MSE for Gulf of Bothnia herring

Max Cardinale (SLU), modified from code
provided by Iago Mosqueira for WKREBUILD and WKNEWREF

19 August, 2024

Contents

1 Gulf of Bothnina herring

GOB herring model has been developed in 2021 and reviewed by a benchmark. The latest update was run by WGBFAS in April 2024. Two different Operative Models (OMs) will be tested, a reference case (OM1; latest model accepted at WGBFAS in April 2024) and an alternative model (OM2), which assumes a reduced productivity of the stock, with R0 as XX% of the original R0 from there reference case. Reference points will be estimated separately for each OM and as an ensamble of the two OMs.

1.1 Load models created by Create FLR stock objects for MSE.R file and attributes (created by MSE_reference points_estimation_best case.R file

Load libraries

```
library(mse)
library(FLRef)
library(FLBRP)
library(progressr)
# handlers(global=TRUE)
library(doFuture)
library(r4ss)
library(ss3diags)
library(ss3om)
library(icesTAF)
library(parallel)
library(mseviz)
library(dplyr)
# devtools::install_github('mebrooks/stockrecruit/StockRecruitSET',
# build_opts = c('--no-resave-data', '--no-manual'))
```

Get the libraries specifications

```
sessionInfo()
   R version 4.2.3 (2023-03-15 ucrt)
   Platform: x86_64-w64-mingw32/x64 (64-bit)
   Running under: Windows 10 x64 (build 22631)

   Matrix products: default

   locale:
   [1] LC_COLLATE=English_United Kingdom.utf8
   [2] LC_CTYPE=English_United Kingdom.utf8
   [3] LC_MONETARY=English_United Kingdom.utf8
   [4] LC_NUMERIC=C
   [5] LC_TIME=English_United Kingdom.utf8

   attached base packages:
   [1] parallel  stats     graphics  grDevices utils     datasets  methods
   [8] base

   other attached packages:
    [1] dplyr_1.1.4          mseviz_0.2.6.9008    patchwork_1.2.0
    [4] icesTAF_4.2.0        ss3om_0.5.2.9005     ss3diags_1.10.2
    [7] r4ss_1.49.3          FLRef_1.10.4         FLSRTMB_1.1.4.9014
   [10] mse_2.2.3.9252       progressr_0.14.0     data.table_1.15.4
```

```
   [13] doFuture_1.0.1       future_1.34.0       foreach_1.5.2
   [16] FLBRP_2.5.9.9022     FLasher_0.7.1.9221  FLFishery_0.3.8.9009
   [19] ggplotFL_2.7.0.9133  ggplot2_3.4.4       FLCore_2.6.20.9204
   [22] iterators_1.0.14     lattice_0.20-45     knitr_1.48

   loaded via a namespace (and not attached):
    [1] jsonlite_1.8.8      viridisLite_0.4.2   stats4_4.2.3
    [4] yaml_2.3.10         ggrepel_0.9.5       globals_0.16.3
    [7] pillar_1.9.0        glue_1.7.0          TAF_4.2.0
   [10] digest_0.6.36       colorspace_2.1-1    cowplot_1.1.3
   [13] htmltools_0.5.8.1   Matrix_1.6-5        pkgconfig_2.0.3
   [16] listenv_0.9.1       purrr_1.0.2         xtable_1.8-4
   [19] corpcor_1.6.10      scales_1.3.0        svglite_2.1.3
   [22] tibble_3.2.1        generics_0.1.3      withr_3.0.1
   [25] furrr_0.3.1         TMB_1.9.14          cli_3.6.3
   [28] magrittr_2.0.3      evaluate_0.24.0     data.tree_1.1.0
   [31] fansi_1.0.6         parallelly_1.38.0   MASS_7.3-58.2
   [34] xml2_1.3.6          tools_4.2.3         gh_1.4.1
   [37] formatR_1.14        lifecycle_1.0.4     stringr_1.5.1
   [40] munsell_0.5.1       kableExtra_1.4.0    compiler_4.2.3
   [43] systemfonts_1.1.0   tinytex_0.52        rlang_1.1.4
   [46] grid_4.2.3          rstudioapi_0.16.0   rmarkdown_2.27
   [49] gtable_0.3.5        codetools_0.2-19    roxygen2_7.3.2
   [52] R6_2.5.1            gridExtra_2.3       fastmap_1.2.0
   [55] future.apply_1.11.2 utf8_1.2.4          stringi_1.8.4
   [58] Rcpp_1.0.13         vctrs_0.6.5         tidyselect_1.2.1
   [61] xfun_0.46           coda_0.19-4.1
```

Define folder with R data files and other additional parameters

```
setwd("~/Max/Commitees/ICES/WGBFAS/2024/GBH")

plan(multisession, workers = 9)

its <- 500
fy <- 2082
iy <- 2024

basecase <- mget(load("Reference_run.rda"))
R0reduced <- mget(load("Reference_run_R0_reduced.rda"))
load("GOB_herring_attributes.rda")
```

1.2 OMs conditioning, defining FLStocks, FLSRs and SS3 refpts

```
stks <- FLStocks(REF = basecase$stk, RED = R0reduced$stk)
srrs <- FLSRs(REF = basecase$srr, RED = R0reduced$srr)
srps <- list(REF = basecase$rps, RED = R0reduced$rps)
brps <- list(REF = basecase$brp, RED = R0reduced$brp)
```

Define functions

```
getabSR <- function(stk, srr) {
    ab(fmle(as.FLSR(stk, model = "bevholtSV"), fixed = list(s = params(srr)$s,
```

```
        v = params(srr)$v, spr0 = params(srr)$v/params(srr)$R0)))
}

getBRPs <- function(stk, srr) {
    # COERCE FLSR as bevholt(a,b)
    nsr <- getabSR(stk, srr)

    # FIT brps
    brp <- brp(FLBRP(stk, sr = nsr))

    # EXTRACT brefpts
    brps <- remap(refpts(brp), R0 = c("virgin", "rec"), MSY = c("msy",
        "yield"))

    return(brps)
}
```

1.2.1 This part is only to hack Blim up to line 160

Load the stock object in FLR

```
stk_single <- readFLSss3(dir = "~/Max/Commitees/ICES/WGBFAS/2024/GBH/Reference_run",
    wtatage = TRUE)
```

Load the SS model and parameters

```
out <- SS_output(dir = "~/Max/Commitees/ICES/WGBFAS/2024/GBH/Reference_run",
    covar = T, printstats = FALSE)
R0 <- exp(out$parameters$Value[out$parameters$Label == "SR_LN(R0)"])
s <- out$parameters$Value[out$parameters$Label == "SR_BH_steep"]
sigmaR <- out$parameters$Value[out$parameters$Label == "SR_sigmaR"]
rho <- out$parameters$Value[out$parameters$Label == "SR_autocorr"]
B0 <- out$derived_quants$Value[out$derived_quants$Label == "SSB_unfished"]
SSBcv <- out$derived_quants$StdDev[out$derived_quants$Label ==
    "SSB_2023"]/out$derived_quants$Value[out$derived_quants$Label ==
    "SSB_2023"]
Fcv <- out$derived_quants$StdDev[out$derived_quants$Label ==
    "F_2023"]/out$derived_quants$Value[out$derived_quants$Label ==
    "F_2023"]
BMSYss <- out$derived_quants$Value[out$derived_quants$Label ==
    "SSB_MSY"]
FMSYss <- out$derived_quants$Value[out$derived_quants$Label ==
    "annF_MSY"]
MSYss <- out$derived_quants$Value[out$derived_quants$Label ==
    "Dead_Catch_MSY"]
TB0ss <- out$derived_quants$Value[out$derived_quants$Label ==
    "Totbio_unfished"]
```

