CINAR MSE Workshop: First MSE

Gavin Fay

January 2022

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 (e.g. from a survey).
* 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 see the code on the landing page of the website to 'install.packages()')

```
library(tidyverse)
library(ggdist)
library(Hmisc)
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 simulations.

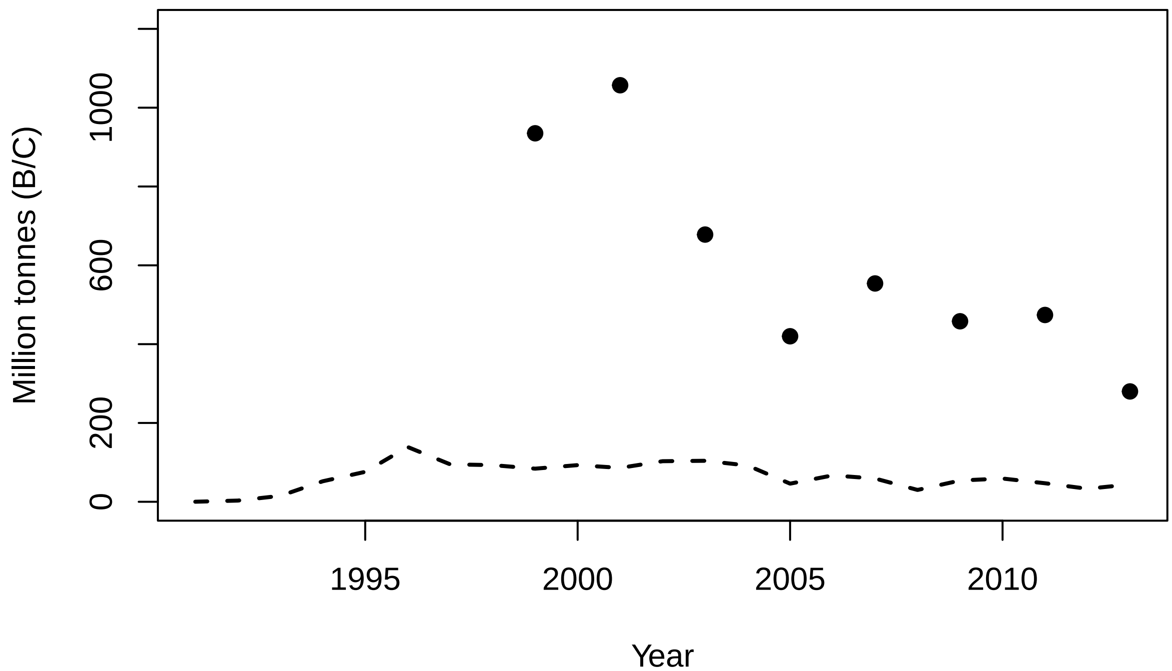
#####Specify input data and associated years

```
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 <- exp(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 <- exp(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)

```
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), log(r), log(sigma)

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

Fit the logistic model to data:

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

```
## $pars
## [1]  7.258575 -2.721061 -1.769252
##
## $biomass
##  [1] 1420.2310 1420.1310 1417.1376 1402.3407
##  [5] 1351.5032 1279.8069 1149.1339 1068.5681
##  [9]  992.9793  928.6366  856.7886  793.1563
## [13]  713.2013  632.5654  563.6513  540.0219
## [17]  495.0459  457.2673  447.6696  413.8427
## [21]  374.1402  345.2745  329.4716  302.1228
##
## $convergence
## [1] 0
##
## $nll
## [1] -2.802687
##
## $vcov
##               [,1]          [,2]          [,3]
## [1,]  4.859291e-03 -2.777817e-02 -2.724771e-06
## [2,] -2.777817e-02  1.690107e-01  1.448294e-05
## [3,] -2.724771e-06  1.448294e-05  6.250267e-02
```

Extract the maximum likelihood and parameter estimates

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

```
##  [1] 1420.2310 1420.1310 1417.1376 1402.3407
##  [5] 1351.5032 1279.8069 1149.1339 1068.5681
##  [9]  992.9793  928.6366  856.7886  793.1563
## [13]  713.2013  632.5654  563.6513  540.0219
## [17]  495.0459  457.2673  447.6696  413.8427
## [21]  374.1402  345.2745  329.4716  302.1228
```

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

```
## [1] 1.420231e+03 6.580487e-02 1.704604e-01
```

To obtain a set of 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.

```r
set.seed(8675309)
#define the number of iterations for the MSE
niter <- 200
#set up a storage matrix for our alternative parameter sets
pars.iter <- matrix(NA,nrow = niter, ncol=3)
colnames(pars.iter) <- c("log_K","log_r","log_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 1319.4401    1
## 2    1992 1319.3401    1
## 3    1993 1316.3495    1
## 4    1994 1301.6394    1
## 5    1995 1251.2907    1
## 6    1996 1181.3680    1
## 7    1997 1053.9929    1
## 8    1998  978.9322    1
## 9    1999  909.6883    1
## 10   2000  852.2532    1
## 11   2001  787.6294    1
## 12   2002  731.4814    1
## 13   2003  659.1325    1
## 14   2004  586.1505    1
## 15   2005  524.7828    1
## 16   2006  508.5032    1
## 17   2007  470.8916    1
## 18   2008  440.3685    1
## 19   2009  437.9575    1
## 20   2010  411.4706    1
## 21   2011  379.0966    1
```

```
## 22  2012  357.5023    1
## 23  2013  349.0110    1
## 24  2014  329.1488    1
## 25  1991 1402.5315    2
## 26  1992 1402.4315    2
## 27  1993 1399.4404    2
## 28  1994 1384.7151    2
## 29  1995 1334.2817    2
## 30  1996 1264.0644    2
## 31  1997 1136.1791    2
## 32  1998 1060.3961    2
## 33  1999  990.4343    2
## 34  2000  932.3526    2
## 35  2001  867.1898    2
## 36  2002  810.6698    2
## 37  2003  738.1380    2
## 38  2004  665.2798    2
## 39  2005  604.4258    2
## 40  2006  589.0586    2
## 41  2007  552.4873    2
## 42  2008  523.3099    2
## 43  2009  522.5271    2
## 44  2010  497.7267    2
## 45  2011  467.3242    2
## 46  2012  448.0770    2
## 47  2013  442.2346    2
## 48  2014  425.2020    2
## 49  1991 1361.3248    3
## 50  1992 1361.2248    3
## 51  1993 1358.2334    3
## 52  1994 1343.4997    3
## 53  1995 1293.0184    3
## 54  1996 1222.6195    3
## 55  1997 1094.3740    3
## 56  1998 1017.9009    3
## 57  1999  947.0697    3
## 58  2000  887.9500    3
## 59  2001  821.6063    3
## 60  2002  763.7277    3
## 61  2003  689.6713    3
## 62  2004  615.0474    3
## 63  2005  552.1556    3
## 64  2006  534.4895    3
## 65  2007  495.5157    3
## 66  2008  463.7230    3
## 67  2009  460.1196    3
## 68  2010  432.4162    3
## 69  2011  398.8893    3
## 70  2012  376.2354    3
## 71  2013  366.7394    3
## 72  2014  345.8710    3
## 73  1991 1607.8739    4
## 74  1992 1607.7739    4
## 75  1993 1604.7773    4
```

