# Simulate covered codend data with sampling fractions

Mollie Brooks

12 Aug 2020

This is a simulation study to test how subsampling can be handled in covered codend analyses. The analysis is based on Millar 1994. We assume that a covered codend study was conducted and the codend has a given selection pattern. We assume that different fractions of the cover and codend were sampled. Then we fit a model with an offset to account for the sampling fractions or raising factors.

```
library(boot) #for inv.logit
library(ggplot2); theme_set(theme_bw())
library(plyr)
library(selfisher)
set.seed(11)
```

## Simulate data

Set up parameters

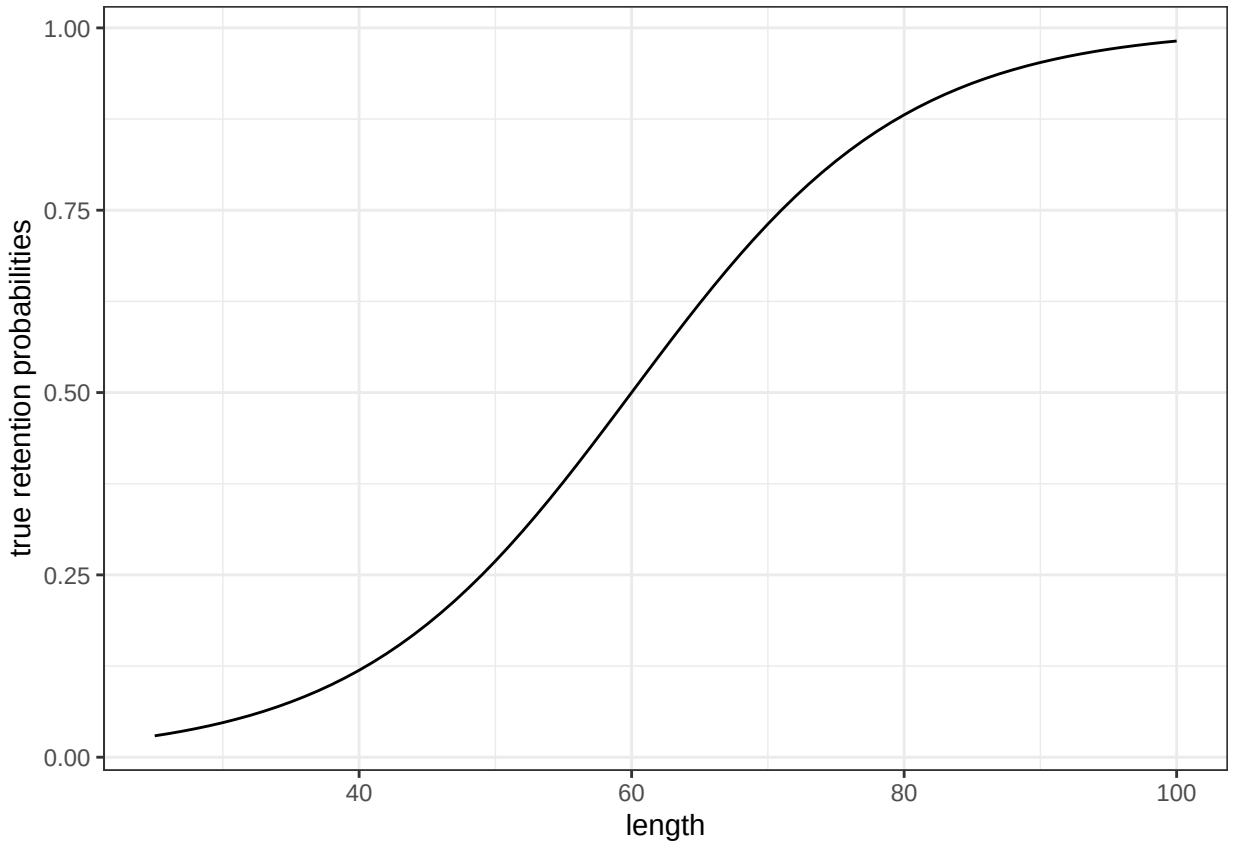
```
length=25:100 #measured length classes
pars1=c(-6, .1) #selection curve parameters for codend
nhaul=10 #number of hauls with each type of gear
q_codend=runif(nhaul, 0.6, 0.8) #sampling fractions for codend
q_cover=runif(nhaul, 0.4, 0.6) #sampling fractions for cover
nfish=5 # avg number of fish encountered in each length class in each haul
```

Organize the model parameters. Simulate the total number of fish of each length class by haul that are encountered (but not necessarily caught).

```
bdat=expand.grid(length = length, haul=1:nhaul)

bdat=transform(bdat,
            sampling_codend =q_codend[haul],
            sampling_cover =q_cover[haul],
            r1=inv.logit(pars1[1]+ pars1[2]* length),
            simtotal=rpois(nrow(bdat), nfish)
            )
```

Plot the true selection curve.

```
ggplot(bdat)+
    geom_line(aes(length, r1))+
    ylab("true retention probabilities")
```




