15, 40) and abline(v = c(23.3, 26.3, 29.3))
## Weight
matplot(cl.mids, t(wt.phaul.sim.80),
        type = "l", col = "darkblue",
        xlim = c(10, 50), ylim = c(0, 20))
matlines(cl.mids, t(wt.phaul.sim.70), type = "l", col = "red1")
mtext(side = 2, line = 2, text = "Weight (kg) per 1mm bin")
## Value
matplot(cl.mids, t(value.phaul.sim.80),
        type = "l", col = "darkblue",
        xlim = c(10, 50), ylim = c(0, 110))
matlines(cl.mids, t(value.phaul.sim.70), type = "l", col = "red1")
mtext(side = 2, line = 2, text = "Value (euro) per 1mm bin")
##
plot.new()
legend("center", legend = c("70mm", "80mm", NA, "Original (30.09mm)", "24mm mean",
                    "27mm mean", "33mm mean"),
       lty = c(1,1,NA, 1:4),
       col = c("red1", "darkblue", NA, rep("darkblue", 4)),
       bty = "n"
       )
```

Summary table (as in BIM report Table 3)

```
## calculate the resulting mean sizes per length class
##apply(count.phaul.sim, 1, FUN = function(z){sum(cl.mids * z) / sum(z)})
mean.cl.70 <- apply(count.phaul.sim.70, 1, FUN = function(z){sum(cl.mids * z) / sum(z)})
mean.cl.80 <- apply(count.phaul.sim.80, 1, FUN = function(z){sum(cl.mids * z) / sum(z)})

## data frame for predictions
na.vec <- rep(NA, 8)
sim.order <- c("original", "mm.33", "mm.27", "mm.24")

pred.df <- data.frame(Mesh.Size = rep(c(70, 80), each = 4),
                      Mean.CL = rep(sim.order, 2),
                      Mean.CL.mesh = c(mean.cl.70[sim.order], mean.cl.80[sim.order]),
                      c.less.mcrs = na.vec,
```

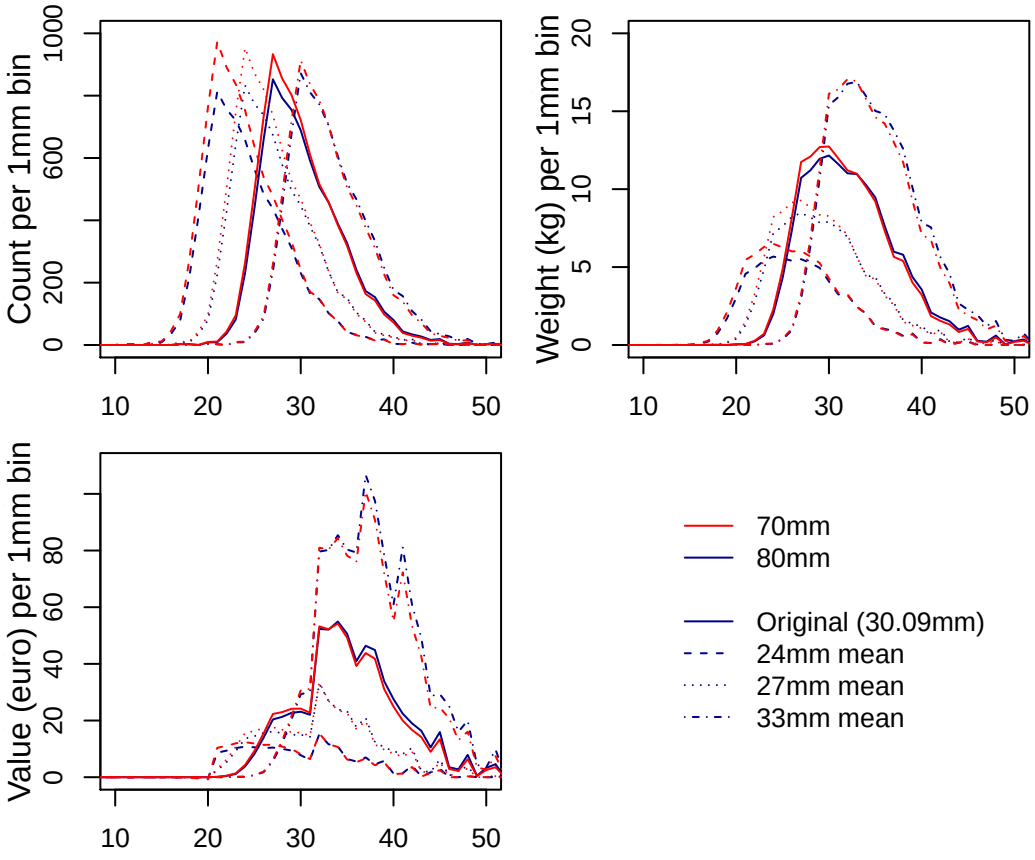


Figure 8: Simulated counts, weight and value by: length class, mesh size and simulated scenario (24mm mean catch, 27mm mean catch, 30mm mean catch (original), 33mm mean catch).

