# ASMFC MSE Workshop: First MSE

Gavin Fay

August 2021

### Introduction to MSE

Here we will work through a simple example of applying MSE. Later this week we will take a more modular approach to implementing MSEs, but we walk through steps here to give you a flavor for how the pieces work and can be put together.

This lab is based (heavily) on tutorial by Katell Hamon & Jan-Jaap Poos, published in Chapter 3 of Edwards & Dankel (eds): *"Management Science in Fisheries, An introduction to simulation-based methods"*. All errors below are completely the fault of GF.

We consider a fishery for a population of *Sebastes electronicus*:
* the operating model population dynamics are governed by a logistic (Schaefer) production model.
* Data available from the fishery are the catch (known without error), and a biomass index.
* We will apply a simple empirical harvest control rule to demonstrate the MSE, and use a small set of performance statistics to compare among alternative versions of the HCR.

There are plenty of places where additional complexity can be built in to this example. We encourage you to play around with adding functionality of interest. Some options could include adding a model-based control rule, changing the dynamics of the operating model, applying the control rule every 3 years instead of every year, etc.
* However, you should be able to walk through this tutorial without tweaks if you just want to get a feel for how things work.

We assume you have installed `R` on your computer and have an appropriate text editor or development environment (e.g. `Rstudio`).
First we install some libraries in `R` that we will use later.
*(If you do not have these packages installed then run the currently commented out lines that call 'install.packages()')*

```
#install.packages('ggplot2')
library(ggplot2)
library(Hmisc)
```

```
## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library(mvtnorm)
```

**The Operating Model** The population dynamics for the operating model (the 'real' dynamics) are governed by the equation:

$$B_{y+1} = B_y + B_y * r * (1 - \frac{B_y}{K}) - C_y$$

where $B_y$ is the biomass in year $y$, $C_y$ is the catch in year $y$, $r$ is the population intrinsic growth rate, and $K$ is the population carrying capacity.

We assume that the population is at carrying capacity in the first year of our simulation (i.e. $B_1 = K$).

Our first task is to condition our operating model, that we will then use to perform the MSE projections.

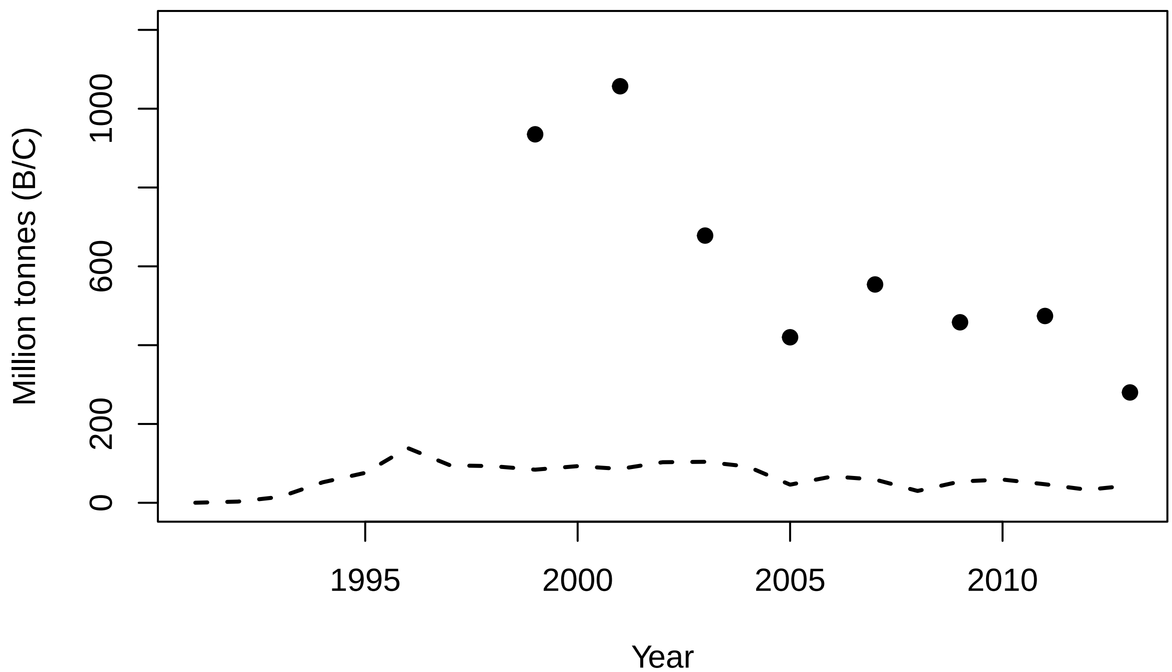
#####Specify input data and associated years

```
data.years <- 1991:2013
harvest <- c(0.1,3,15,52,76,139,95,93,84,93,86,103,104,
             92,46,67,59,30,54,59,47,33,44)
index <- c(NA,NA,NA,NA,NA,NA,NA,NA,935,NA,1057,NA,678,NA,
           420,NA,554,NA,458,NA,474,NA,280)
```

We create time series of the years, catches (harvest), and biomass index data for our historical period that are already available.

We can plot these:

```
plot(data.years,index, pch=19,xlab="Year",ylab="Million tonnes (B/C)",
     ylim=c(0,1200))
lines(data.years,harvest,lty=2,lwd=2)
```




We see that the biomass index has been declining.

Now we will create some functions to use as we develop the operating model.

First, the logistic production function:

```
schaefer <- function(B,C,K,r) {
  #function schaefer takes the current biomass, a catch,
  #and the model parameters to compute next year's biomass
```

```
  res <- B + B * r * (1 - B/K) - C
  return(max(0.001,res))  # we add a constraint to prevent negative biomass
}
```

Now a function to do the biomass projection:

```
dynamics <- function(pars,C,yrs) {
  # dynamics takes the model parameters, the time series of catch,
  # & the yrs to do the projection over

  # first extract the parameters from the pars vector (we estimate K in log-space)
  K <- exp(pars[1])
  r <- pars[2]

  # find the total number of years
  nyr <- length(C) + 1

  # if the vector of years was not supplied we create
  # a default to stop the program crashing
  if (missing(yrs)) yrs <- 1:nyr

  #set up the biomass vector
  B <- numeric(nyr)

  #intialize biomass at carrying capacity
  B[1] <- K
  # project the model forward using the schaefer model
  for (y in 2:nyr) {
    B[y] <- schaefer(B[y-1],C[y-1],K,r)
  }

  #return the time series of biomass
  return(B[yrs])

#end function dynamics
}
```

We are going to condition the operating model by estimating the parameters based on the historical biomass index data.

To do this we make a function that shows how well the current parameters fit the data, we assume that the observation errors around the true biomass are log-normally distributed.

```
# function to calculate the negative log-likelihood
nll <- function(pars,C,U) {  #this function takes the parameters, the catches, and the index data
  sigma <- pars[3]  # additional parameter, the standard deviation of the observation error
  B <- dynamics(pars,C)  #run the biomass dynamics for this set of parameters
  Uhat <- B   #calculate the predicted biomass index - here we assume an unbiased absolute biomass esti
  output <- -sum(dnorm(log(U),log(Uhat),sigma,log=TRUE),na.rm=TRUE)   #calculate the negative log-likel
  return(output)
#end function nll
}
```

Function to perform the assessment and estimate the operating model parameters
(i.e. to fit the logistic model to abundance data)

```r
assess <- function(catch,index,calc.vcov=FALSE,pars.init) {
  # assess takes catch and index data, initial values for the parameters,
  # and a flag saying whether to compute uncertainty estimates for the model parameters

  #fit model
  # optim runs the function nll() repeatedly with differnt values for the parameters,
  # to find the values that give the best fit to the index data
  res <- optim(pars.init,nll,C=catch,U=index,hessian=TRUE)

  # store the output from the model fit
  output <- list()
  output$pars <- res$par
  output$biomass <- dynamics(res$par,catch)
  output$convergence <- res$convergence
  output$nll <- res$value
  if (calc.vcov)
    output$vcov <- solve(res$hessian)

  return(output)
#end function assess
}
```

Now we have written the functions to do the calculations, we can run them and perform the assessment.