Coerce FLSR as bevholt(a,b) from SS

```
nsr <- ab(fmle(as.FLSR(stk, model = "bevholtSV"), fixed = list(s = s,
    v = B0, spr0 = B0/R0)))
     Nelder-Mead direct search function minimizer
   function value for initial parameters = -27.508534
     Scaled convergence tolerance is 4.09909e-07
```

```
   Stepsize computed as 0.100000
   Exiting from Nelder Mead minimizer
       1 function evaluations used
```

Fit brps as single stock to hack Blim for the oms list. Blim set as 32% of B0

```
brp_single <- brp(FLBRP(stk, sr = nsr))
brps_single <- remap(refpts(brp_single), R0 = c("virgin", "rec"),
    MSY = c("msy", "yield"))
brps_single$Blim <- B0 * 0.31787
```

2 Get Brps and add Blim to refpts

```
brps <- lapply(srps, rbind, Blim = brps_single$Blim)
```

3 Change rpts for the for the reduced R0 model and fix the units

```
brps$RED[1] <- brpred[5]
brps$RED[2] <- brpred[4]
brps$RED[4] <- brpred[2]
brps$RED[5] <- brpred[1]
brps$RED[6] <- brpred[7]

brps["RED"]$RED@units <- brps["REF"]$REF@units
```

3.1 Coerce SRRs to ab bevholt

```
nsrs <- Map(function(x, y) getabSR(x, y), x = stks, y = srrs)
     Nelder-Mead direct search function minimizer
   function value for initial parameters = -31.613710
     Scaled convergence tolerance is 4.71081e-07
   Stepsize computed as 0.100000
   Exiting from Nelder Mead minimizer
       1 function evaluations used
     Nelder-Mead direct search function minimizer
   function value for initial parameters = -31.613710
     Scaled convergence tolerance is 4.71081e-07
   Stepsize computed as 0.100000
   Exiting from Nelder Mead minimizer
       1 function evaluations used
```

3.2 Create FLoms

```
oms <- Map(function(x, y, z) FLom(stock = x, sr = y, refpts = z),
    x = stks, y = srrs, z = brps)
```

3.3 Extend to the future

```
oms <- lapply(oms, function(x) propagate(fwdWindow(x, end = fy),
    its))
```

3.4 Add SR deviances, same across OMs, and add those to OMs

```
devs <- rlnormar1(its, sdlog = sigmaR, rho = rho, years = seq(2023,
    fy))

oms <- lapply(oms, function(x) {
    deviances(x) <- devs
    return(x)
})
```

3.5 F and SSB deviances for shortcut and STF

```
sdevs <- shortcut_devs(oms[[1]], Fcv = Fcv, Fphi = 0.423, SSBcv = SSBcv)
```

4 MP setup

4.1 Setup standard ICES advice rule

```
arule <- mpCtrl(list(

  # (est)imation method: shortcut.sa + SSB deviances
  est = mseCtrl(method=shortcut.sa,
    args=list(SSBdevs=sdevs$SSB)),

  # hcr: hockeystick (fbar ~ ssb | lim, trigger, target, min)
  hcr = mseCtrl(method=hockeystick.hcr,
    args=list(lim=0, trigger=trigger, target=target,
    min=0, metric="ssb", output="fbar")),

  # (i)mplementation (sys)tem: tac.is (C ~ F) + F deviances
  # rec as GM ignoring last 2 years
  isys = mseCtrl(method=tac.is,
    args=list(recyrs=-2, fmin=0, Fdevs=sdevs$F))
  ))
```

Plotting the HCR

```
setwd("~/Max/Commitees/ICES/WGBFAS/2024/GBH")
plot_hockeystick.hcr(arule$hcr, labels = c(trigger = "Btrigger",
    target = "Ftarget")) + xlab("SSB (t)") + ylab(expression(bar(F))) +
    ggtitle("HCR")
```

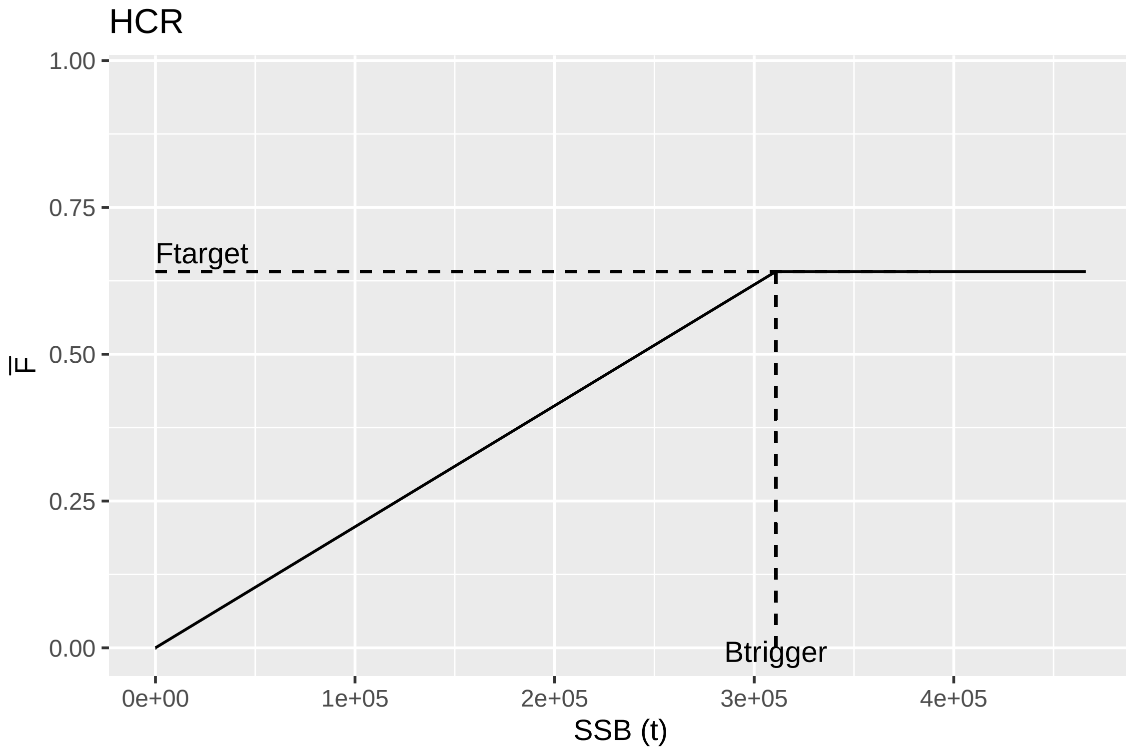

Figure 1: Harvest control rule used in the MSE

```
mseargs <- list(iy = iy - 1, fy = fy, data_lag = 1, management_lag = 1,
    frq = 1)
```

