SELECT package demonstration  
Paired-haul relative selectivity

## Summary

**This case study demonstrates:**

* Reading data from separate gear-specific files and merging them into a single data frame.
* Working with sub-sampled data.
* Fitting catch share (relative selectivity) curves using splines via the SplineSELECT function.
* Fitting catch share curves using an improved version of the averaged polynomial approach via the PolySELECT function (see bottom part of the code).
* Using the bootstrap function bootSELECT to estimate the uncertainty in the catch share curve.
* Using the permutation function permSELECT to quantify the evidence for a length effect on the catch share. Two different test statistics are used, deviance explained and ratio of commercial to non-commercial sizes.

### Data source

The data are for school prawn relative selectivity in penaeid trawls from the experiments conducted by Broadhurst et al., (2018, T45 side panels improve penaeid-trawl selection. Fisheries Research, 204: 8-15). The dats used here are from the twin-trawl experiment that fished two trawls differing in square mesh sizes used for the side panel, 32 mm or 35 mm.

### Load required packages

require(tidyverse)  
require(mgcv)  
require(SELECT)  
require(readxl) #This package is installed with tidyverse  
nsim=1000 #For bootstrapping and permutation tests, set to >=1000 in practice

### Read in the data

The data for each mesh size are in individual xlsx files and require merging.

GearA.df=read\_excel("SchoolPrawnLenFreqs.xlsx", sheet = "32 square trawl")  
GearB.df=read\_excel("SchoolPrawnLenFreqs.xlsx", sheet = "35 square trawl")  
  
GearA.df = GearA.df |> rename(Haul=Day, nA=No.school, sfA=Sf.school)  
GearB.df = GearB.df |> rename(Haul=Day, nB=No.school, sfB=Sf.school)

### Merge the separate dataframes to create a dataframe for the twin tows

Note the conversion from sub-sampling scaling factors to sampling fractions and removal of CLs that are outside of the range of measured data.

#Merge 32mm and 35mm gears with identical haul ID (i.e., twin tows)  
Pairs.df = inner\_join(GearA.df,GearB.df,by=c("Haul","CL"))  
#Convert scaling factors to sampling fractions  
Pairs.df = Pairs.df |>  
 transform(qA=1/sfA,qB=1/sfB) |> filter(CL>=5 & CL<=25)

### Define variable names

names(Pairs.df)

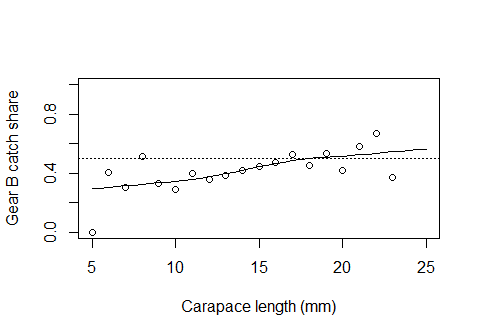
## [1] "Haul" "CL" "nA" "sfA" "nB" "sfB" "qA" "qB"

var.names=c("CL","nA","nB")  
q.names=c("qA","qB")

### Define a prediction function to be used with the bootstrap

This fit used the SplineSELECT defaults, but more generally one may want to try other values of k, say 5 and 10.

#Define the bootstrap prediction function  
CLseq=seq(5,25,0.5) #Carapace lengths to use for predn  
PrednFnc=function(data,var.names) {  
 SplineFit=SplineSELECT(data,var.names,q.names,bs="tp",  
 quasi=T,sumHauls=T,k=7,q.ODadjust = T) #Defaults  
 predict(SplineFit,newdata=data.frame(CL=CLseq),type="response") }  
#Check that it works  
predn=PrednFnc(Pairs.df,var.names)  
  
#Plot predictions against observed proportions  
Tots.df=Raw2Tots(Pairs.df,var.names,q.names) |>   
 transform(lgth=CL, y=nB/(nA+nB))  
plot(y~CL,data=Tots.df,ylim=c(0,1),xlab="Carapace length (mm)",  
 ylab="Gear B catch share")  
points(CLseq,predn,type="l")  
abline(h=0.5,lty=3)