First define initial parameter vector for: log(K), r, sigma

```r
ini.parms <- c(log(1200), 0.1, 0.3)
```

Fit the logistic model to data:

```r
redfish <- assess(harvest,index,calc.vcov=TRUE,ini.parms)
redfish
```

```
## $pars
## [1] 7.25855230 0.06581353 0.17045412
##
## $biomass
##  [1] 1420.1990 1420.0990 1417.1056 1402.3088 1351.4713 1279.7757 1149.1036
##  [8] 1068.5396  992.9529  928.6124  856.7668  793.1370  713.1845  632.5512
## [15]  563.6396  540.0126  495.0391  457.2628  447.6675  413.8430  374.1428
## [22]  345.2795  329.4789  302.1325
##
## $convergence
## [1] 0
##
## $nll
## [1] -2.802687
##
## $vcov
##               [,1]          [,2]          [,3]
## [1,]  4.854417e-03 -1.825539e-03  4.901250e-06
## [2,] -1.825539e-03  7.307203e-04 -1.993010e-06
## [3,]  4.901250e-06 -1.993010e-06  1.815276e-03
```

Extract the maximum likelihood and parameter estimates

```
biomass.mle <- redfish$biomass
print(biomass.mle)
```

```
##  [1] 1420.1990 1420.0990 1417.1056 1402.3088 1351.4713 1279.7757 1149.1036
##  [8] 1068.5396  992.9529  928.6124  856.7668  793.1370  713.1845  632.5512
## [15]  563.6396  540.0126  495.0391  457.2628  447.6675  413.8430  374.1428
## [22]  345.2795  329.4789  302.1325
```

```
pars.mle <- redfish$pars
print(pars.mle)
```

```
## [1] 7.25855230 0.06581353 0.17045412
```

To obtain plausible alternatives for the parameters of the operating model, we will use the statistical uncertainty from the estimation by sampling parameter sets from the estimated variance-covariance matrix.

```
#define the number of iterations for the MSE
niter <- 500
#set up a storage matrix for our alternative parameter sets
pars.iter <- matrix(NA,nrow = niter, ncol=3)
colnames(pars.iter) <- c("logK","r","sigma")

# generate the sets of parameter values
for (i in 1:niter) {
  pars.iter[i,] <- mvtnorm::rmvnorm(1, mean = redfish$pars,
                       sigma = redfish$vcov)
}

# Now generate replicate model outputs
biomass.iter <- data.frame()
for (i in 1:niter) {
  #here we calculate the biomass trajectory for each of the above sampled parameter vectors
  biomass.iter <- rbind(biomass.iter,
                        data.frame(year = seq(min(data.years),
                                              max(data.years)+1),
                                   biomass = dynamics(pars.iter[i,], harvest),
                                   iter = i))
}
biomass.iter
```

```
##    year  biomass iter
## 1  1991 1464.8687    1
## 2  1992 1464.7687    1
## 3  1993 1461.7745    1
## 4  1994 1446.9522    1
## 5  1995 1395.9706    1
## 6  1996 1323.7489    1
## 7  1997 1192.0875    1
## 8  1998 1109.8619    1
## 9  1999 1032.3402    1
## 10 2000  965.8812    1
## 11 2001  891.8147    1
## 12 2002  825.8911    1
## 13 2003  743.6224    1
## 14 2004  660.6919    1
## 15 2005  589.5640    1
```

```
## 16  2006  563.8366    1
## 17  2007  516.7943    1
## 18  2008  477.0419    1
## 19  2009  465.5540    1
## 20  2010  429.8303    1
## 21  2011  388.3075    1
## 22  2012  357.7297    1
## 23  2013  340.2884    1
## 24  2014  311.3217    1
## 25  1991 1615.4000    2
## 26  1992 1615.3000    2
## 27  1993 1612.3011    2
## 28  1994 1597.3367    2
## 29  1995 1545.5421    2
## 30  1996 1470.3108    2
## 31  1997 1332.8296    2
## 32  1998 1240.5111    2
## 33  1999 1150.8221    2
## 34  2000 1070.6287    2
## 35  2001  981.7812    2
## 36  2002  900.2102    2
## 37  2003  801.7941    2
## 38  2004  702.4386    2
## 39  2005  615.0045    2
## 40  2006  573.3849    2
## 41  2007  510.6388    2
## 42  2008  455.6552    2
## 43  2009  429.4176    2
## 44  2010  379.0436    2
## 45  2011  323.3801    2
## 46  2012  279.3548    2
## 47  2013  249.0121    2
## 48  2014  207.4346    2
## 49  1991 1322.3153    3
## 50  1992 1322.2153    3
## 51  1993 1319.2247    3
## 52  1994 1304.5134    3
## 53  1995 1254.1579    3
## 54  1996 1184.2113    3
## 55  1997 1056.7928    3
## 56  1998  981.6638    3
## 57  1999  912.3450    3
## 58  2000  854.8325    3
## 59  2001  790.1318    3
## 60  2002  733.9095    3
## 61  2003  661.4903    3
## 62  2004  588.4459    3
## 63  2005  527.0272    3
## 64  2006  510.7087    3
## 65  2007  473.0615    3
## 66  2008  442.5116    3
## 67  2009  440.0819    3
## 68  2010  413.5764    3
## 69  2011  381.1913    3
```

```
## 70  2012  359.5963    3
## 71  2013  351.1120    3
## 72  2014  331.2602    3
## 73  1991 1307.0656    4
## 74  1992 1306.9656    4
## 75  1993 1303.9756    4
## 76  1994 1289.2843    4
## 77  1995 1239.0410    4
## 78  1996 1169.4996    4
## 79  1997 1042.8276    4
## 80  1998  968.9426    4
## 81  1999  901.0473    4
## 82  2000  845.0807    4
## 83  2001  781.9971    4
## 84  2002  727.4603    4
## 85  2003  656.7694    4
## 86  2004  585.4965    4
## 87  2005  525.8696    4
## 88  2006  511.3486    4
## 89  2007  475.5273    4
## 90  2008  446.8272    4
## 91  2009  446.2810    4
## 92  2010  421.7175    4
## 93  2011  391.3275    4
## 94  2012  371.7871    4
## 95  2013  365.4322    4
## 96  2014  347.7999    4
## 97  1991 1328.5430    5
## 98  1992 1328.4430    5
## 99  1993 1325.4518    5
## 100 1994 1310.7212    5
## 101 1995 1260.2570    5
## 102 1996 1189.9149    5
## 103 1997 1061.7601    5
## 104 1998  985.3833    5
## 105 1999  914.6150    5
## 106 2000  855.5054    5
## 107 2001  789.1120    5
## 108 2002  731.0982    5
## 109 2003  656.8155    5
## 110 2004  581.8227    5
## 111 2005  518.3867    5
## 112 2006  499.9983    5
## 113 2007  460.2350    5
## 114 2008  427.5088    5
## 115 2009  422.8342    5
## 116 2010  394.0127    5
## 117 2011  359.2215    5
## 118 2012  335.1143    5
## 119 2013  324.0020    5
## 120 2014  301.4006    5
## 121 1991 1433.7662    6
## 122 1992 1433.6662    6
## 123 1993 1430.6719    6
```

```
## 124 1994 1415.8490    6
## 125 1995 1364.8640    6
## 126 1996 1292.6264    6
## 127 1997 1160.9256    6
## 128 1998 1078.5982    6
## 129 1999 1000.9248    6
## 130 2000  934.2581    6
## 131 2001  859.9288    6
## 132 2002  793.6714    6
## 133 2003  710.9967    6
## 134 2004  627.5565    6
## 135 2005  555.7984    6
## 136 2006  529.3215    6
## 137 2007  481.4752    6
## 138 2008  440.8193    6
## 139 2009  428.3314    6
## 140 2010  391.5614    6
## 141 2011  348.8883    6
## 142 2012  317.0316    6
## 143 2013  298.1962    6
## 144 2014  267.7439    6
## 145 1991 1311.2513    7
## 146 1992 1311.1513    7
## 147 1993 1308.1605    7
## 148 1994 1293.4434    7
## 149 1995 1243.0550    7
## 150 1996 1172.9864    7
## 151 1997 1045.3340    7
## 152 1998  969.7833    7
## 153 1999  899.9534    7
## 154 2000  841.8521    7
## 155 2001  776.5012    7
## 156 2002  719.5544    7
## 157 2003  646.3440    7
## 158 2004  572.4136    7
## 159 2005  510.0047    7
## 160 2006  492.5965    7
## 161 2007  453.8125    7
## 162 2008  422.0383    7
## 163 2009  418.2962    7
## 164 2010  390.4308    7
## 165 2011  356.5856    7
## 166 2012  333.4043    7
## 167 2013  323.2152    7
## 168 2014  301.5595    7
## 169 1991 1426.7233    8
## 170 1992 1426.6233    8
## 171 1993 1423.6298    8
## 172 1994 1408.8305    8
## 173 1995 1357.9792    8
## 174 1996 1286.2334    8
## 175 1997 1155.4683    8
## 176 1998 1074.7515    8
## 177 1999  998.9902    8
```

```
## 178 2000  934.4628    8
## 179 2001  862.4254    8
## 180 2002  798.6033    8
## 181 2003  718.4626    8
## 182 2004  637.6519    8
## 183 2005  568.5811    8
## 184 2006  544.8163    8
## 185 2007  499.7122    8
## 186 2008  461.8225    8
## 187 2009  452.1295    8
## 188 2010  418.2101    8
## 189 2011  378.4306    8
## 190 2012  349.5090    8
## 191 2013  333.6663    8
## 192 2014  306.2868    8
## 193 1991 1587.2840    9
## 194 1992 1587.1840    9
## 195 1993 1584.1871    9
## 196 1994 1569.2821    9
## 197 1995 1517.8290    9
## 198 1996 1443.8700    9
## 199 1997 1308.8789    9
## 200 1998 1220.9337    9
## 201 1999 1136.5933    9
## 202 2000 1062.5106    9
## 203 2001  980.3054    9
## 204 2002  905.8251    9
## 205 2003  814.7758    9
## 206 2004  722.9615    9
## 207 2005  643.0591    9
## 208 2006  608.8144    9
## 209 2007  553.3474    9
## 210 2008  505.4238    9
## 211 2009  486.0099    9
## 212 2010  442.3720    9
## 213 2011  393.1775    9
## 214 2012  355.2670    9
## 215 2013  330.7409    9
## 216 2014  294.7867    9
## 217 1991 1529.1261   10
## 218 1992 1529.0261   10
## 219 1993 1526.0298   10
## 220 1994 1511.1427   10
## 221 1995 1459.7926   10
## 222 1996 1386.2127   10
## 223 1997 1251.9498   10
## 224 1998 1165.2474   10
## 225 1999 1082.3861   10
## 226 2000 1009.9484   10
## 227 2001  929.4863   10
## 228 2002  856.8135   10
## 229 2003  767.5877   10
## 230 2004  677.5652   10
## 231 2005  599.3619   10
```

```
## 232 2006  566.6869   10
## 233 2007  512.7284   10
## 234 2008  466.1895   10
## 235 2009  448.0384   10
## 236 2010  405.6205   10
## 237 2011  357.5174   10
## 238 2012  320.5332   10
## 239 2013  296.7964   10
## 240 2014  261.5421   10
## 241 1991 1418.0760   11
## 242 1992 1417.9760   11
## 243 1993 1414.9838   11
## 244 1994 1400.2264   11
## 245 1995 1349.6122   11
## 246 1996 1278.7356   11
## 247 1997 1149.6152   11
## 248 1998 1071.7277   11
## 249 1999  999.3093   11
## 250 2000  938.5128   11
## 251 2001  870.4684   11
## 252 2002  810.8989   11
## 253 2003  735.1989   11
## 254 2004  659.0364   11
## 255 2005  594.7731   11
## 256 2006  575.9246   11
## 257 2007  535.8174   11
## 258 2008  503.0291   11
## 259 2009  498.5513   11
## 260 2010  469.9701   11
## 261 2011  435.6765   11
## 262 2012  412.4085   11
## 263 2013  402.4051   11
## 264 2014  381.0671   11
## 265 1991 1463.6852   12
## 266 1992 1463.5852   12
## 267 1993 1460.5908   12
## 268 1994 1445.7655   12
## 269 1995 1394.7670   12
## 270 1996 1322.4827   12
## 271 1997 1190.7011   12
## 272 1998 1108.2656   12
## 273 1999 1030.4918   12
## 274 2000  963.7475   12
## 275 2001  889.3721   12
## 276 2002  823.1163   12
## 277 2003  740.4976   12
## 278 2004  657.1982   12
## 279 2005  585.6862   12
## 280 2006  559.5639   12
## 281 2007  512.1200   12
## 282 2008  471.9572   12
## 283 2009  460.0499   12
## 284 2010  423.8978   12
## 285 2011  381.9355   12
```