Simulate retained fish and sampled fish separately.

```
dat=transform(bdat, length= length, haul=haul,
    codend=rbinom(nrow(bdat), size= simtotal, prob=r1))

dat= transform(dat,
                    cover= simtotal-codend,
                    codend_samp=rbinom(nrow(bdat), size= codend, prob=sampling_codend),
                    cover_samp=rbinom(nrow(bdat), size= simtotal-codend, prob=sampling_cover))
```

## Organize data

Here we do a few transformaitons:

- calculate the ratio of the sampling fractions to use as an offset
- calculate the binomial response (proportion and total) for comparing test 1 vs test2
- calculate the true ratio of the reteiontion probabilities

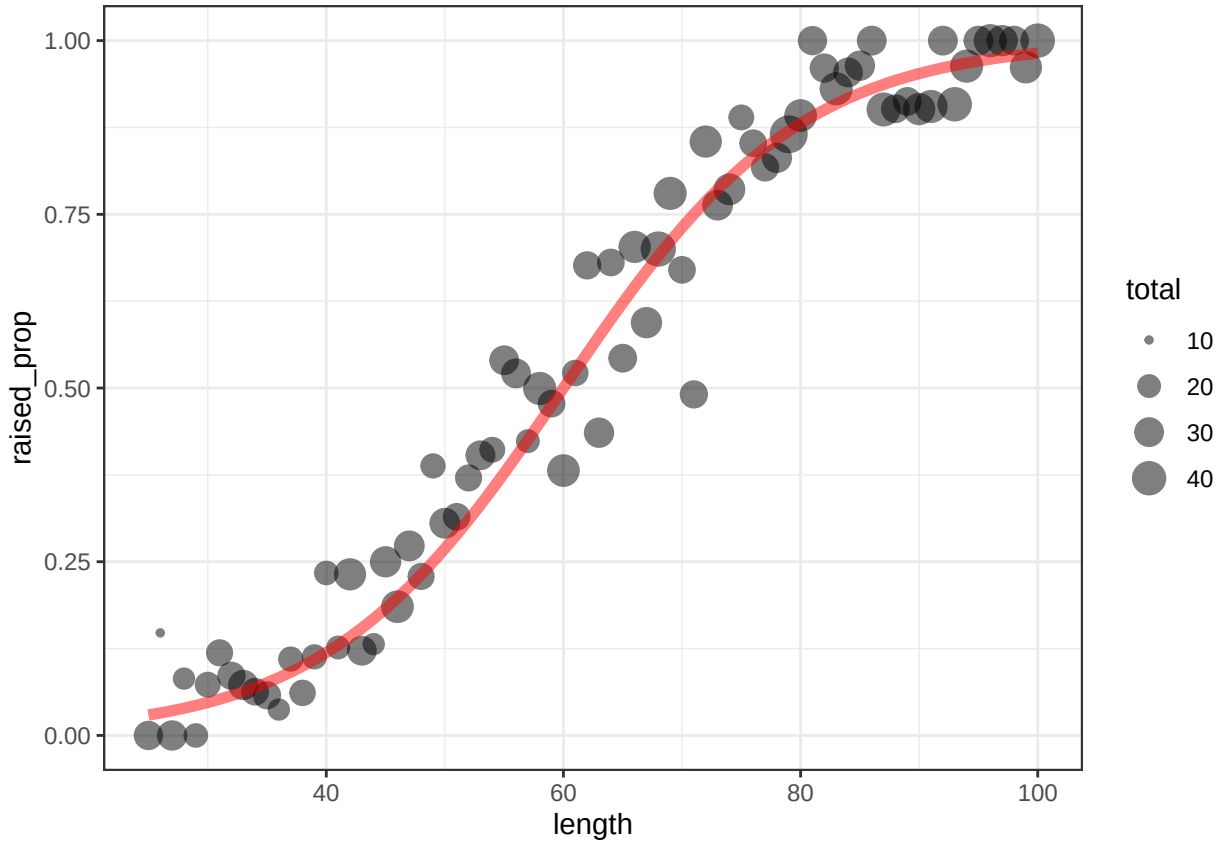
```
dat=transform(dat,
      q_ratio=sampling_codend/sampling_cover,
      prop=codend_samp/(codend_samp+cover_samp),
      total=codend_samp+cover_samp,
      true_selection_prob=r1)

dat=subset(dat, !is.na(prop))
```

Aggregate the data across hauls for plotting.

```
sumdat=ddply(dat, ~length + true_selection_prob, summarize,
      prop=sum(codend_samp)/sum(total),
      total=sum(total),
      raised_prop=sum(codend_samp/sampling_codend)/
                        (sum(codend_samp/sampling_codend)+sum(cover_samp/sampling_cover)),
      raised_total=sum(codend_samp/sampling_codend)+sum(cover_samp/sampling_cover))
```

Plot aggregated data.

```
p1=ggplot(sumdat, aes(x=length))+
    geom_point(aes(size=total, y=raised_prop), alpha=.5)+
    geom_line(data=sumdat, aes(x=length, y=true_selection_prob), colour="red", alpha=.5, lwd=2)
p1
```