```
## 76  1994 1589.8796    4
## 77  1995 1538.4685    4
## 78  1996 1464.6668    4
## 79  1997 1329.9849    4
## 80  1998 1242.5935    4
## 81  1999 1158.9378    4
## 82  2000 1085.6489    4
## 83  2001 1004.3207    4
## 84  2002  930.7996    4
## 85  2003  840.7738    4
## 86  2004  750.0515    4
## 87  2005  671.2973    4
## 88  2006  638.2407    4
## 89  2007  583.9810    4
## 90  2008  537.2906    4
## 91  2009  519.1324    4
## 92  2010  476.7681    4
## 93  2011  428.8701    4
## 94  2012  392.2796    4
## 95  2013  369.0965    4
## 96  2014  334.5094    4
## 97  1991 1487.6994    5
## 98  1992 1487.5994    5
## 99  1993 1484.6044    5
## 100 1994 1469.7584    5
## 101 1995 1418.6420    5
## 102 1996 1345.9248    5
## 103 1997 1213.3190    5
## 104 1998 1129.4746    5
## 105 1999 1050.0327    5
## 106 2000  981.4324    5
## 107 2001  905.0820    5
## 108 2002  836.7521    5
## 109 2003  752.0040    5
## 110 2004  666.5429    5
## 111 2005  592.8837    5
## 112 2006  564.6611    5
## 113 2007  515.1263    5
## 114 2008  472.9144    5
## 115 2009  458.9958    5
## 116 2010  420.8179    5
## 117 2011  376.8623    5
## 118 2012  343.8904    5
## 119 2013  324.0711    5
## 120 2014  292.7074    5
## 121 1991 1542.3313    6
## 122 1992 1542.2313    6
## 123 1993 1539.2361    6
## 124 1994 1524.3858    6
## 125 1995 1473.2455    6
## 126 1996 1400.4439    6
## 127 1997 1267.6881    6
## 128 1998 1183.6290    6
## 129 1999 1103.9709    6
```

```
## 130 2000 1035.1784    6
## 131 2001  958.6761    6
## 132 2002  890.2593    6
## 133 2003  805.5017    6
## 134 2004  720.1528    6
## 135 2005  646.7591    6
## 136 2006  618.9608    6
## 137 2007  569.9209    6
## 138 2008  528.3363    6
## 139 2009  515.1714    6
## 140 2010  477.8002    6
## 141 2011  434.7838    6
## 142 2012  402.9161    6
## 143 2013  384.3427    6
## 144 2014  354.3287    6
## 145 1991 1491.4066    7
## 146 1992 1491.3066    7
## 147 1993 1488.3113    7
## 148 1994 1473.4579    7
## 149 1995 1422.2998    7
## 150 1996 1349.4286    7
## 151 1997 1216.5272    7
## 152 1998 1132.1718    7
## 153 1999 1052.1183    7
## 154 2000  982.8305    7
## 155 2001  905.7415    7
## 156 2002  836.6271    7
## 157 2003  751.0648    7
## 158 2004  664.7648    7
## 159 2005  590.2571    7
## 160 2006  561.1888    7
## 161 2007  510.8060    7
## 162 2008  467.7505    7
## 163 2009  452.9921    7
## 164 2010  413.9657    7
## 165 2011  369.1634    7
## 166 2012  335.3511    7
## 167 2013  314.6919    7
## 168 2014  282.4793    7
## 169 1991 1459.2658    8
## 170 1992 1459.1658    8
## 171 1993 1456.1714    8
## 172 1994 1441.3433    8
## 173 1995 1390.3290    8
## 174 1996 1317.9864    8
## 175 1997 1186.0917    8
## 176 1998 1103.4555    8
## 177 1999 1025.4375    8
## 178 2000  958.4130    8
## 179 2001  883.7301    8
## 180 2002  817.1384    8
## 181 2003  734.1606    8
## 182 2004  650.4742    8
## 183 2005  578.5495    8
```

```
## 184 2006  551.9929    8
## 185 2007  504.1032    8
## 186 2008  463.4767    8
## 187 2009  451.0879    8
## 188 2010  414.4417    8
## 189 2011  371.9652    8
## 190 2012  340.3981    8
## 191 2013  321.9313    8
## 192 2014  291.9029    8
## 193 1991 1343.9982    9
## 194 1992 1343.8982    9
## 195 1993 1340.9071    9
## 196 1994 1326.1821    9
## 197 1995 1275.7500    9
## 198 1996 1205.5278    9
## 199 1997 1077.6052    9
## 200 1998 1001.6549    9
## 201 1999  931.4103    9
## 202 2000  872.9115    9
## 203 2001  807.1998    9
## 204 2002  749.9537    9
## 205 2003  676.5173    9
## 206 2004  602.4828    9
## 207 2005  540.1291    9
## 208 2006  522.9420    9
## 209 2007  484.4345    9
## 210 2008  453.0668    9
## 211 2009  449.8530    9
## 212 2010  422.5451    9
## 213 2011  389.3827    9
## 214 2012  367.0493    9
## 215 2013  357.8451    9
## 216 2014  337.2628    9
## 217 1991 1458.4753   10
## 218 1992 1458.3753   10
## 219 1993 1455.3812   10
## 220 1994 1440.5633   10
## 221 1995 1389.6065   10
## 222 1996 1317.4756   10
## 223 1997 1185.9859   10
## 224 1998 1104.0514   10
## 225 1999 1026.8714   10
## 226 2000  960.7898   10
## 227 2001  887.1219   10
## 228 2002  821.6139   10
## 229 2003  739.7686   10
## 230 2004  657.2640   10
## 231 2005  586.5543   10
## 232 2006  561.2310   10
## 233 2007  514.5896   10
## 234 2008  475.2267   10
## 235 2009  464.1179   10
## 236 2010  428.7760   10
## 237 2011  387.6259   10
```