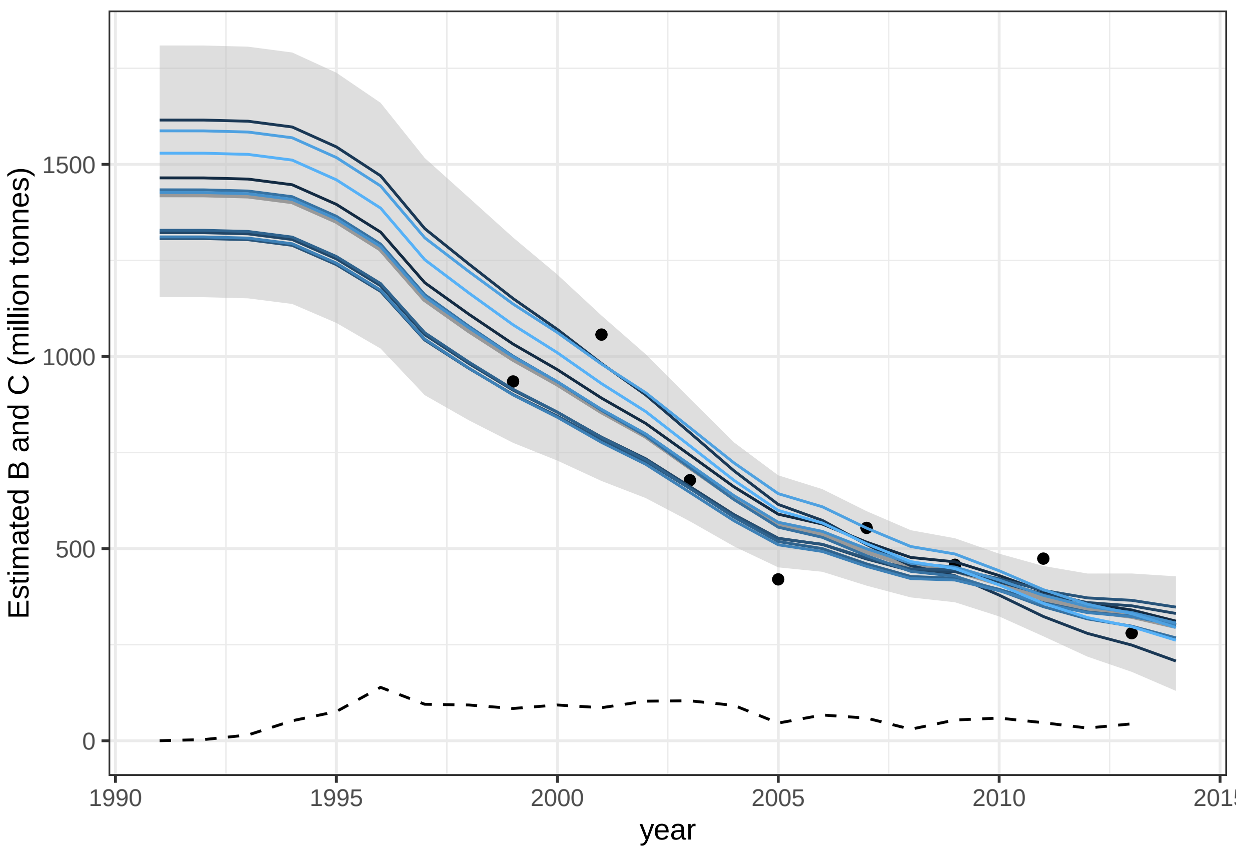
```
## 286 2012  350.9062   12
## 287 2013  333.0003   12
## 288 2014  303.5547   12
## 289 1991 1318.6069   13
## 290 1992 1318.5069   13
## 291 1993 1315.5157   13
## 292 1994 1300.7876   13
## 293 1995 1250.3370   13
## 294 1996 1180.0428   13
## 295 1997 1051.9726   13
## 296 1998  975.7219   13
## 297 1999  905.0853   13
## 298 2000  846.1033   13
## 299 2001  779.8268   13
## 300 2002  721.9117   13
## 301 2003  647.7056   13
## 302 2004  572.7526   13
## 303 2005  509.3078   13
## 304 2006  490.8598   13
## 305 2007  451.0191   13
## 306 2008  418.1752   13
## 307 2009  413.3446   13
## 308 2010  384.3567   13
## 309 2011  349.3595   13
## 310 2012  324.9940   13
## 311 2013  313.5793   13
## 312 2014  290.6456   13
## 313 1991 1391.4418   14
## 314 1992 1391.3418   14
## 315 1993 1388.3501   14
## 316 1994 1373.6049   14
## 317 1995 1323.0593   14
## 318 1996 1252.4301   14
## 319 1997 1123.7652   14
## 320 1998 1046.6217   14
## 321 1999  975.0453   14
## 322 2000  915.1468   14
## 323 2001  848.0216   14
## 324 2002  789.3776   14
## 325 2003  714.5899   14
## 326 2004  639.3018   14
## 327 2005  575.8458   14
## 328 2006  557.7258   14
## 329 2007  518.3284   14
## 330 2008  486.1933   14
## 331 2009  482.3202   14
## 332 2010  454.3499   14
## 333 2011  420.6244   14
##  [ reached 'max' / getOption("max.print") -- omitted 11667 rows ]
```