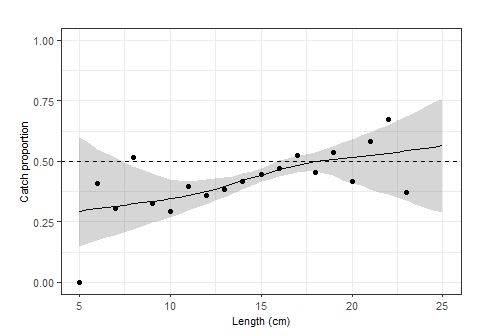
### Do the bootstrap of the catch share curve

BootPreds=bootSELECT(Pairs.df,var.names,PrednFnc,haul="Haul",nsim=nsim,  
 paired=T,verbose=F)

##   
## Bootstrap successfully completed

BootPlot(BootPreds,CLseq,predn,Data=Tots.df) +  
 geom\_hline(yintercept=0.5,linetype="dashed")

## Warning: Removed 2 rows containing missing values or values outside the scale range  
## (`geom\_point()`).



### Define a goodness of fit function to be used with the permutation test

#Define the deviance explained function  
DevExplained=function(data,var.names) {  
 SplineFit=SplineSELECT(data,var.names,q.names,bs="tp",  
 quasi=T,sumHauls=T,k=7,q.ODadjust = T)   
 summary(SplineFit)$dev.expl }  
#Check that it works  
ObsDev=DevExplained(Pairs.df,var.names)  
cat("Proportion of deviance explained is",ObsDev,"\n")

## Proportion of deviance explained is 0.7180213

PermDev=permSELECT(Pairs.df,var.names,DevExplained,haul="Haul",nsim=nsim,  
 paired=T,verbose=F)

##   
## Permutations successfully completed

#Proportion of permuted gof values greater than the observed  
cat("The p-value for a CL effect is",mean(PermDev>ObsDev))

## The p-value for a CL effect is 0.024

### Define a commercial vs non-commercial ratio function for 2nd permutation test

Commercial is >=15 CL.

#Define the deviance explained function  
Ratio=function(data,var.names) {  
 LgthTotals=Raw2Tots(data,var.names,q.names)   
 Totals=apply(LgthTotals,2,sum)[2:3]  
 Propn=apply(LgthTotals |> filter(CL>=15),2,sum)[2:3]/Totals  
 PropnRatio=Propn[2]/Propn[1] }  
#Check that it works  
ObsRatio=Ratio(Pairs.df,var.names)  
cat("Ratio of commercial to non-commercial is",ObsRatio,"\n")

## Ratio of commercial to non-commercial is 1.194963

PermRatio=permSELECT(Pairs.df,var.names,Ratio,haul="Haul",nsim=nsim,paired=T,  
 verbose=F)

##   
## Permutations successfully completed

#Proportion of permuted gof values greater than the observed  
cat("The p-value for a gear effect on ratio is",mean(PermRatio>ObsRatio))

## The p-value for a gear effect on ratio is 0

## Demonstration of the averaged-polynomial catch curve

**This is included for completeness - we recommend using the spline approach instead of the averaged polynomial approach.**

**The default use of PolySELECT includes improvements to the Herrmann et al. (2017) implementation that reduces the tendency of the averaged polynomial to overfit. See PolySELECT documentation for details.**

### Averaged polynomial fits

require(MuMIn) #For MuMIn::dredge  
options(na.action = "na.fail") #To ensure dredge terminates if there are NAs  
#The default averaged-polynomial fit  
PolyFit=PolySELECT(Pairs.df,var.names,q.names)   
#The averaged-polynomial fit used by Herrmann et al. (2017)  
HerrmannFit=PolySELECT(Pairs.df,var.names,q.names,q.ODadjust=F,quasi=F,All=T)

PolyPredn=predict(PolyFit$avg.fit,newdata=data.frame(CL=CLseq),  
 type="response")  
HerrmannPredn=predict(HerrmannFit$avg.fit,newdata=data.frame(CL=CLseq),  
 type="response")  
plot(y~CL,data=Tots.df,ylim=c(0,1),xlab="Carapace length (mm)",  
 ylab="Gear B catch share")  
points(CLseq,predn,type="l",col="blue") #Spline  
points(CLseq,PolyPredn,type="l",col="blue",lty=2) #PolySELECT  
points(CLseq,HerrmannPredn,type="l",col="red",lty=2) #Herrmann et al. (2017)  
legend("topleft",  
 legend=c("Spline","PolySELECT default","Herrmann et al. (2017)"),  
 col=c("blue","blue","red"),lty=c(1,2,2),bty="n")  
abline(h=0.5,lty=3)