## Model

Fit the model.

```
m1=selfisher(prop~offset(log(q_ratio))+length, total=total, dat, haul=haul)
```

Calculate predictions from the fitted model.

```
newdata=data.frame(length=unique(dat$length))
newdata=transform(newdata,
    total=1,
    haul=NA,
    q_ratio=1
)
```

```
newdata$est_selection_prob=predict(m1, newdata=newdata, type="selection")
```

## Bootstrap confidence intervals

### Code for Mac and Linux

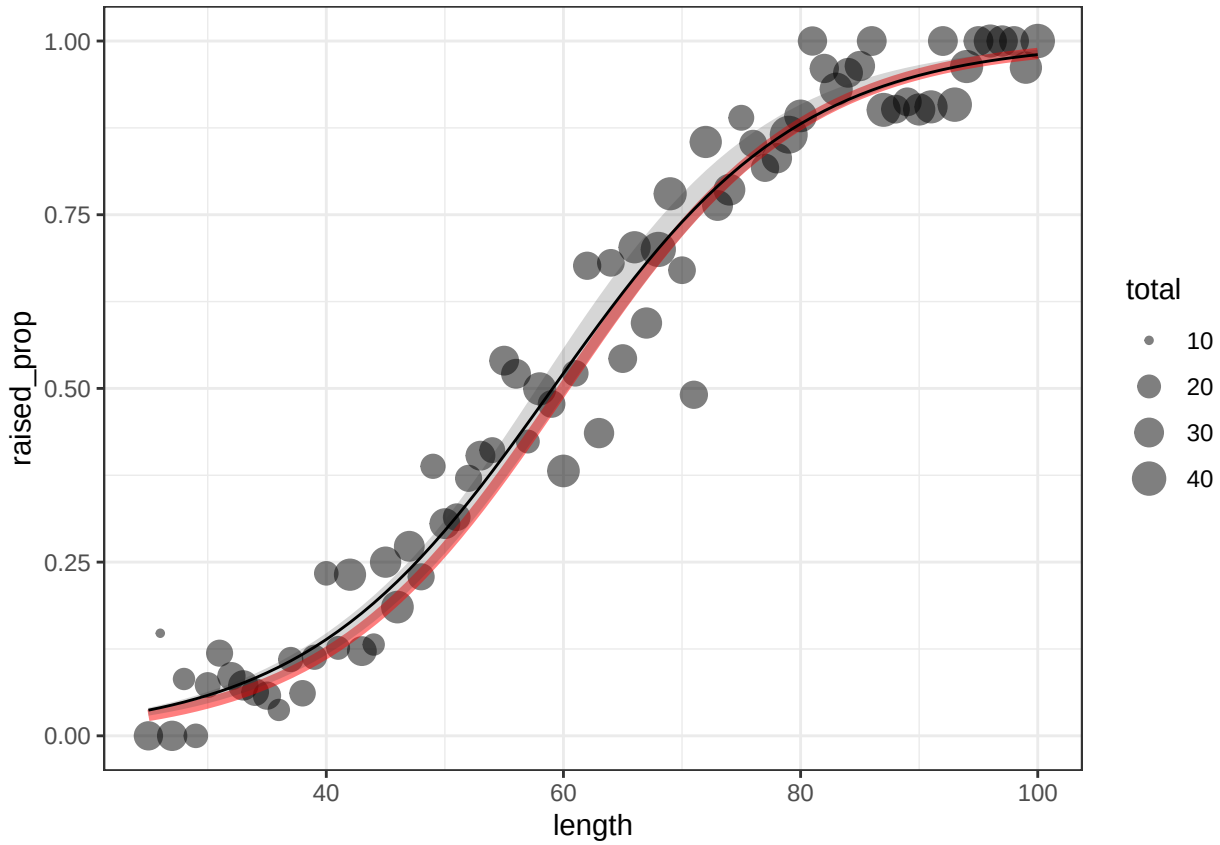
```
bs=bootSel(m1, nsim=10, parallel = "multicore", ncpus = 4,
        FUN=function(mod){predict(mod, newdata=newdata, type="selection")})
```

### Code for Windows

```
library(snow)
ncpus = 4
cl = makeCluster(rep("localhost", ncpus)) clusterExport(cl, "newdata")
bs = bootSel(m1, nsim=1000, parallel = "snow", cl=cl,
    FUN=function(mod){predict(mod, newdata=newdata, type="selection")})
stopCluster(cl)
```

### Organize and plot bootstraps

```
#apply quantile function to bootstraps and match them with the newdata used for predictions
quants=apply(bs$t, 2, quantile, c(0.025, 0.5, 0.975))

newdata[,c("lo", "mid", "hi")]=t(quants)

p1+
    geom_ribbon(data=newdata, aes(ymin=lo, ymax=hi), alpha=0.2)+
    geom_line(data=newdata, aes(y=est_selection_prob))
```


raised_prop
1.00
0.75
0.50
0.25
0.00
40
60
80
100
length
total
10
20
30
40