We can now plot the estimated biomass time series

```
Fig1 <- ggplot(data=biomass.iter,aes(x=year,y=biomass))
Fig1 +
     stat_summary(fun.data = "median_hilow",
```

```
                     fun.args = list(conf.int=1),
                     geom = "ribbon" ,
                     alpha=0.5,
                     fun.min = function(x)0,
                     fill = "gray")  +
    stat_summary(fun.data = "median_hilow",
                     geom = "smooth" ,
                     color = gray(0.6))  +
    geom_line(aes(y=harvest,x=year), data = data.frame(harvest = harvest,
                year = data.years),lty=2) +
    geom_point(aes(y=index, x=year), data = data.frame(index=index,
                year = data.years)) +
    geom_line(aes(y=biomass,x=year,group=iter,col=iter),data = subset(biomass.iter,iter%in%1:10)) +
     ylab("Estimated B and C (million tonnes)") +
  theme_bw() +
  guides(col=FALSE)
```

```
## Warning: Removed 15 rows containing missing values (geom_point).
```




The shaded area indicates the range of the biomass time series, with the dark line the median. The lighter lines indicate indiviudal biomass trajectories.

**Applying the Management Strategy** We have now conditioned our operating model. We will conduct the MSE loop over a 20 year projection period, with the catches set each year by repeated estimation of the current biomass and application of a harvest control rule.

Define the years for the projection:

```r
proj.years <- 2014:2034
```

**Data generation** We write a function to generate the observations (new biomass index values) from the operating model.

```r
observe <- function(biomass, sigma) {
  biomass * exp(rnorm(1, -0.5*sigma^2, sigma))
}
```

This function takes the true biomass from the operating model, and generates the data by adding (lognormally distributed) observation error.

**Harvest Control Rule** We first demonstrate the MSE using a fixed target exploitation rate - the control rule calculates the catch for next year based on a fixed percentage (10%) of the most recent biomass estimate.

```r
control.pars <- list()
control.pars$Htarg <- 0.1
control <- function(estimated.biomass, control.pars) {
  control.pars$Htarg
}
```

We assume perfect implementation of the strategy - in that the realized catch is the same as the TAC.

```r
implement <- function(TAC,...) {
  TAC
}
```

Evaluation function that projects the operating model forward & implements the mgmt procudeure at each time step.
We will first step through this for one iteration to view how things work.

```r
#evaluate <- function(pars.iter, biomass.iter,
#                      control.pars, data.years, proj.years,
#                      iterations, ...) {
    # function arguments:
    # pars.iter & biomass.iter, the parameters & historical biomass trajectories of the operating model
    # control.pars, the specifications of the harvest control rule

    # set up some indexing values
    iyr <- length(data.years)+1
    pyr <- length(proj.years)
    yrs <- c(data.years, proj.years, max(proj.years)+1)

    # set up a data frame to store the results
    res <- data.frame()

    # loop over the iterations of the MSE, each iteration conducts a 20 year projection with annual gen
    # observations and appliations of the control rule.
    #for(i in 1:iterations) {
    i = 1

      #extract the parameters for this iteration
      K.i <- exp(pars.iter[i,1])
      r.i <- pars.iter[i,2]
      sig.i <- pars.iter[i,3]
```

```r
      #set up vectors for time series of interest.
      biomass.i <- c(subset(biomass.iter, iter==i)$biomass, numeric(pyr))
      index.i <- c(index,numeric(pyr))
      catch.i <- c(harvest, numeric(pyr))
      TAC.i <- c(rep(NA,iyr-1),numeric(pyr))

      # loop over the projection period.
      #for (y in iyr:(iyr+pyr-1)) {
      y <- iyr

        #generate the data for the most recent year
        index.i[y] <- observe(biomass.i[y] , sig.i)
        #calculate the TAC based on the harvest control rule
        # note that the control rule ONLY sees the index data, not the operating model biomass.
        TAC.i[y]  <- control(index.i[y], control.pars) * index.i[y]
        #find the realized catch after implementation error
        catch.i[y] <- implement(TAC.i[y])

        # update the true biomass of the operating model based on the output of the HCR
        biomass.i[y+1] <- schaefer(biomass.i[y],catch.i[y],K.i,r.i)


      # loop over the remaining years of the projection period.
      for (y in (iyr+1):(iyr+pyr-1)) {
        #generate the data for the most recent year
        index.i[y] <- observe(biomass.i[y] , sig.i)
        #calculate the TAC based on the harvest control rule
        # note that the control rule ONLY sees the index data, not the operating model biomass.
        TAC.i [y]  <- control(index.i[y], control.pars) * index.i[y]
        #find the realized catch after implementation error
        catch.i[y] <- implement(TAC.i[y])

        # update the true biomass of the operating model based on the output of the HCR
        biomass.i[y+1] <- schaefer(biomass.i[y],catch.i[y],K.i,r.i)

      #end projection year loop for iteration i
      }

      #store the results for this iteration
      res <- rbind(res, data.frame(year = yrs[-length(yrs)],
                                   value = index.i, type = "index", iter = i),
                   data.frame(year = yrs[-length(yrs)],
                              value = catch.i, type = "catch", iter=i),
                   data.frame(year = yrs, value = biomass.i,
                              type= "biomass", iter=i),
                   data.frame(year = yrs[-length(yrs)],
                              value = TAC.i, type = "tac", iter=i))
    #end loop over iterations
    #}
    #return(res)
#end function evaluate()
#}
```

Reloading the full function with lines uncommented (code hidden from html to save scrolling time), means we can then run this.