Get candidate values for Btrigger & Ftarget creating combinations of based on fraction of B0 (FBx ~ Bx * c(0.30, 0.45, 0.05)) and adding FMSY ~ BMSY * c(0.6, 0.8, 1)

```
frps <- Map(function(x, y) {
    lapply(seq(30, 55, by = 5), function(i) {
        Fbrp(computeFbrp(x, sr = y, proxy = "bx", x = i, blim = 0.31787))
    })
}, x = stks, y = nsrs)
   Computing Fsb30 with Btgt = Bsb30

     Blim = 0.31787B0
   Computing Fsb35 with Btgt = Bsb35

     Blim = 0.31787B0
   Computing Fsb40 with Btgt = Bsb40

     Blim = 0.31787B0
   Computing Fsb45 with Btgt = Bsb45

     Blim = 0.31787B0
   Computing Fsb50 with Btgt = Bsb50

     Blim = 0.31787B0
   Computing Fsb55 with Btgt = Bsb55

     Blim = 0.31787B0
   Computing Fsb30 with Btgt = Bsb30

     Blim = 0.31787B0
   Computing Fsb35 with Btgt = Bsb35

     Blim = 0.31787B0
   Computing Fsb40 with Btgt = Bsb40

     Blim = 0.31787B0
   Computing Fsb45 with Btgt = Bsb45

     Blim = 0.31787B0
   Computing Fsb50 with Btgt = Bsb50

     Blim = 0.31787B0
   Computing Fsb55 with Btgt = Bsb55

     Blim = 0.31787B0

opts <- Map(function(x, y) {
    res <- list(target = rep(unlist(lapply(x, "[", 1)), each = 3),
        trigger = unlist(lapply(seq(0.3, 0.55, by = 0.05), function(i) (c(refpts(y)$SB0) *
            i) * c(0.6, 0.8, 1))))
    res$target <- c(res$target, rep(c(refpts(y)$FMSY), 3))
```

```
    res$trigger <- c(res$trigger, c(refpts(y)$SBMSY) * c(0.6,
        0.8, 1))
    return(res)
}, x = frps, y = oms)

nms <- c(paste0(rep(paste0("FB", seq(30, 55, by = 5)), each = 3),
    rep(c(".6", ".8", "1"), 6)), c("FMSY.6", "FMSY.8", "FMSY1"))
```

Run for all options on 'hcr' control element for all OMs and ICES advice. To be done on a cluster pc

```
system.time(plans <- Map(function(x, y) {

    runs <- mps(x, ctrl = arule, args = mseargs, hcr = y, names = nms)

    runs$advice <- advice <- mp(x, ctrl = arule, args = mseargs)

    return(runs)

}, x = oms, y = opts))
```

Save the MSE results

```
save(oms, plans, file = "plans.rda", compress = "xz")
```

4.2 Compute MSE performances

Define performances metrics

```
metrics <- list(SB = ssb, F = fbar, C = landings, TC = catch,
    Rec = rec)

stats <- list(medianFmsy = list(~yearMedians(F/FMSY), name = "F/Fmsy",
    desc = "Median annual F/Fmsy"), medianBmsy = list(~yearMedians(SB/SBMSY),
    name = "B/Bmsy", desc = "Median annual B/Bmsy"), medianCmsy = list(~yearMedians(C/MSY),
    name = "Catch/MSY", desc = "Median Catch/MSY over years"),
    aavC = list(~yearMedians(iav(C)), name = "AAV", desc = "Median annual variation in catches"),
    riskBlim = list(~apply(iterMeans((SB/Blim) < 1), 1, max),
        name = "P3(B<Blim)", desc = "Probability that SSB < Blim"),
    risk10SB0 = list(~apply(iterMeans((SB/(SB0 * 0.1)) < 1),
        1, mean), name = "P(B<SB0.10)", desc = "Probability that SSB < 10% SB0"),
    P80BMSY = list(~apply(iterMeans((SB/(SBMSY * 0.8)) > 1),
        1, max), name = "B>80Bmsy", desc = "Probability that SSB > 80% x Bmsy"),
    medianSBMSY = list(~yearMedians(SB/SBMSY), name = "SSB/SSB[MSY]",
        desc = "Median annual SSB/SSBmsy"), medianFMSY = list(~yearMedians(F/FMSY),
        name = "F/F[MSY]", desc = "Median annual F/FMSY"))
```

Compute performances, add HCR parameters Btrigger and Ftarget and define long time horizon for evaluation

```
plans_perf <- lapply(plans, function(x) {

    res <- performance(x, statistics = stats, years = list(all = 2024:fy,
        long = 2053:fy))

    hps <- rbindlist(lapply(x, function(i) args(control(i)$hcr)[c("trigger",
        "target")]), idcol = "mp")

```

```
    performance(x) <- merge(res, hps, by = "mp")

    return(x)
})
```

Create MSE performance table

```
perf <- rbindlist(lapply(plans_perf, performance), idcol = "om")
write.csv(perf, file = "msePerf_data.csv")
```

Save performances objects

```
save(oms, plans, plans_perf, file = "plans_oms_new.rda", compress = "xz")
```

Load performances objects derived from a run made on a 16 cores cluster computer on Linux

```
load("~/Max/Commitees/ICES/WGBFAS/2024/GBH/MSE/plans_oms_new.rda")
```

5 MSE Performance plots and tables

```
ncol = length(unique(perf$mp)) # n colors
perfom1 = perf[perf$om=="REF",]
pbp = plotBPs(perfom1[perfom1$year=="long",],
statistics=c("medianFmsy","medianBmsy","medianCmsy", "aavC", "riskBlim", "P80BMSY"),
size=3, target = c(medianFmsy=1,medianBmsy=1, medianCmsy=1),
limit= c(riskBlim=0.05,P80BMSY=0.95),
yminmax = c(0.05, 0.95))+theme_bw()+
facet_wrap(~name,scales = "free_y",ncol=2)+
ggtitle(paste0("Performance: Long"))+
ylab("Performance statistics")+
scale_fill_manual(values=ss3col(ncol))+ # USE FLRef::ss3col
theme(axis.text.x=element_blank())+xlab("Candidates")
pbp
```

```
ncol = length(unique(perf$mp)) # n colors
perfom2 = perf[perf$om=="RED",]
pbp = plotBPs(perfom2[perfom2$year=="long",],
statistics=c("medianFmsy","medianBmsy","medianCmsy", "aavC", "riskBlim", "P80BMSY"),
size=3, target = c(medianFmsy=1,medianBmsy=1, medianCmsy=1),
limit= c(riskBlim=0.05,P80BMSY=0.95),
yminmax = c(0.05, 0.95))+theme_bw()+
facet_wrap(~name,scales = "free_y",ncol=2)+
ggtitle(paste0("Performance: Long"))+
ylab("Performance statistics")+
scale_fill_manual(values=ss3col(ncol))+ # USE FLRef::ss3col
theme(axis.text.x=element_blank())+xlab("Candidates")
pbp
```

```
ncol = length(unique(perf$mp)) # n colors
pbp = plotBPs(perf[perf$year=="long",],statistics=c("medianFmsy","medianBmsy","medianCmsy", "aavC", "ri
              target = c(medianFmsy=1,medianBmsy=1, medianCmsy=1),
              limit= c(riskBlim=0.05,P80BMSY=0.95),
              yminmax = c(0.05, 0.95))+theme_bw()+
  facet_wrap(~name,scales = "free_y",ncol=2)+
```