```
## 238 2012  357.4077   10
## 239 2013  340.3178   10
## 240 2014  311.7023   10
## 241 1991 1332.7347   11
## 242 1992 1332.6347   11
## 243 1993 1329.6449   11
## 244 1994 1314.9613   11
## 245 1995 1264.7609   11
## 246 1996 1195.3807   11
## 247 1997 1069.0236   11
## 248 1998  995.7313   11
## 249 1999  928.5703   11
## 250 2000  873.4686   11
## 251 2001  811.3580   11
## 252 2002  757.9314   11
## 253 2003  688.4780   11
## 254 2004  618.6324   11
## 255 2005  560.6491   11
## 256 2006  547.9806   11
## 257 2007  514.0936   11
## 258 2008  487.5002   11
## 259 2009  489.2288   11
## 260 2010  467.0048   11
## 261 2011  439.1365   11
## 262 2012  422.3527   11
## 263 2013  418.9599   11
## 264 2014  404.4388   11
## 265 1991 1501.3864   12
## 266 1992 1501.2864   12
## 267 1993 1498.2913   12
## 268 1994 1483.4438   12
## 269 1995 1432.3189   12
## 270 1996 1359.5715   12
## 271 1997 1226.9108   12
## 272 1998 1142.9830   12
## 273 1999 1063.4518   12
## 274 2000  994.7642   12
## 275 2001  918.3342   12
## 276 2002  849.9388   12
## 277 2003  765.1435   12
## 278 2004  679.6652   12
## 279 2005  606.0279   12
## 280 2006  577.8684   12
## 281 2007  528.4150   12
## 282 2008  486.3191   12
## 283 2009  472.5496   12
## 284 2010  434.5346   12
## 285 2011  390.7767   12
## 286 2012  358.0461   12
## 287 2013  338.5057   12
## 288 2014  307.4482   12
## 289 1991 1451.5919   13
## 290 1992 1451.4919   13
## 291 1993 1448.4977   13
```

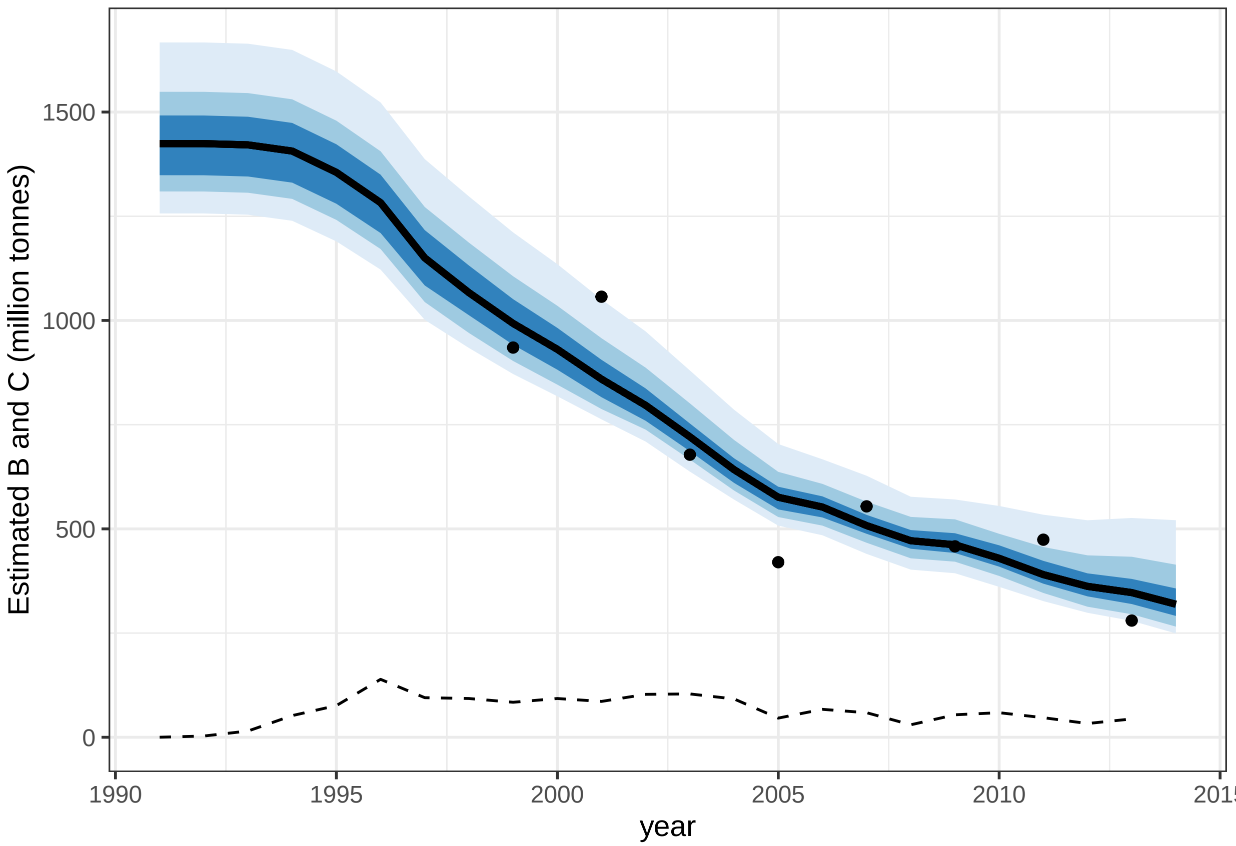
```
## 292 1994 1433.6759   13
## 293 1995 1382.6969   13
## 294 1996 1310.4836   13
## 295 1997 1178.8344   13
## 296 1998 1096.6158   13
## 297 1999 1019.0897   13
## 298 2000  952.6103   13
## 299 2001  878.5054   13
## 300 2002  812.5184   13
## 301 2003  730.1595   13
## 302 2004  647.0987   13
## 303 2005  575.7926   13
## 304 2006  549.8382   13
## 305 2007  502.5475   13
## 306 2008  462.5063   13
## 307 2009  450.6907   13
## 308 2010  414.6223   13
## 309 2011  372.7133   13
## 310 2012  341.6976   13
## 311 2013  323.7731   13
## 312 2014  294.2885   13
## 313 1991 1472.9962   14
## 314 1992 1472.8962   14
## 315 1993 1469.9013   14
## 316 1994 1455.0599   14
## 317 1995 1403.9696   14
## 318 1996 1331.3478   14
## 319 1997 1198.9214   14
## 320 1998 1115.3755   14
## 321 1999 1036.2797   14
## 322 2000  968.0550   14
## 323 2001  892.0939   14
## 324 2002  824.1580   14
## 325 2003  739.7981   14
## 326 2004  654.7057   14
## 327 2005  581.3804   14
## 328 2006  553.4497   14
## 329 2007  504.1897   14
## 330 2008  462.2164   14
## 331 2009  448.5020   14
## 332 2010  410.5188   14
## 333 2011  366.7227   14
##  [ reached 'max' / getOption("max.print") -- omitted 4467 rows ]
```

We can now plot the estimated biomass time series