Project with fixed 10% exploitation rate of estimated biomass for all iterations & 20 yrs

```
project.fixed <- evaluate(pars.iter, biomass.iter, control.pars, data.years,
                          proj.years, niter)
tail(project.fixed)
```

```
##       year    value type iter
## 88495 2029 15.89713  tac  500
## 88496 2030 15.40564  tac  500
## 88497 2031 19.54445  tac  500
## 88498 2032 18.44814  tac  500
## 88499 2033 16.56991  tac  500
## 88500 2034 19.61119  tac  500
```

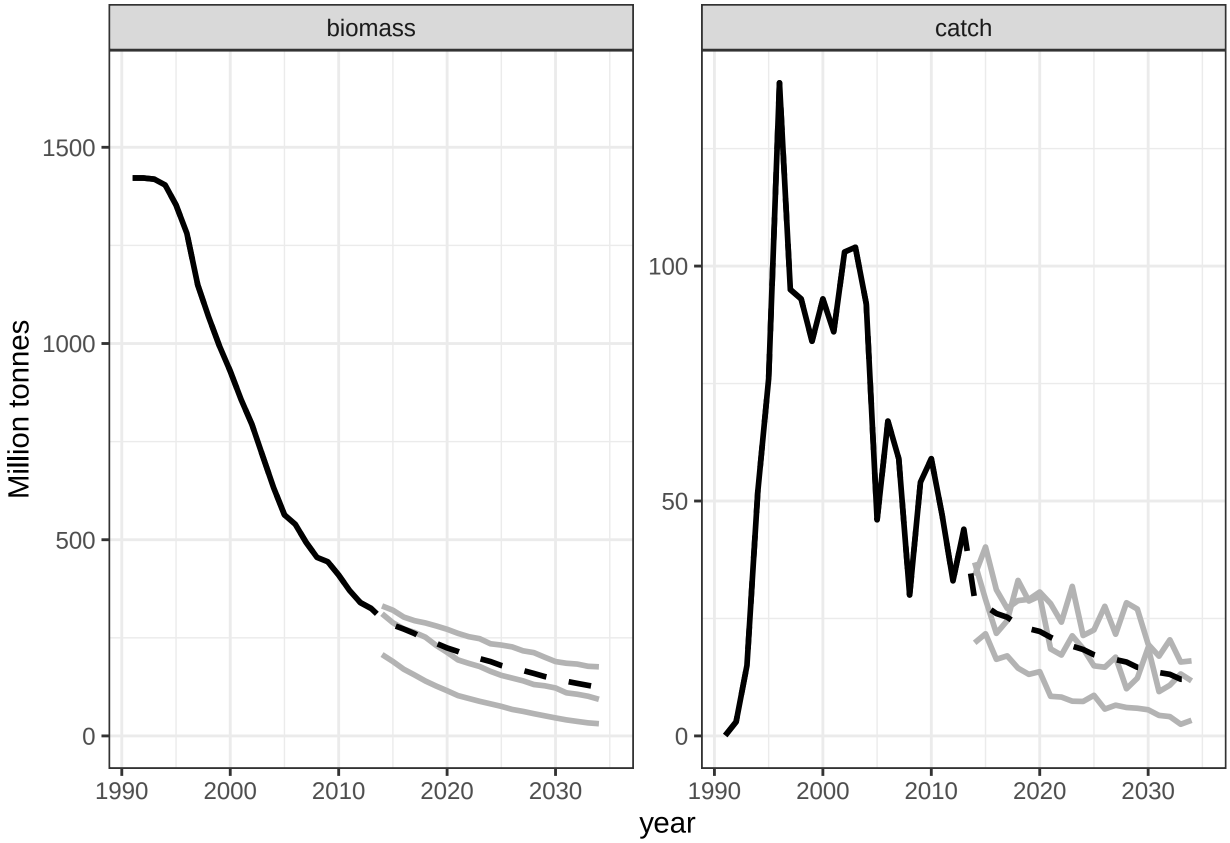
We can view the trajectories of catch and operating model biomass from the output.
We will do this again so write a function to repeat the task easily

```
projection.plot <- function(project.results) {
  Fig2 <- ggplot(data = subset(project.results, type != "index"),
               aes(x = year, y = value))
  Fig2 + geom_line(aes(y=value,x=year),data = subset(project.results, type != "index" & iter==1 & year %
           geom_line(aes(y=value,x=year),data = subset(project.results, type != "index" & iter==2 & year
           geom_line(aes(y=value,x=year),data = subset(project.results, type != "index" & iter==3 & year
stat_summary(fun.data = "median_hilow", geom = "smooth", col="black",
                    fill = gray(0.5), lty = 2, aes=0.1) +
       stat_summary(fun = median, fun.min = function(x)0, geom="line",
                    data = subset(project.results, type != "index" &  year %in% data.years), lwd=1) +fa

}
```

Plot the projection:

```
projection.plot(dplyr::filter(project.fixed, type != "tac"))
```

```
## Warning: Ignoring unknown parameters: aes
```




**Management using alternative harvest control rules** Define a HCR that converts estimated biomass into a harvest rate using a functional form determined by values in 'control.pars'.

```
control <- function(estimated.biomass, control.pars) {
  H1 <- control.pars$H1
  H2 <- control.pars$H2
  Bmax <- control.pars$Bmax
  B2 <- control.pars$B2
  B1 <- control.pars$B1

  harv <- ifelse(estimated.biomass >= B1, H1,
               ifelse(estimated.biomass < B2, H2,
                      (H1-H2)/(B1-B2)*(estimated.biomass - B2) + H2))

  return(harv)

#end function control
}
```

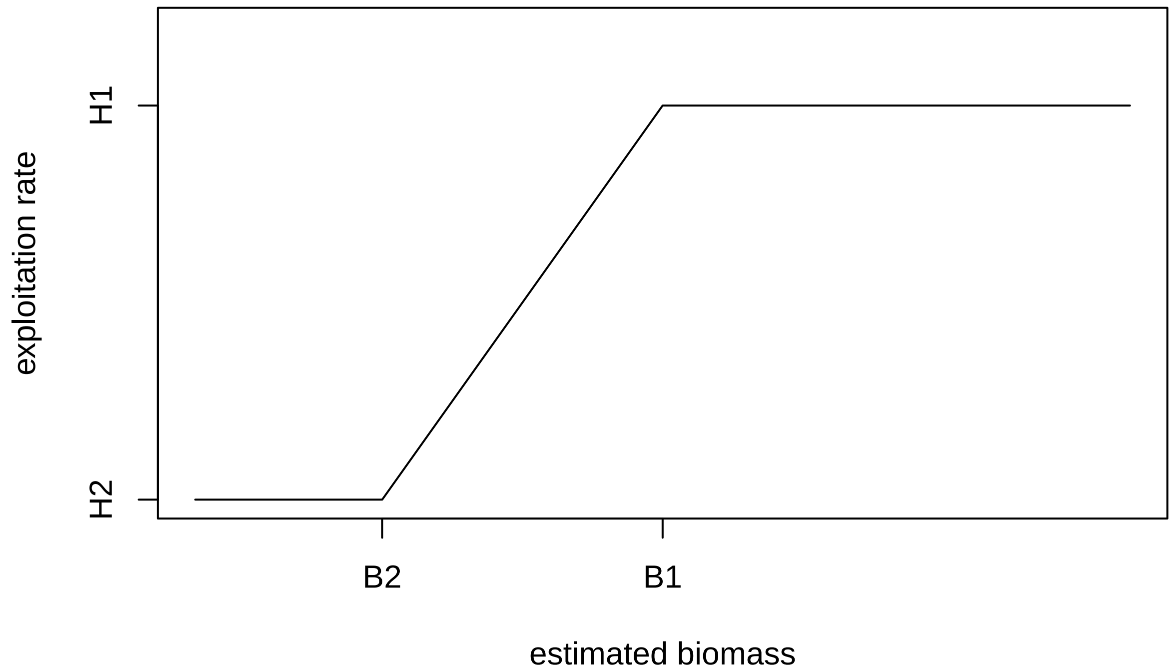
Define control parameters for HCR using reference points
We (arbitrarily) set the threshold and limit biomass reference points as 50% & 20% of the maximum observed survey index value during the historical period.
The target exploitation rate is set at 5%.

```
control.pars <- list()
control.pars$H1 <- 0.05
control.pars$H2 <- 0
control.pars$Bmax <- max(index, na.rm =TRUE)
```

```
control.pars$B2 <- 0.2*control.pars$Bmax
control.pars$B1 <- 0.5*control.pars$Bmax
```