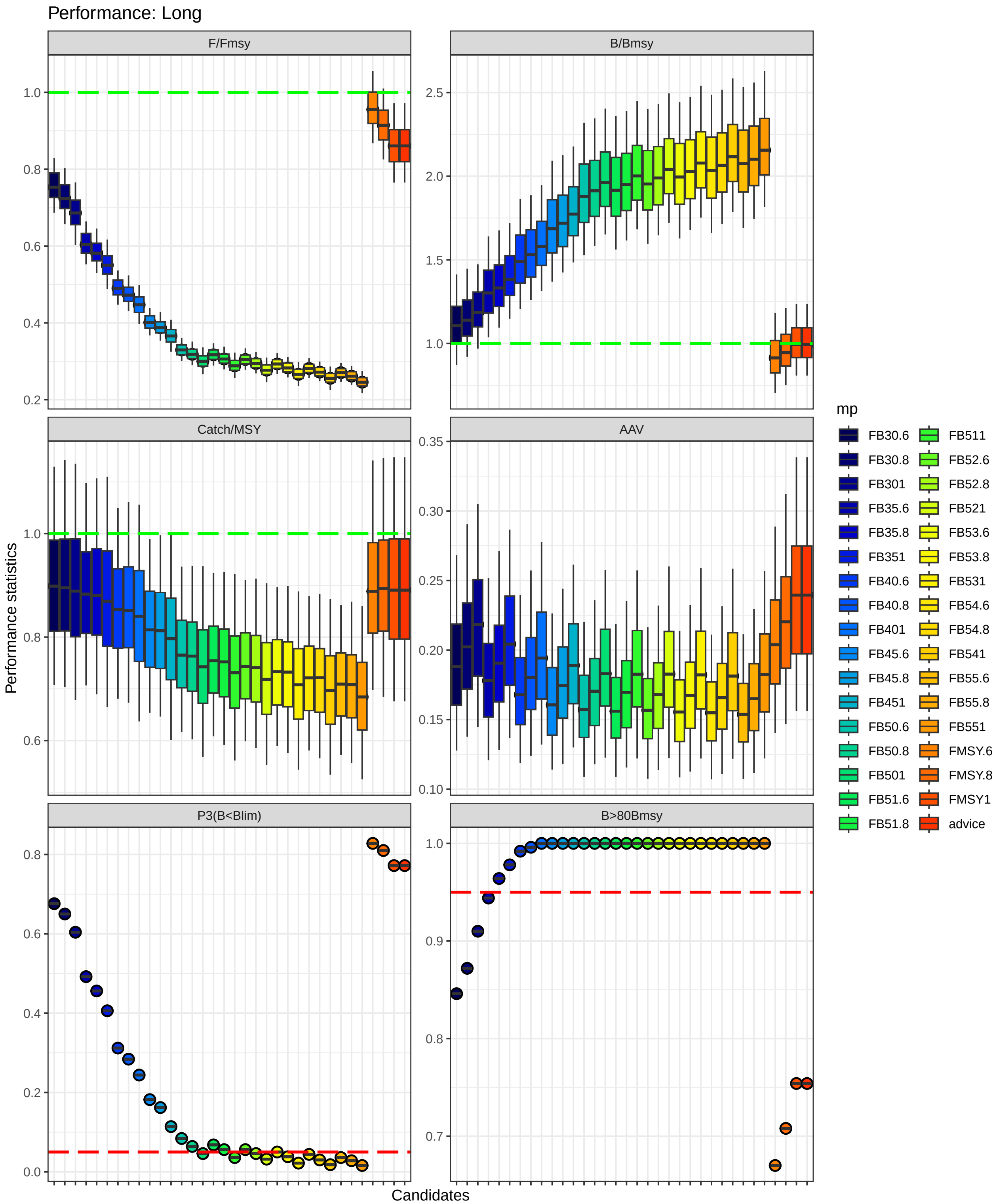

Figure 2: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). MSE performance plot for OM1

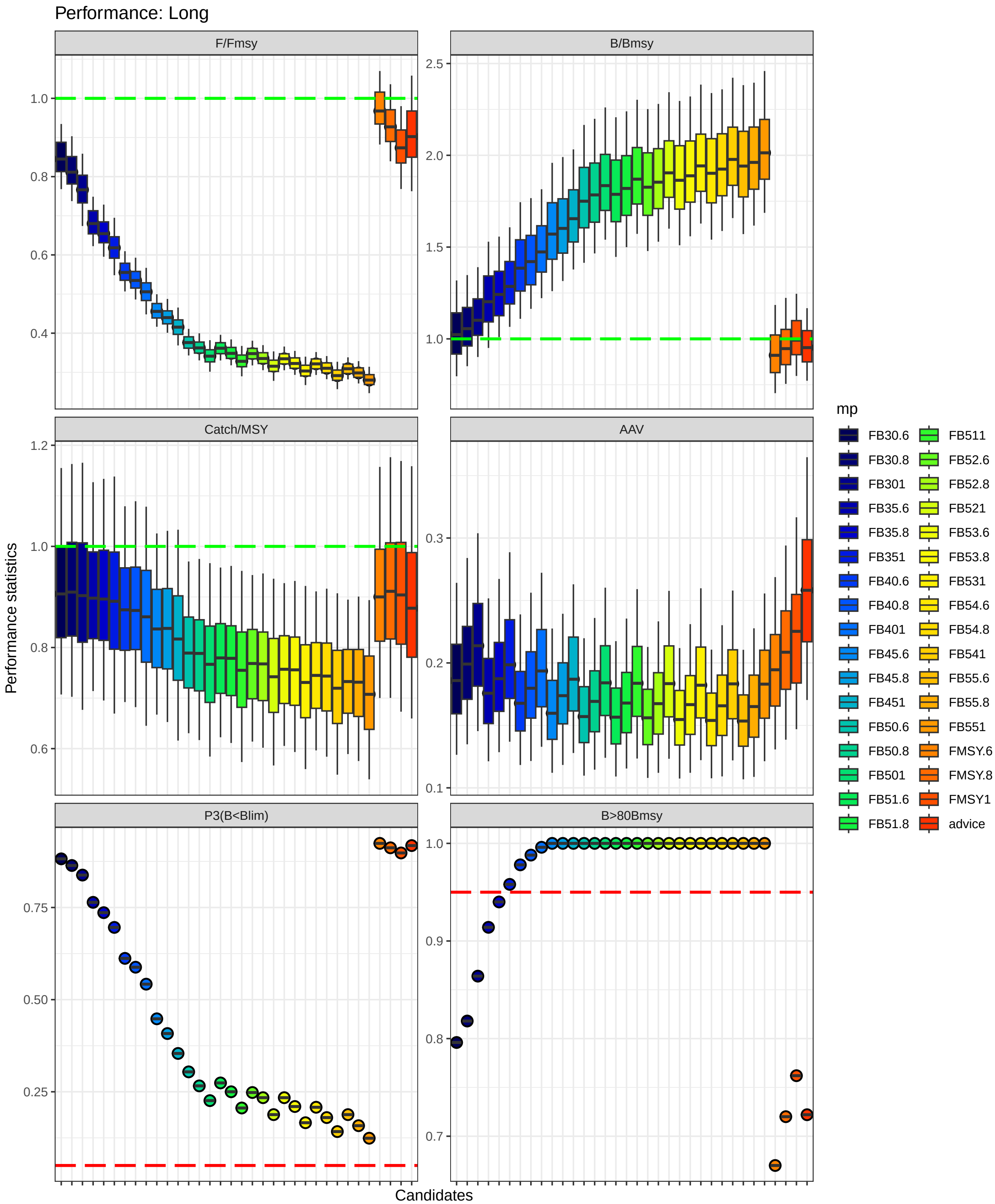

Figure 3: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). MSE performance plot for OM2

```r
  ggtitle(paste0("Performance: Long"))+
  ylab("Performance statistics")+
  scale_fill_manual(values=ss3col(ncol))+ # USE FLRef::ss3col
  theme(axis.text.x=element_blank())+xlab("Candidates")
pbp
```