```
biomass.iter %>%
  group_by(year) %>%
  median_qi(biomass, .width = c(.5, .8, .95)) %>%
  ggplot() +
  geom_lineribbon(aes(x = year, y = biomass, ymin = .lower, ymax = .upper),
                  show.legend = FALSE) +
  scale_fill_brewer() +
  #theme_bw() +
  geom_line(aes(y=harvest,x=year), data = tibble(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(scale = "none")
```

```
## 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. (Uncomment the call to geom_line() to view some 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:

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

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

```
##### Data generation
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.

```
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.

```
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.

```
# 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 gener
  # 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 <- exp(pars.iter[i,2])
    sig.i <- exp(pars.iter[i,3])

    #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 <- numeric(pyr)
```

```r
    # loop over the projection period.
    for (y in iyr:(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))
    #end loop over iterations
  #}
  res
```

```
##      year       value    type iter
## 1    1991          NA   index    1
## 2    1992          NA   index    1
## 3    1993          NA   index    1
## 4    1994          NA   index    1
## 5    1995          NA   index    1
## 6    1996          NA   index    1
## 7    1997          NA   index    1
## 8    1998          NA   index    1
## 9    1999   935.00000   index    1
## 10   2000          NA   index    1
## 11   2001 1057.00000   index    1
## 12   2002          NA   index    1
## 13   2003   678.00000   index    1
## 14   2004          NA   index    1
## 15   2005   420.00000   index    1
## 16   2006          NA   index    1
## 17   2007   554.00000   index    1
## 18   2008          NA   index    1
## 19   2009   458.00000   index    1
## 20   2010          NA   index    1
## 21   2011   474.00000   index    1
## 22   2012          NA   index    1
## 23   2013   280.00000   index    1
## 24   2014   283.33595   index    1
## 25   2015   359.23307   index    1
## 26   2016   300.97875   index    1
```

```
## 27  2017  291.35544   index    1
## 28  2018  297.94297   index    1
## 29  2019  262.32044   index    1
## 30  2020  406.55734   index    1
## 31  2021  235.75842   index    1
## 32  2022  270.27238   index    1
## 33  2023  204.42207   index    1
## 34  2024  247.79346   index    1
## 35  2025  176.28106   index    1
## 36  2026  226.06974   index    1
## 37  2027  185.40519   index    1
## 38  2028  240.85454   index    1
## 39  2029  224.99211   index    1
## 40  2030  193.64773   index    1
## 41  2031  232.47238   index    1
## 42  2032  255.01417   index    1
## 43  2033  170.25400   index    1
## 44  2034  227.92807   index    1
## 45  1991    0.10000   catch    1
## 46  1992    3.00000   catch    1
## 47  1993   15.00000   catch    1
## 48  1994   52.00000   catch    1
## 49  1995   76.00000   catch    1
## 50  1996  139.00000   catch    1
## 51  1997   95.00000   catch    1
## 52  1998   93.00000   catch    1
## 53  1999   84.00000   catch    1
## 54  2000   93.00000   catch    1
## 55  2001   86.00000   catch    1
## 56  2002  103.00000   catch    1
## 57  2003  104.00000   catch    1
## 58  2004   92.00000   catch    1
## 59  2005   46.00000   catch    1
## 60  2006   67.00000   catch    1
## 61  2007   59.00000   catch    1
## 62  2008   30.00000   catch    1
## 63  2009   54.00000   catch    1
## 64  2010   59.00000   catch    1
## 65  2011   47.00000   catch    1
## 66  2012   33.00000   catch    1
## 67  2013   44.00000   catch    1
## 68  2014   28.33360   catch    1
## 69  2015   35.92331   catch    1
## 70  2016   30.09788   catch    1
## 71  2017   29.13554   catch    1
## 72  2018   29.79430   catch    1
## 73  2019   26.23204   catch    1
## 74  2020   40.65573   catch    1
## 75  2021   23.57584   catch    1
## 76  2022   27.02724   catch    1
## 77  2023   20.44221   catch    1
## 78  2024   24.77935   catch    1
## 79  2025   17.62811   catch    1
## 80  2026   22.60697   catch    1
```

```
## 81  2027   18.54052   catch    1
## 82  2028   24.08545   catch    1
## 83  2029   22.49921   catch    1
## 84  2030   19.36477   catch    1
## 85  2031   23.24724   catch    1
## 86  2032   25.50142   catch    1
## 87  2033   17.02540   catch    1
## 88  2034   22.79281   catch    1
## 89  1991 1319.44006 biomass    1
## 90  1992 1319.34006 biomass    1
## 91  1993 1316.34946 biomass    1
## 92  1994 1301.63940 biomass    1
## 93  1995 1251.29068 biomass    1
## 94  1996 1181.36805 biomass    1
## 95  1997 1053.99286 biomass    1
## 96  1998  978.93220 biomass    1
## 97  1999  909.68828 biomass    1
## 98  2000  852.25320 biomass    1
## 99  2001  787.62941 biomass    1
## 100 2002  731.48145 biomass    1
## 101 2003  659.13248 biomass    1
## 102 2004  586.15050 biomass    1
## 103 2005  524.78281 biomass    1
## 104 2006  508.50320 biomass    1
## 105 2007  470.89158 biomass    1
## 106 2008  440.36848 biomass    1
## 107 2009  437.95745 biomass    1
## 108 2010  411.47063 biomass    1
## 109 2011  379.09658 biomass    1
## 110 2012  357.50229 biomass    1
## 111 2013  349.01103 biomass    1
## 112 2014  329.14884 biomass    1
## 113 2015  324.04531 biomass    1
## 114 2016  311.10973 biomass    1
## 115 2017  303.36875 biomass    1
## 116 2018  296.20119 biomass    1
## 117 2019  288.00714 biomass    1
## 118 2020  282.94600 biomass    1
## 119 2021  263.19119 biomass    1
## 120 2022  259.42754 biomass    1
## 121 2023  251.99877 biomass    1
## 122 2024  250.72724 biomass    1
## 123 2025  245.04457 biomass    1
## 124 2026  246.17955 biomass    1
## 125 2027  242.40265 biomass    1
## 126 2028  242.46857 biomass    1
## 127 2029  236.99347 biomass    1
## 128 2030  232.77686 biomass    1
## 129 2031  231.43935 biomass    1
## 130 2032  226.13786 biomass    1
## 131 2033  218.25655 biomass    1
## 132 2034  218.35976 biomass    1
## 133 2035  212.70205 biomass    1
```