Plot the HCR shape:

```
plot(c(0,control.pars$B2,control.pars$B1,control.pars$Bmax),
     c(control.pars$H2,control.pars$H2,control.pars$H1,control.pars$H1),
     type='l',axes=F,xlab="estimated biomass",ylab="exploitation rate",
     ylim=c(0,1.2*control.pars$H1))
axis(1,at=c(control.pars$B2,control.pars$B1),labels=c("B2","B1"))
axis(2,at=c(control.pars$H2,control.pars$H1),labels=c("H2","H1"))
box()
```



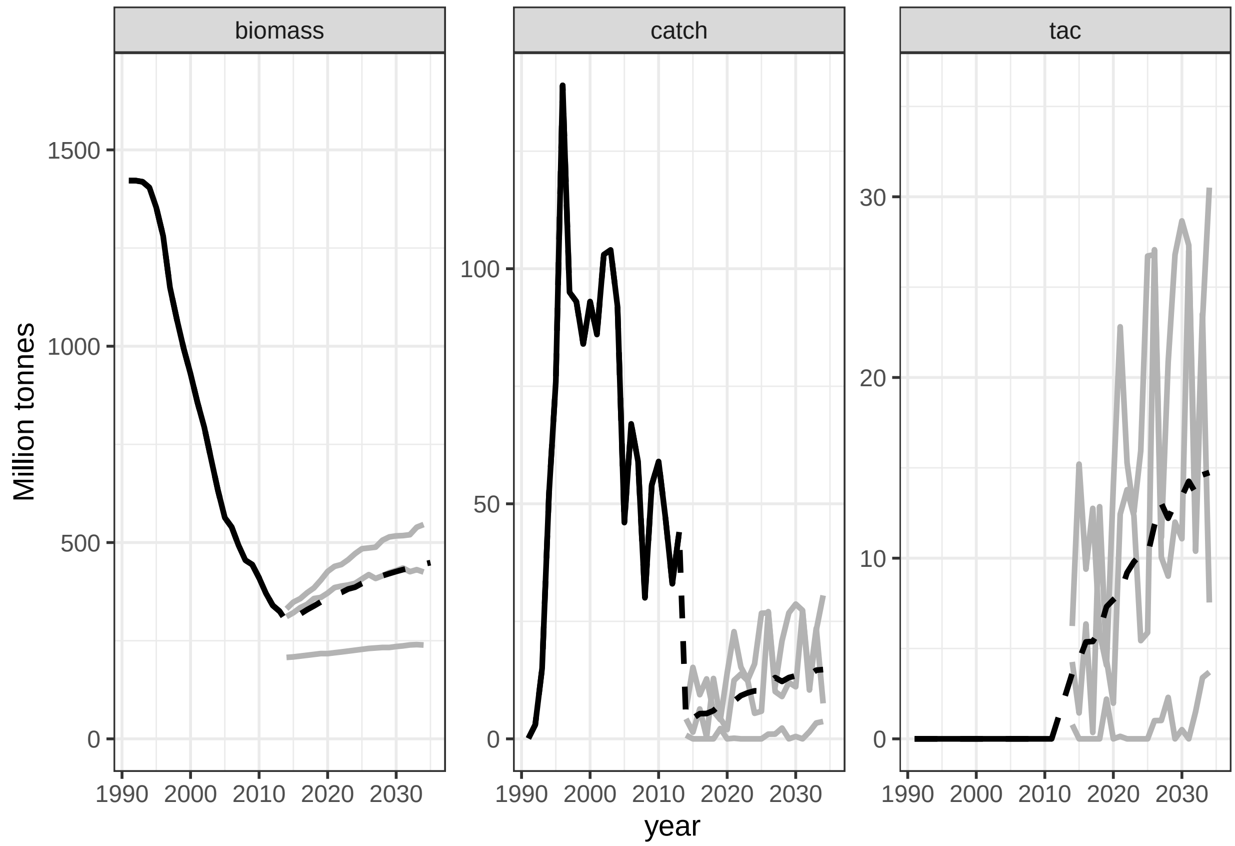

Conduct the evaluation by projecting system forward in time

```
project.hcr <- evaluate(pars.iter, biomass.iter, control.pars,
                        data.years, proj.years, niter)
```

Plot the trajectories:

```
projection.plot(project.hcr)
```

```
## Warning: Ignoring unknown parameters: aes

## Warning: Removed 1000 rows containing non-finite values (stat_summary).

## Warning: Removed 1000 rows containing non-finite values (stat_summary).
```




Now let's add potential for overshooting the TAC

```
implement <- function(TAC, overshoot, ...) {
  TAC * (1 + overshoot)
}
```

**Comparing different HCRs & accounting for possible TAC overshoot** Set the HCR parameters

```
control.pars <- list()
control.pars$H1 <- 0.05
control.pars$H2 <- 0
control.pars$Bmax <- max(index, na.rm =TRUE)
control.pars$B2 <- 0.2*control.pars$Bmax
control.pars$B1 <- 0.5*control.pars$Bmax
```

Conduct the base scenario (no TAC overshoot)

```
proj.hcr1.noerror <- evaluate(pars.iter, biomass.iter,
                              control.pars, data.years,
                              proj.years, niter,
                              overshoot = 0)
```

Now run the HCR with 20% overshoot in TAC

```
proj.hcr1.error <- evaluate(pars.iter, biomass.iter,
                                control.pars, data.years,
                                proj.years, niter,
                                overshoot = 0.2)
```

We will further do two more HCRs where we increase the target harvest rate:

```r
control.pars$H1 <- 0.15
```

Run both scenarios with this new target harvest rate

```r
proj.hcr2.noerror <- evaluate(pars.iter, biomass.iter,
                              control.pars, data.years,
                              proj.years, niter,
                              overshoot = 0)
proj.hcr2.error <- evaluate(pars.iter, biomass.iter,
                            control.pars, data.years,
                            proj.years, niter,
                            overshoot = 0.2)
```