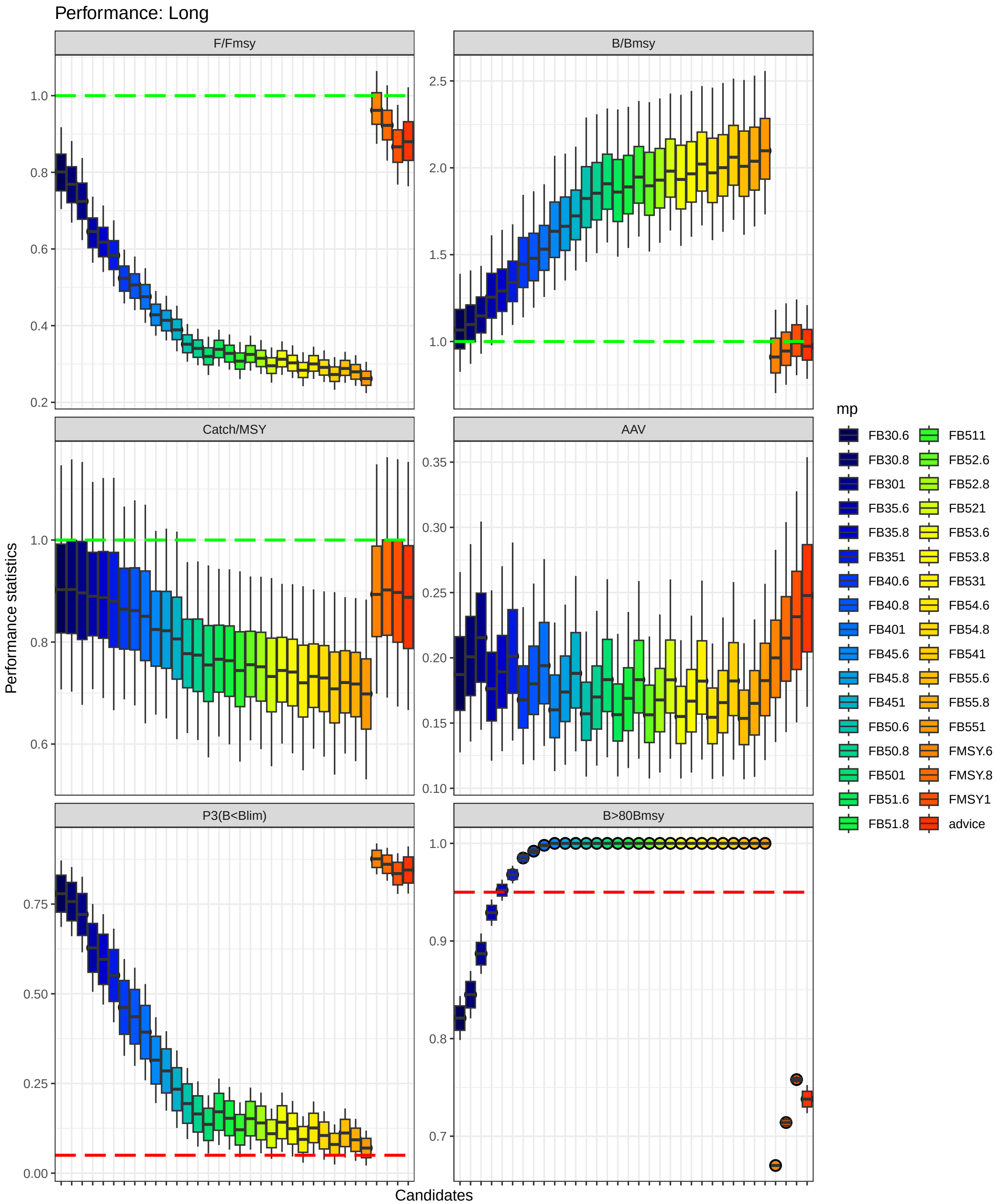

Figure 4: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). MSE performance plot, ensemble

5.1 MSE Table

```
perf_table <- perf %>%                                  # Specify data frame
  group_by(om, name, mp, year, trigger, target) %>%     # Specify group indicator
  summarise_at(vars(data),                              # Specify column
               list(Median = median),na.rm = TRUE)  # Specify function

write.csv(perf_table,file="msePerf_table.csv")
```

5.2 MSE kobe plot

```
kbcex =function(){theme(plot.title = element_text(size=10),
                        legend.key.size = unit(0.3, 'cm'), #change legend key size
                        legend.key.height = unit(0.4, 'cm'), #change legend key height
                        legend.key.width = unit(0.4, 'cm'), #change legend key width
                        legend.text = element_text(size=10)) #change legend text font size
}
kobeMPs(perf,y="medianFmsy", x="medianBmsy", SBlim=NULL, Ftarget = 1)+
  ylab(expression(F/F[MSY]))+xlab(expression(B/B[MSY]))+ylim(0,2.5)+kbcex()+theme()
   Scale for y is already present.
   Adding another scale for y, which will replace the existing scale.
```

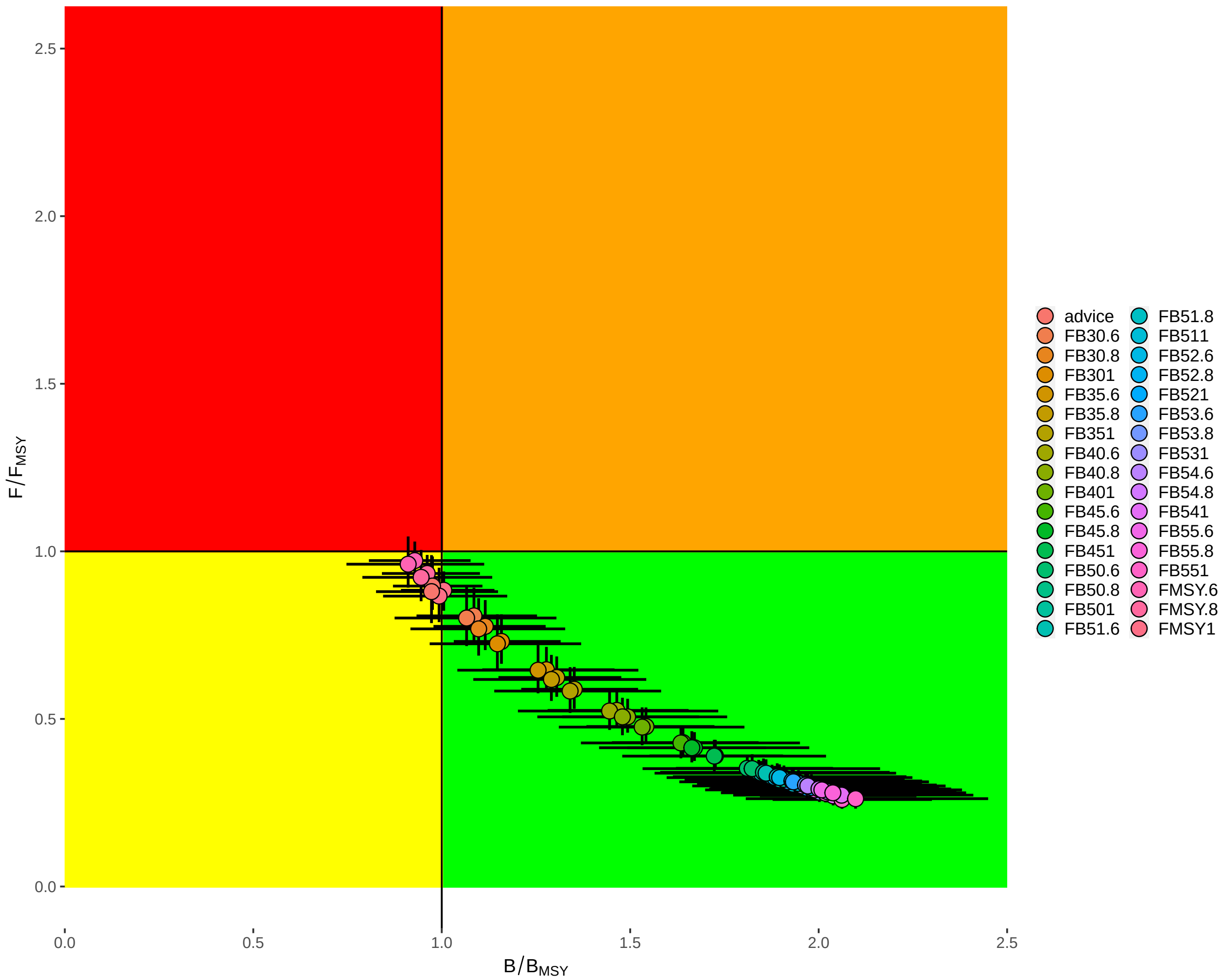

Figure 5: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). MSE kobe plot, ensemble