```
                        c.greater.mcrs = na.vec,
                        total.catch = na.vec,
                        v.less.mcrs = na.vec,
                        v.greater.mcrs = na.vec,
                        total.value = na.vec,
                        stringsAsFactors = FALSE)
##

for(i in 1:dim(pred.df)[1]){
  print(i)
  wt <- get(paste("wt.phaul.sim.", pred.df$Mesh.Size[i], sep = ""))
  value <- get(paste("value.phaul.sim.", pred.df$Mesh.Size[i], sep = ""))
  ## catch
  pred.df$c.less.mcrs[i] <- sum(wt[pred.df$Mean.CL[i], cl.mids < 20])
  pred.df$c.less.mcrs[i] <- round(pred.df$c.less.mcrs[i], 2)
  pred.df$c.greater.mcrs[i] <- sum(wt[pred.df$Mean.CL[i], cl.mids >= 20])
  pred.df$c.greater.mcrs[i] <- round(pred.df$c.greater.mcrs[i], 2)
  pred.df$total.catch[i] <- sum(wt[pred.df$Mean.CL[i], ])
  pred.df$total.catch[i] <- round(pred.df$total.catch[i], 2)
  ## value
  pred.df$v.less.mcrs[i] <- sum(value[pred.df$Mean.CL[i], cl.mids < 20])
  pred.df$v.less.mcrs[i] <- round(pred.df$v.less.mcrs[i], 2)
  pred.df$v.greater.mcrs[i] <- sum(value[pred.df$Mean.CL[i], cl.mids >= 20])
  pred.df$v.greater.mcrs[i] <- round(pred.df$v.greater.mcrs[i], 2)
  pred.df$total.value[i] <- sum(value[pred.df$Mean.CL[i], ])
  pred.df$total.value[i] <- round(pred.df$total.value[i], 2)
}


## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
```

```
library(xtable)
print(xtable(pred.df, digits = 2, align = "lccccccccc"), include.rownames = FALSE)
```

Mesh.Size	Mean.CL	Mean.CL.mesh	c.less.mcrs	c.greater.mcrs	total.catch	v.less.mcrs	v.greater.mcrs	total.value
70.00	original	29.96	0.02	152.80	152.82	-0.00	628.79	628.79
70.00	mm.33	32.96	0.00	200.60	200.60	0.00	1113.07	1113.07
70.00	mm.27	26.97	0.25	112.89	113.14	-0.05	353.73	353.68
70.00	mm.24	23.97	3.46	77.50	80.96	-0.69	190.95	190.26
80.00	original	30.23	0.02	150.65	150.67	-0.00	648.12	648.12
80.00	mm.33	33.23	0.00	206.10	206.10	0.00	1187.36	1187.36
80.00	mm.27	27.23	0.20	106.90	107.11	-0.04	350.37	350.33
80.00	mm.24	24.24	2.78	70.86	73.65	-0.56	182.10	181.55