```
#  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)
```

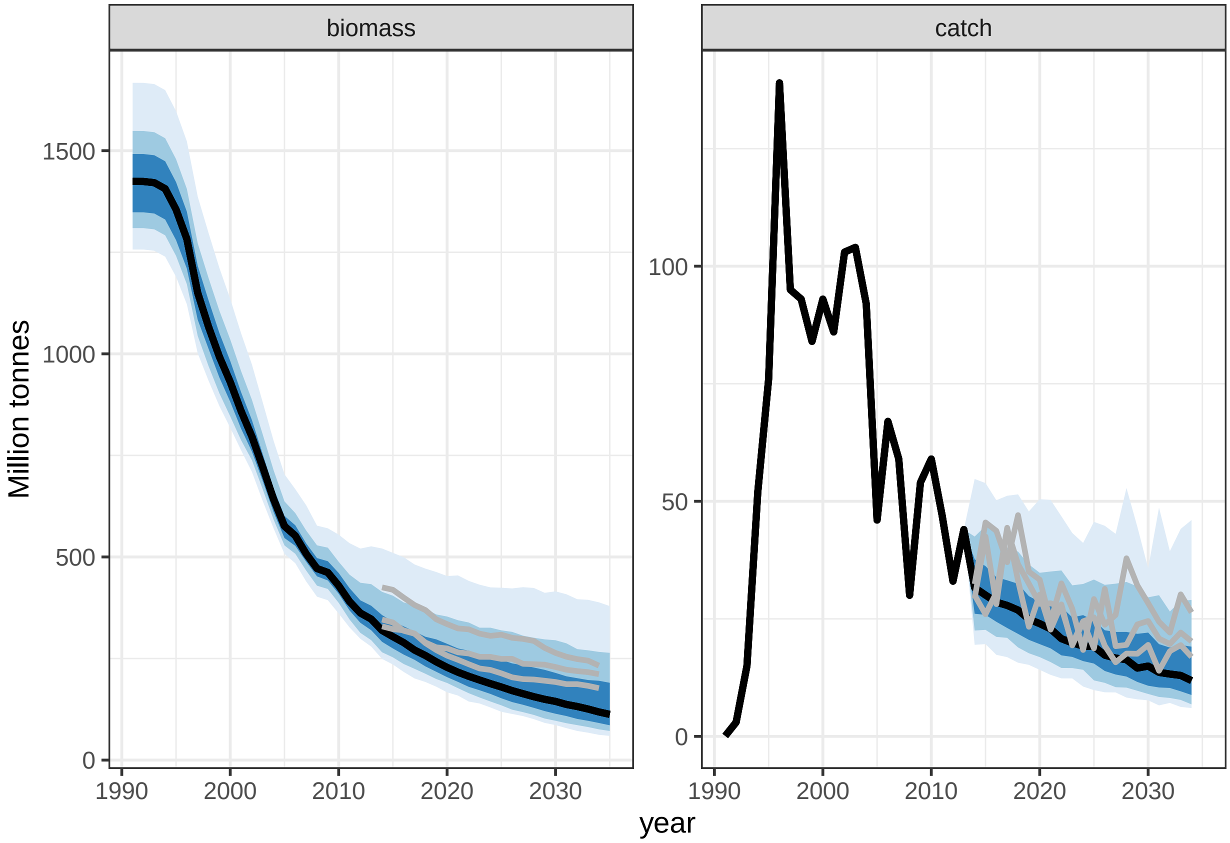
```
##       year    value    type iter
## 26595 2030 228.3833 biomass  200
## 26596 2031 223.1049 biomass  200
## 26597 2032 219.1810 biomass  200
## 26598 2033 212.3689 biomass  200
## 26599 2034 213.2140 biomass  200
## 26600 2035 205.1584 biomass  200
```

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))
  project.results %>%
    filter(type %in% c("biomass","catch")) %>%
    group_by(type, year) %>%
    median_qi(value, .width = c(.5, .8, .95)) %>%
    ggplot() +
    geom_lineribbon(aes(x = year, y = value, ymin = .lower, ymax = .upper),
                  show.legend = FALSE) +
    scale_fill_brewer() +
    geom_line(aes(y=value,x=year),data = subset(project.results, type != "index" & iter==1 & year %in% p
    geom_line(aes(y=value,x=year),data = subset(project.results, type != "index" & iter==2 & year %in% p
    geom_line(aes(y=value,x=year),data = subset(project.results, type != "index" & iter==3 & year %in% p
#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)
    facet_wrap(~type, scale = "free_y") +
    ylab("Million tonnes") +
    theme_bw()

}
```