5.3 Trajectories plot for each OM

```r
plot(oms[[1]], plans[[1]])
```

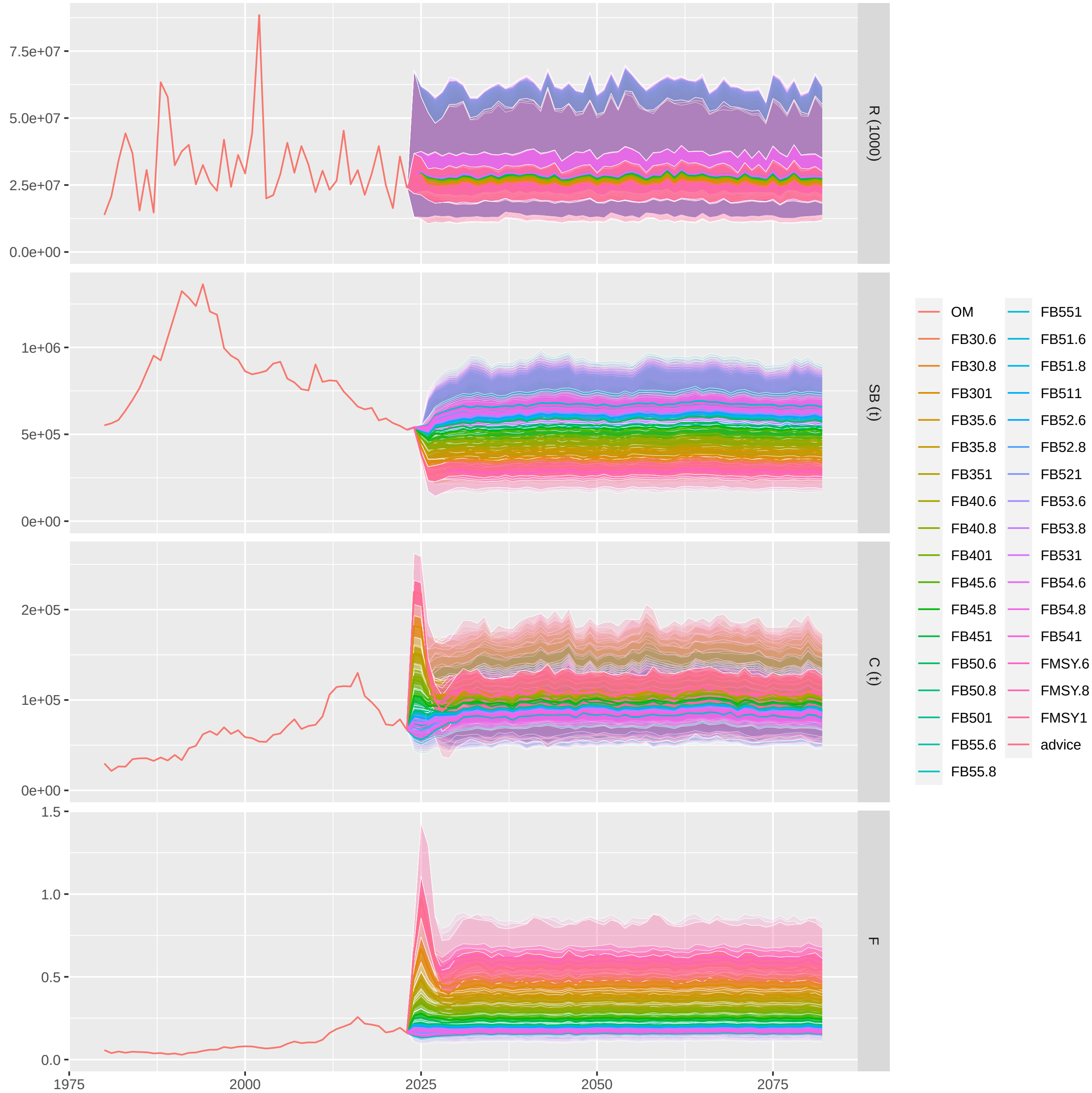

Figure 6: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for OM1

```r
plot(oms[[2]], plans[[2]])
```