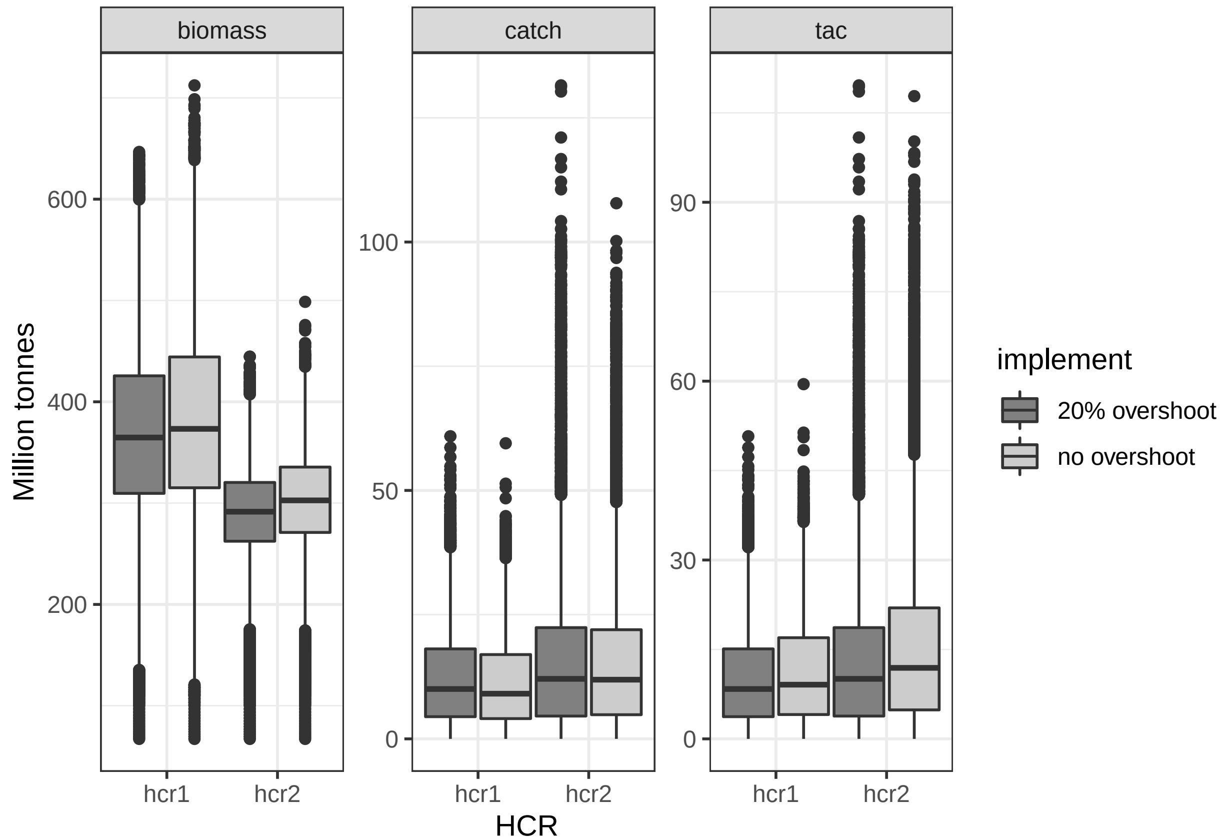
**Diagnostics** We have run the evaluations for 4 HCRs. We can now compare these. Create an object containing all the results:

```r
MSE <- rbind(cbind(proj.hcr1.noerror, HCR="hcr1",
                   implement = "no overshoot"),
             cbind(proj.hcr1.error, HCR="hcr1",
                   implement = "20% overshoot"),
             cbind(proj.hcr2.noerror, HCR="hcr2",
                   implement = "no overshoot"),
             cbind(proj.hcr2.error, HCR="hcr2",
                   implement = "20% overshoot"))
head(MSE)
```

```
##   year value  type iter  HCR    implement
## 1 1991    NA index    1 hcr1 no overshoot
## 2 1992    NA index    1 hcr1 no overshoot
## 3 1993    NA index    1 hcr1 no overshoot
## 4 1994    NA index    1 hcr1 no overshoot
## 5 1995    NA index    1 hcr1 no overshoot
## 6 1996    NA index    1 hcr1 no overshoot
```

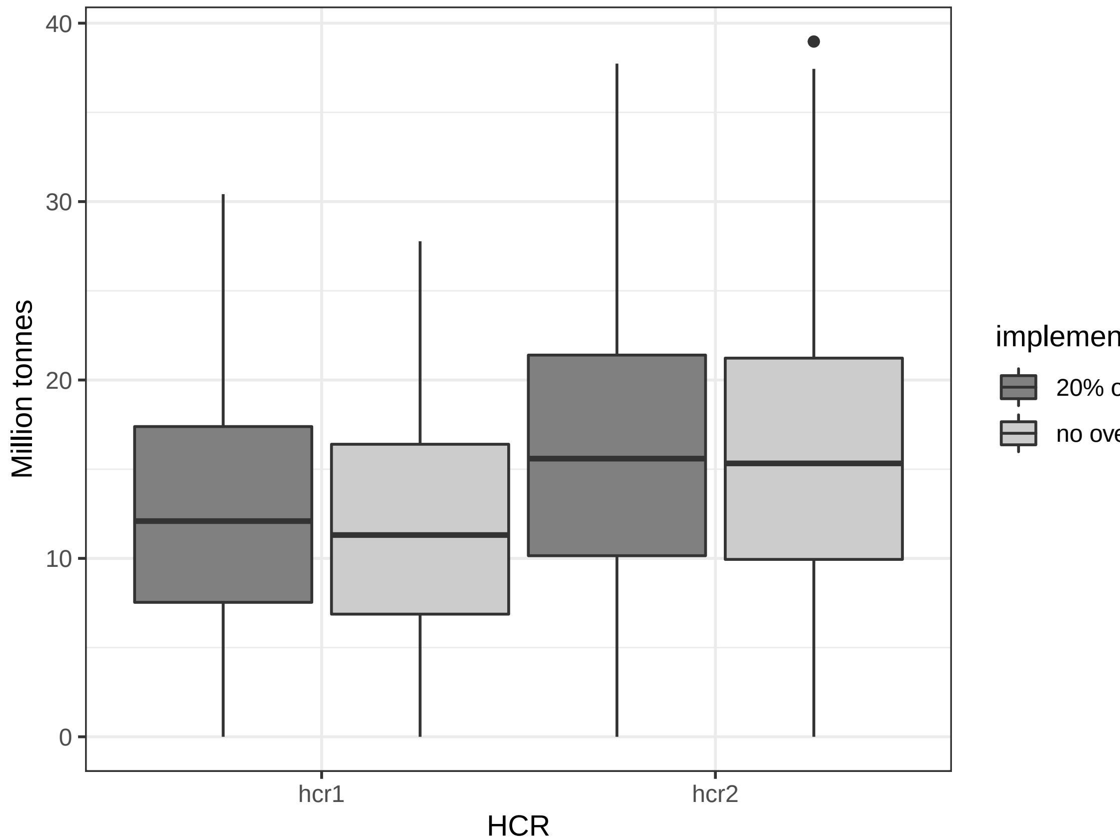
Summarize biomass & catch for all 4 options:

```r
Fig5 <- ggplot(data=subset(MSE, type !="index" &
                          year %in% proj.years),
               aes(x=HCR, y=value, ymin=0))
Fig5 + geom_boxplot(aes(fill=implement), width = 1) + facet_wrap(~type, scale="free_y") + ylab("Million
```