Plot the projection:

```
projection.plot(project.fixed)
```



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
}
```

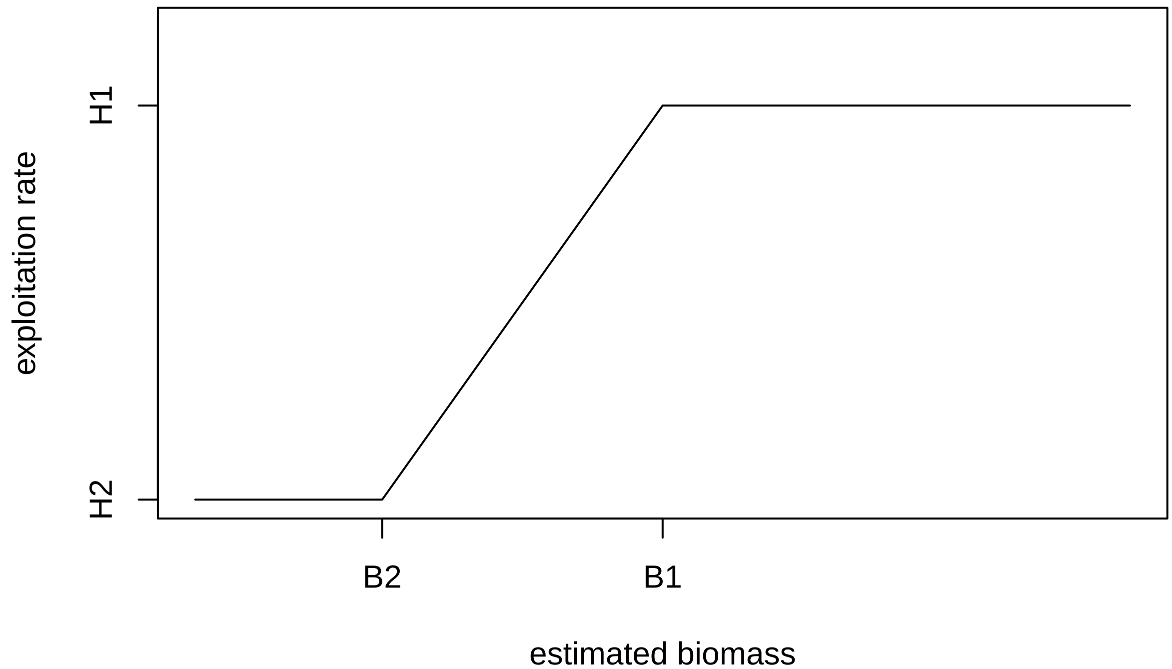
Define control parameters for HCR using reference points
We (arbitrarily) set the threshold and limit biomass reference points as 50% & 20% of the maximum observed survey index value during the historical period.
The target exploitation rate is set at 5%.

```
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()
```

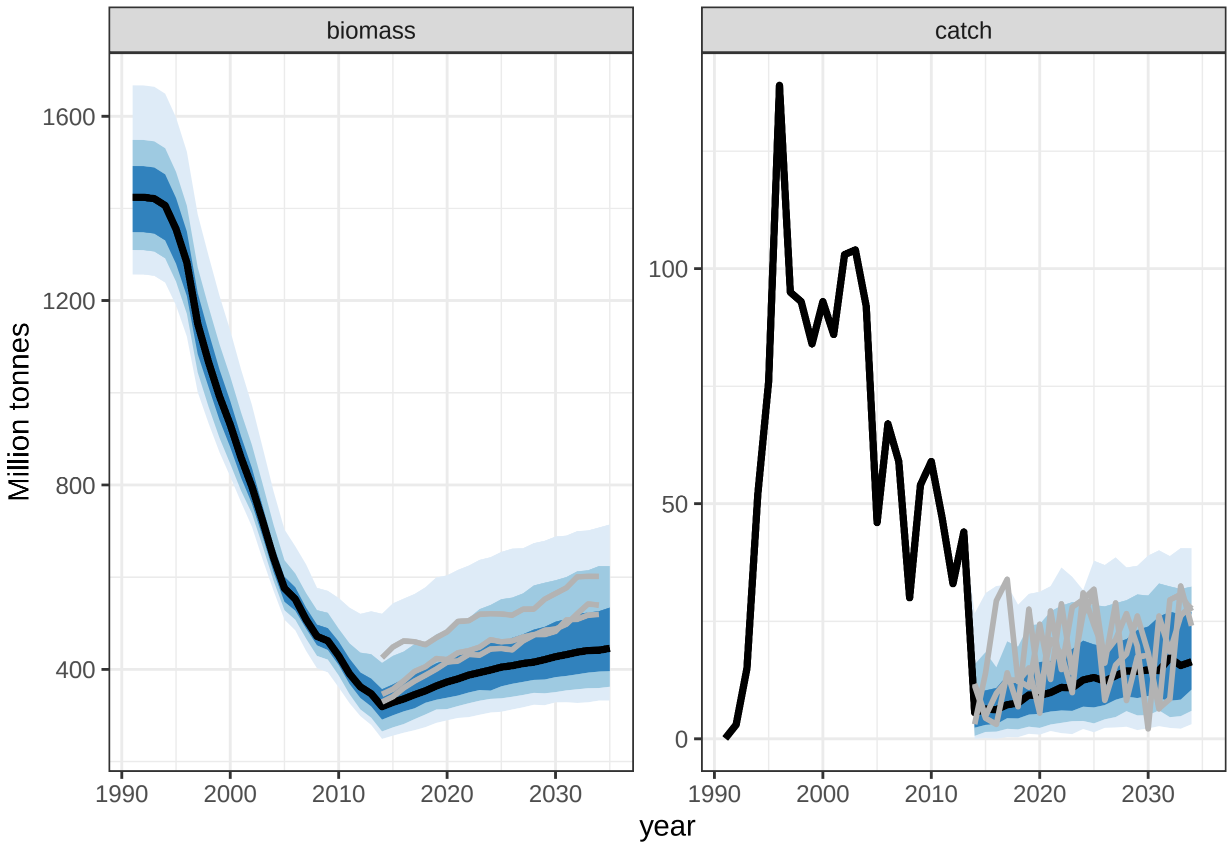


Conduct the evaluation by projecting system forward in time

```
set.seed(42)
project.hcr <- evaluate(pars.iter, biomass.iter, control.pars,
                        data.years, proj.years, niter)
```

Plot the trajectories:

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



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)

```
set.seed(42)
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

```
set.seed(42)
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:

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

Run both scenarios with this new target harvest rate