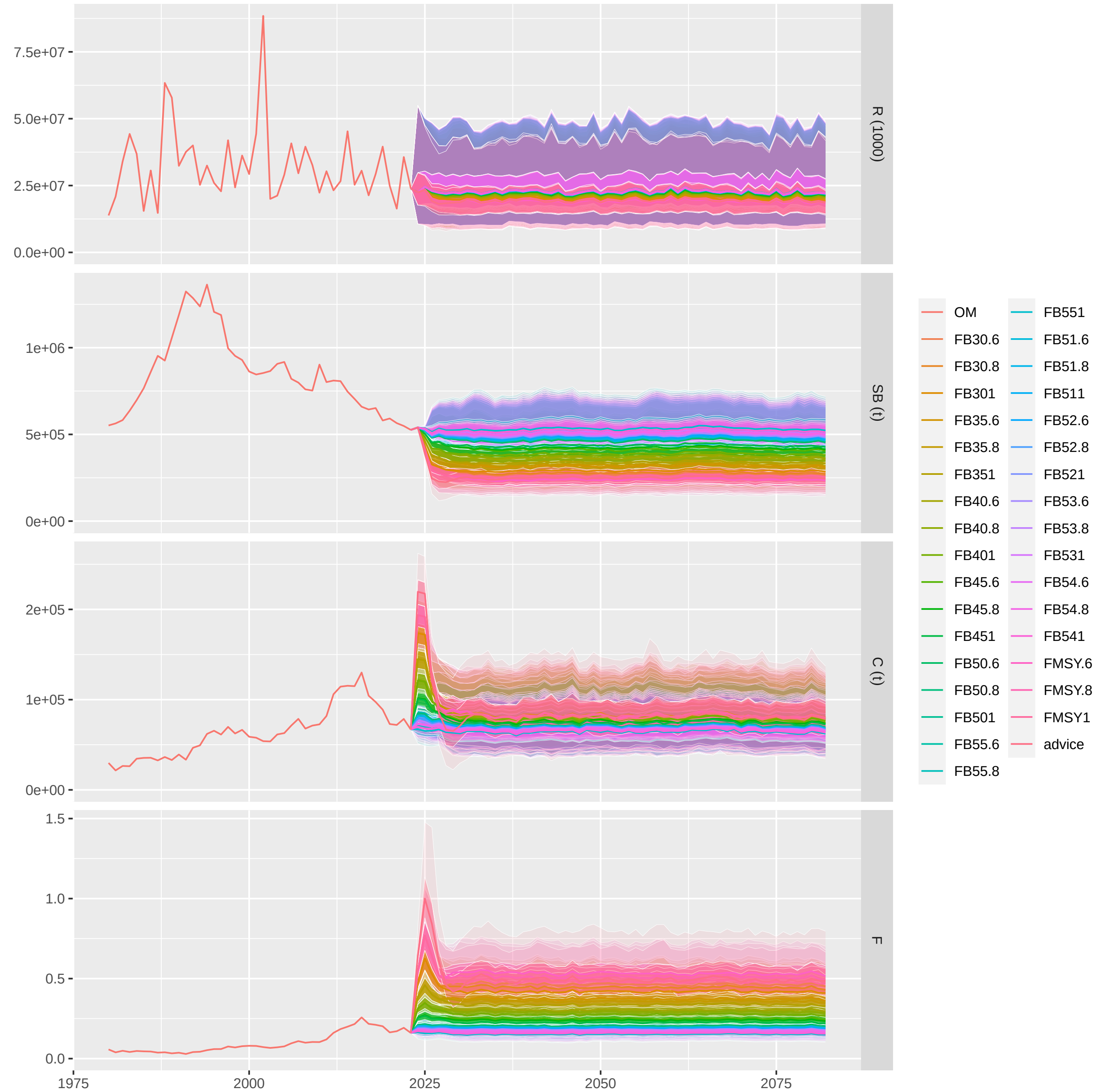

Figure 7: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for OM2

5.4 Plotting single trajectories and medians for each OMs

```
rp = oms$REF@refpts
om1 = window(FLStockR(oms[[1]]@stock), end = 2023)
om1@refpts = FLPar(Fmsy = rp["FMSY"], Bmsy = rp["SBMSY"], Blim = rp["Blim"],
    MSY = rp["MSY"])
stks1 = FLStocks(lapply(plans[[1]], function(x) {
    out = FLStockR(x@om@stock)
    out@refpts = om1@refpts
    out
}))

res1 = FLStocks(c(FLStocks(om1), stks1))
names(res1)[1] = "om1"

# Medians
med1 = FLStocks(lapply(res1, function(x) {
    iterMedians(x)
}))
```

```
rp = oms$RED@refpts
om2 = window(FLStockR(oms[[2]]@stock), end = 2023)
om2@refpts = FLPar(Fmsy = rp["FMSY"], Bmsy = rp["SBMSY"], Blim = rp["Blim"],
    MSY = rp["MSY"])
stks2 = FLStocks(lapply(plans[[2]], function(x) {
    out = FLStockR(x@om@stock)
    out@refpts = om1@refpts
    out
}))

res2 = FLStocks(c(FLStocks(om2), stks2))
names(res2)[1] = "om2"

# Medians
med2 = FLStocks(lapply(res2, function(x) {
    iterMedians(x)
}))
```

```
plotAdvice(med1) + geom_line(linewidth = 0.8) + scale_color_manual(values = c("black",
    sscol(length(stks1)))) + scale_x_continuous(breaks = seq(1960,
    fy, 5)) + theme(axis.text.x = element_text(size = 8, angle = 90)) +
    guides(col = guide_legend(ncol = 1))
```

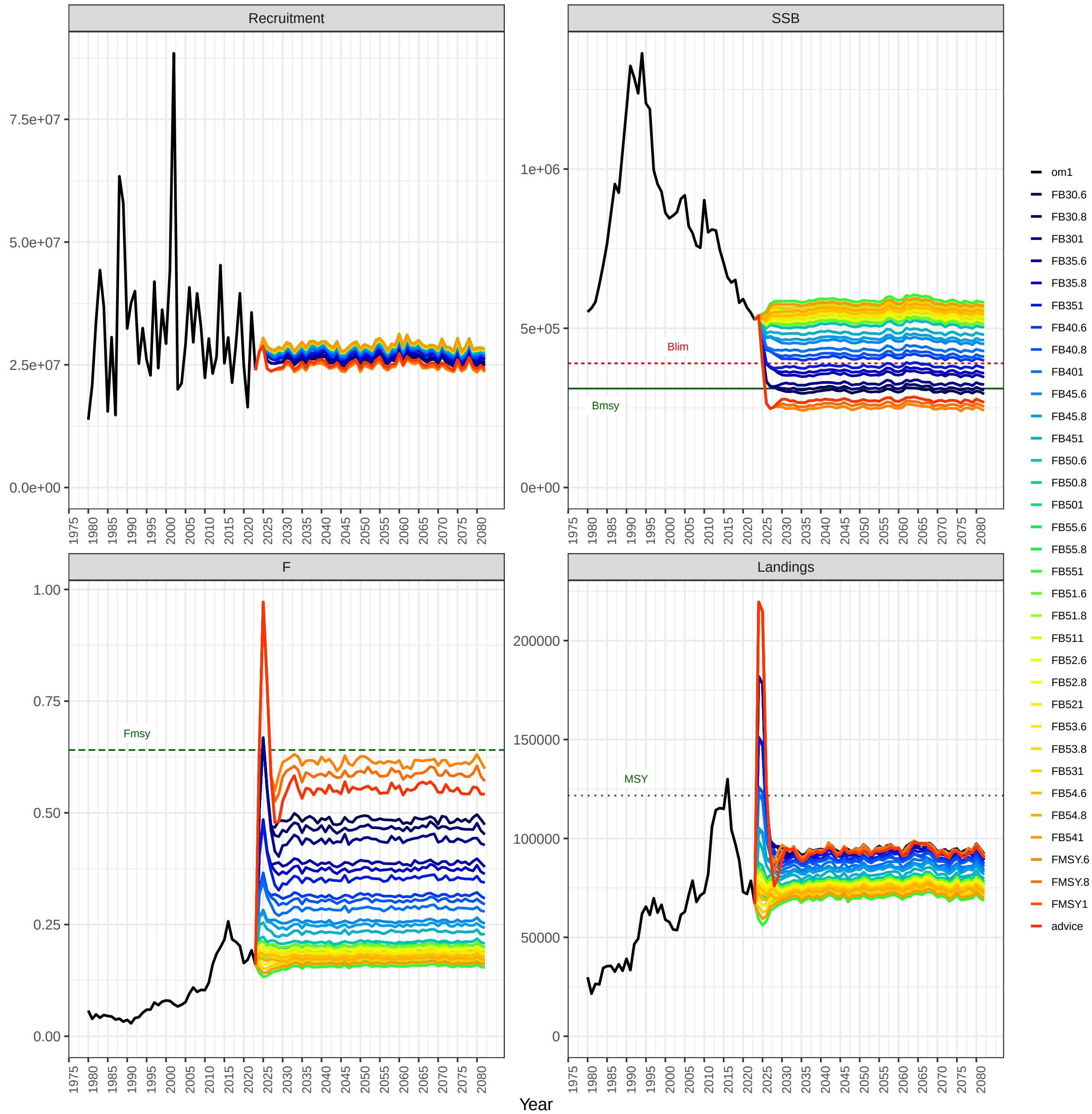

Figure 8: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for OM1 medians

```r
plotAdvice(med2) + geom_line(linewidth = 0.8) + scale_color_manual(values = c("black",
    sscol(length(stks2)))) + scale_x_continuous(breaks = seq(1960,
    fy, 5)) + theme(axis.text.x = element_text(size = 8, angle = 90)) +
    guides(col = guide_legend(ncol = 1))
```