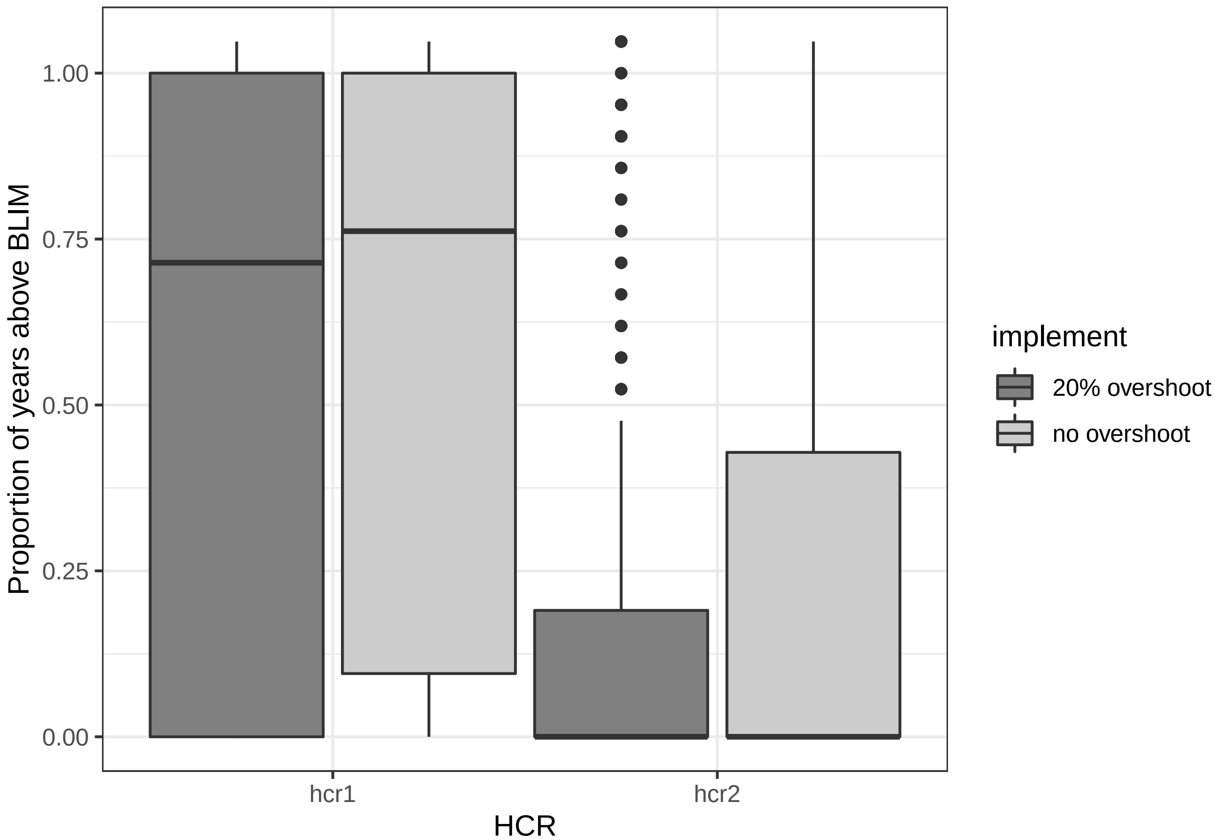

We immediately see a yield-biomass tradeoff - HCR2 gives more catch but leads to lower biomass. There is not much change when the catch is 20% higher than the TAC.

```
#Yield based metrics  (e.g. average annual catch)
#Stock Biomass metrics (e.g. distribution for B/BMSY, P(B>BLIM), etc.)
#Inter-annual stability of catch advice (e.g. how often the control rule closes the fishery)

aac2 <- with(MSE[MSE$year>max(data.years) & MSE$type=="catch",],
            aggregate(value,by=list(iter=iter,HCR=HCR,implement=implement),FUN=mean,na.rm=TRUE))


Fig6 <- ggplot(data=subset(aac2),
               aes(x=HCR, y=x, ymin=0))
Fig6 + geom_boxplot(aes(fill=implement), width = 1)  + ylab("Million tonnes") + scale_fill_grey(start=0
```

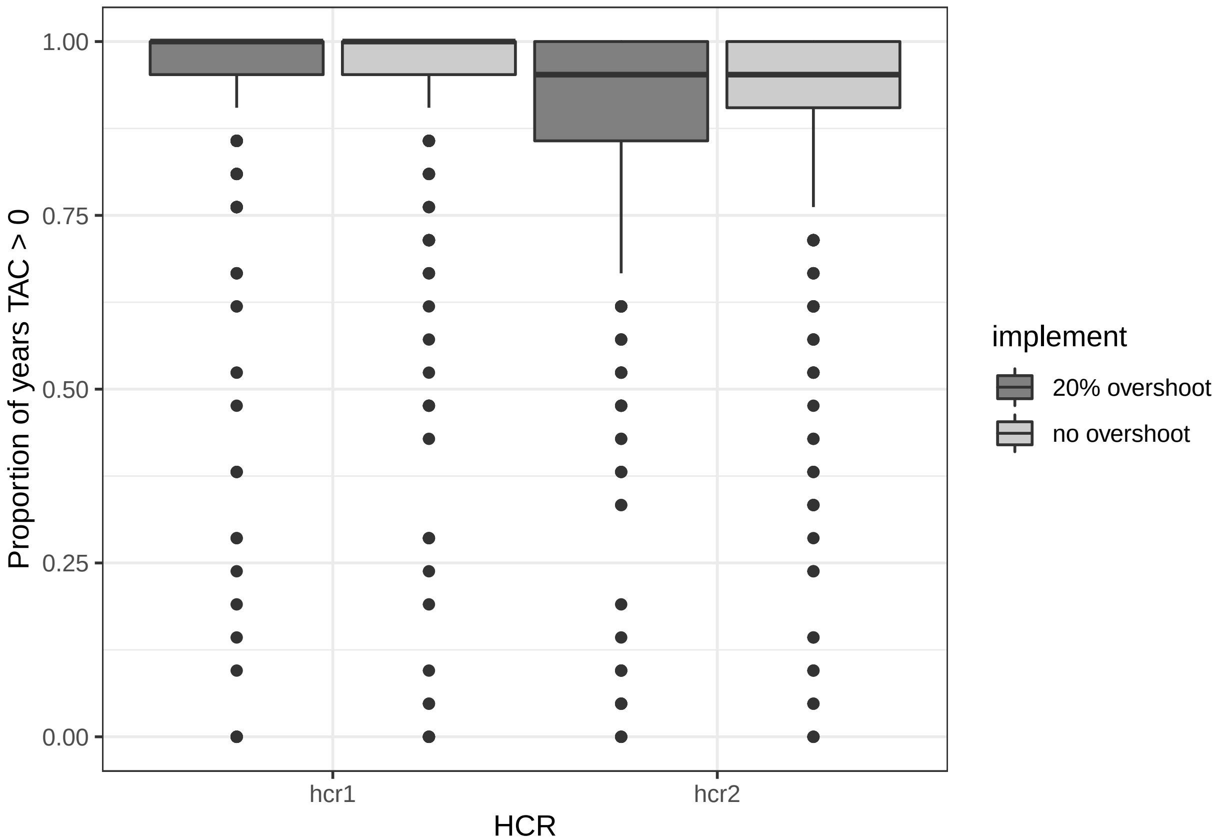



**Performance statistics**

```
# years B > BLIM
# BLIM = 0.25*K  (we specify BLIM for our performance as half BMSY)
blim <- 0.25*exp(pars.iter[,1])

num.above <- function(vec,threshold) {
 length(vec[vec>threshold])/length(vec)
}
MSE$blim <- blim[MSE$iter]
MSE$above.blim <- ifelse(MSE$value>MSE$blim,1,0)

above.blim <- with(MSE[MSE$year>max(data.years) & MSE$type=="biomass",],
            aggregate(above.blim,by=list(iter=iter,HCR=HCR,implement=implement),FUN=sum,na.rm=TRUE))
above.blim$x <- above.blim$x/length(proj.years)

Fig7 <- ggplot(data=subset(above.blim),
               aes(x=HCR, y=x, ymin=0))
Fig7 + geom_boxplot(aes(fill=implement), width = 1)  + ylab("Proportion of years above BLIM") + scale_fi
```




```
# num years fishery is open
not.closed <- with(MSE[MSE$year>max(data.years) & MSE$type=="catch",],
             aggregate(value,by=list(iter=iter,HCR=HCR,implement=implement),FUN=num.above,threshold=0))

Fig8 <- ggplot(data=subset(not.closed),
               aes(x=HCR, y=x, ymin=0))
Fig8 + geom_boxplot(aes(fill=implement), width = 1)  + ylab("Proportion of years TAC > 0") + scale_fill_
```




**Next Steps**

Your turn to add features!

Suggestions:
1. Produce a trade-off plot (hint: perhaps think about some alternative performance statistics that integrate across iterations)
2. Add a model-based control rule by performing a stock assessment (e.g. production model) each year in the projection period. Then use the catch associated with the estimated FMSY as the TAC. Be careful not to give the assessment model the true parameter values from the operating model.
3. Implement the HCR every 3 yrs rather than every 1.
4. Add a more complicated implementation function (say based on price?)
5. Add environmental variability (process error) into the population dynamics