```
set.seed(42)
proj.hcr2.noerror <- evaluate(pars.iter, biomass.iter,
                              control.pars, data.years,
                              proj.years, niter,
                              overshoot = 0)
set.seed(42)
proj.hcr2.error <- evaluate(pars.iter, biomass.iter,
                            control.pars, data.years,
                            proj.years, niter,
                            overshoot = 0.2)
```

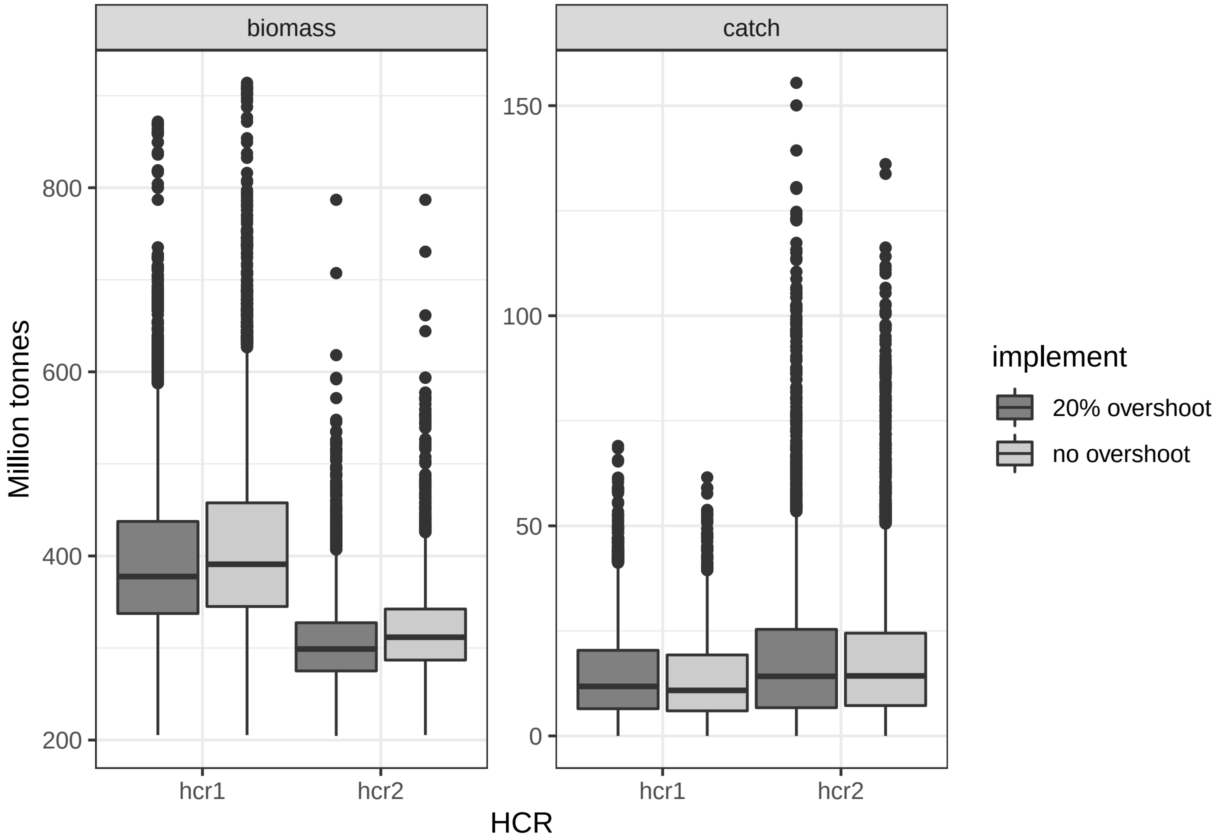
Diagnostics We have run the evaluations for 4 HCRs. We can now compare these. Create an object containing all the results:

```
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
```

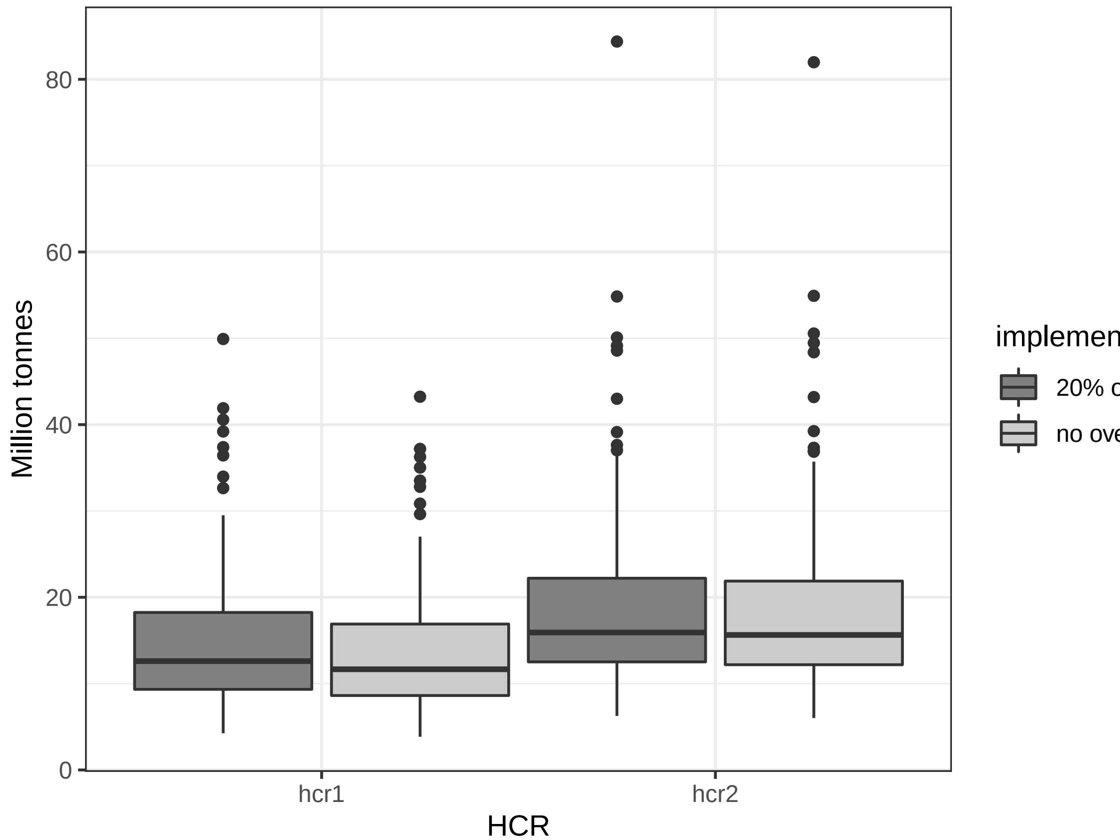
Summarize biomass & catch for all 4 options:

```
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
```



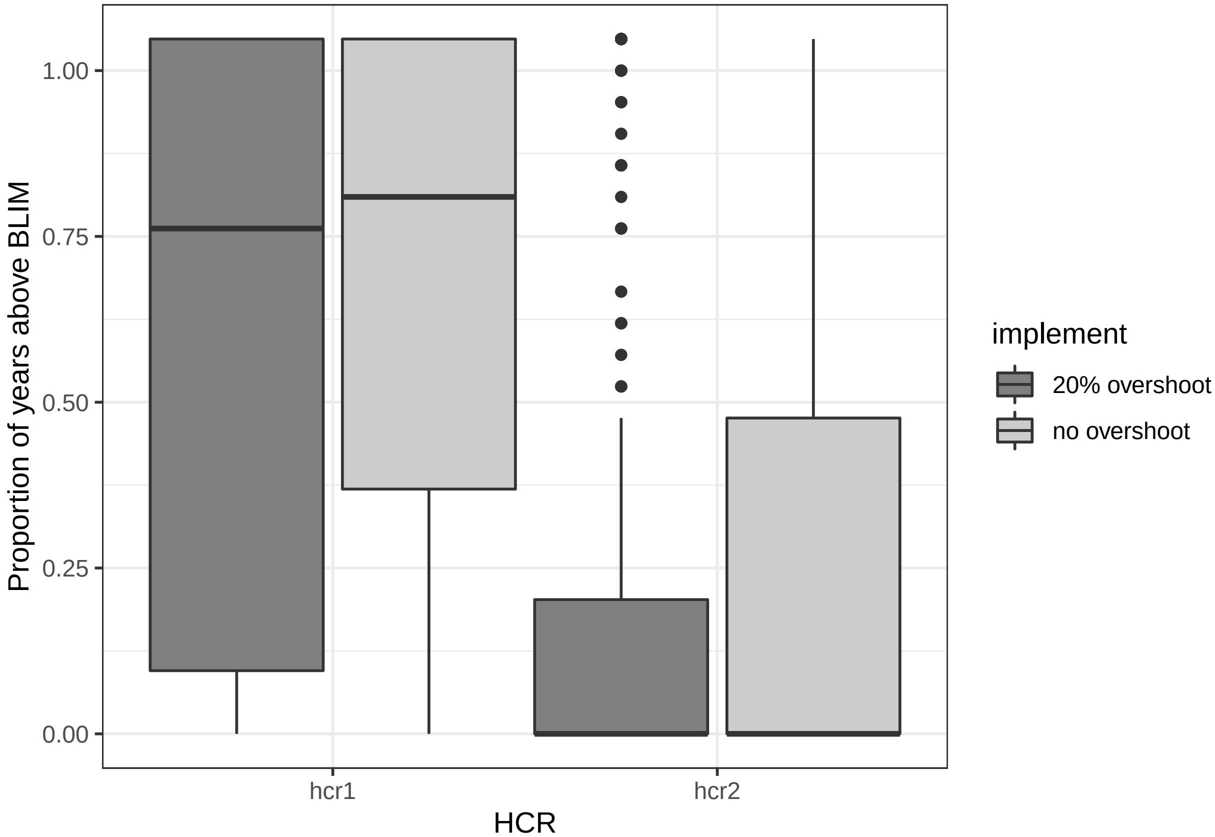
We immediately see a yield-biomass tradeoff - HCR2 gives more catch but leads to lower biomass. There is not much change when the catch is 20% higher than the TAC.

```
#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
```

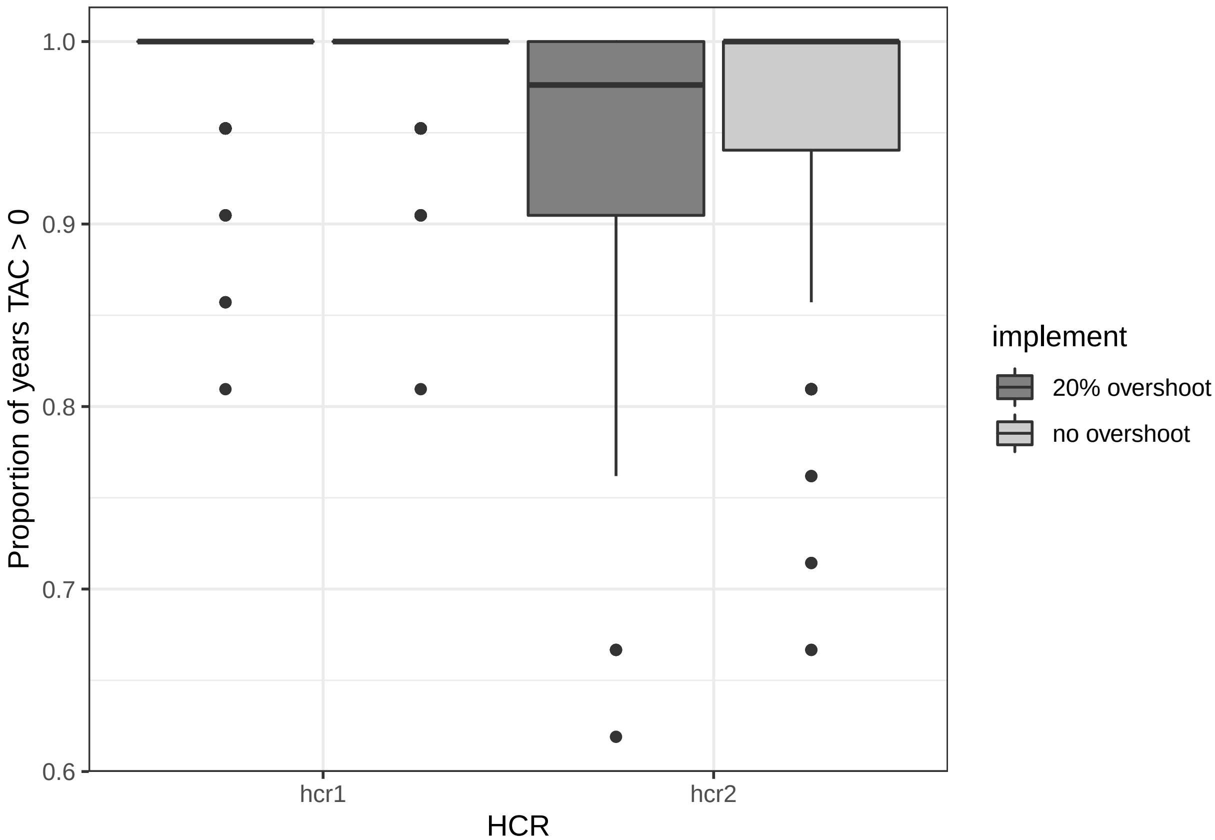


Performance statistics

```
# 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