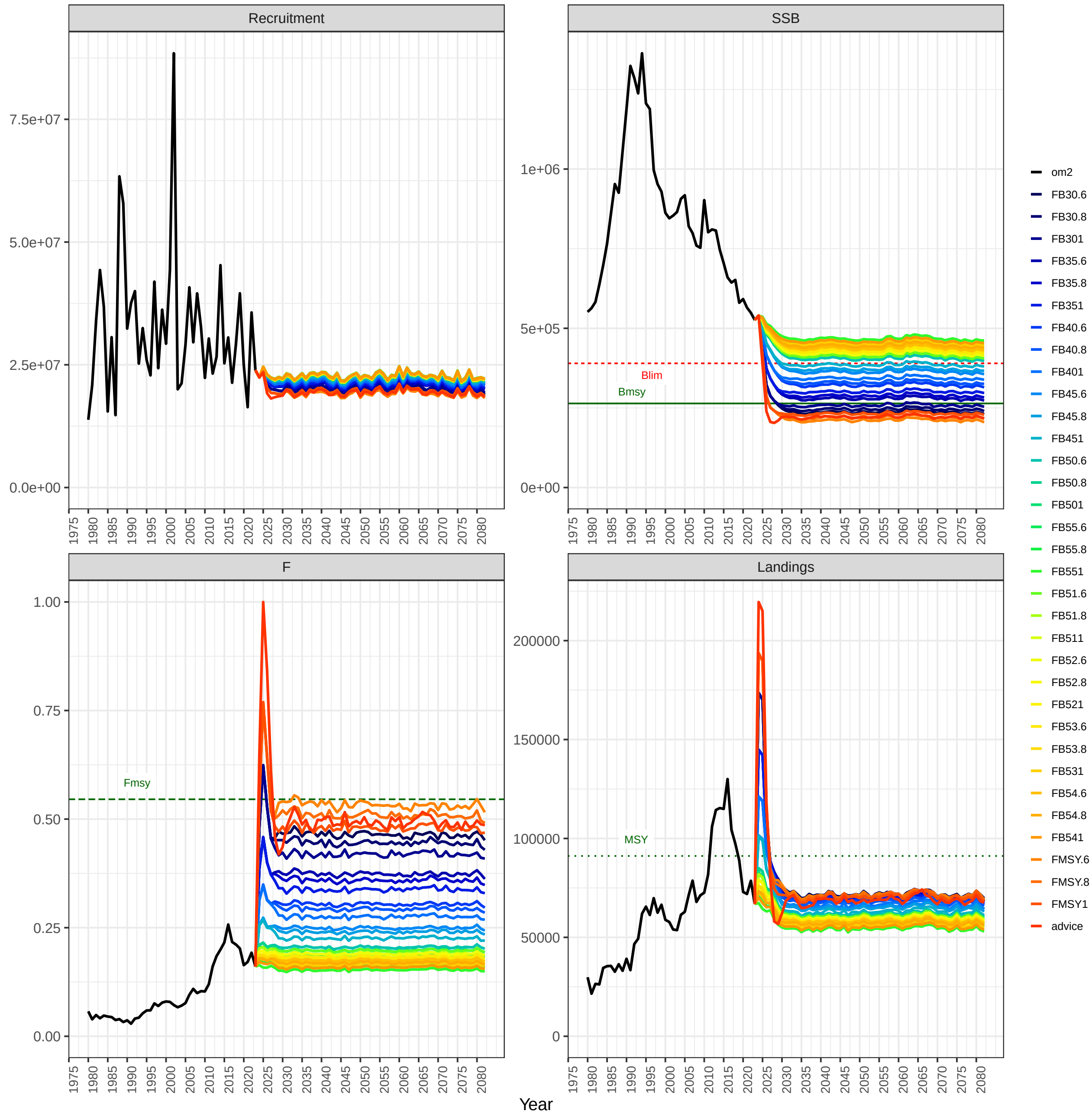

Figure 9: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for OM2 medians

```r
plotAdvice(res1) + scale_color_manual(values = c("black", sscol(length(stks1)))) +
    scale_fill_manual(values = c("black", sscol(length(stks1)))) +
    scale_x_continuous(breaks = seq(1960, fy, 5)) + theme(axis.text.x = element_text(size = 8,
    angle = 90)) + guides(col = guide_legend(ncol = 1))
```

```r
plotAdvice(res2) + scale_color_manual(values = c("black", sscol(length(stks2)))) +
    scale_fill_manual(values = c("black", sscol(length(stks2)))) +
    scale_x_continuous(breaks = seq(1960, fy, 5)) + theme(axis.text.x = element_text(size = 8,
    angle = 90)) + guides(col = guide_legend(ncol = 1))
```

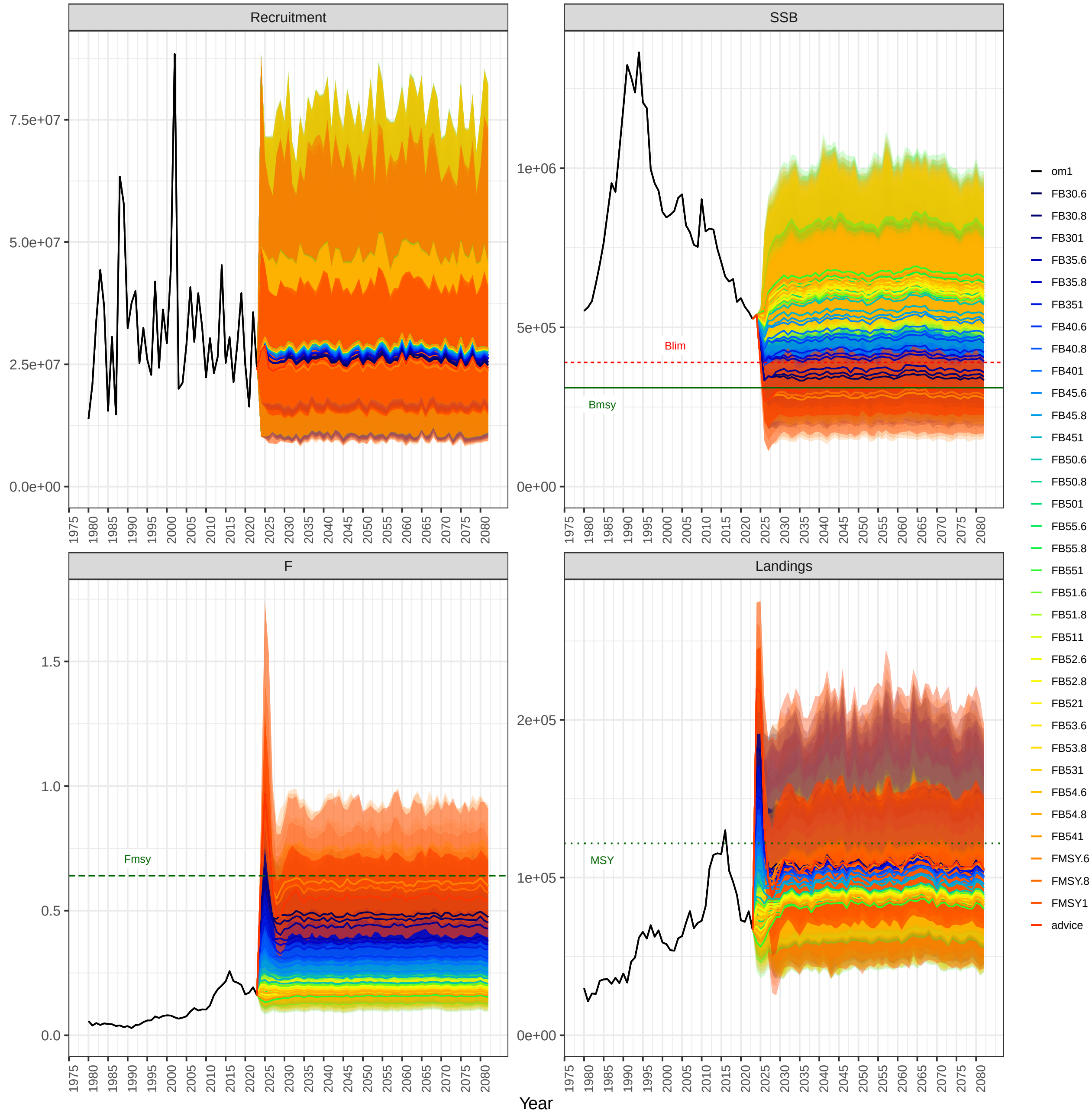

Figure 10: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for OM1, all

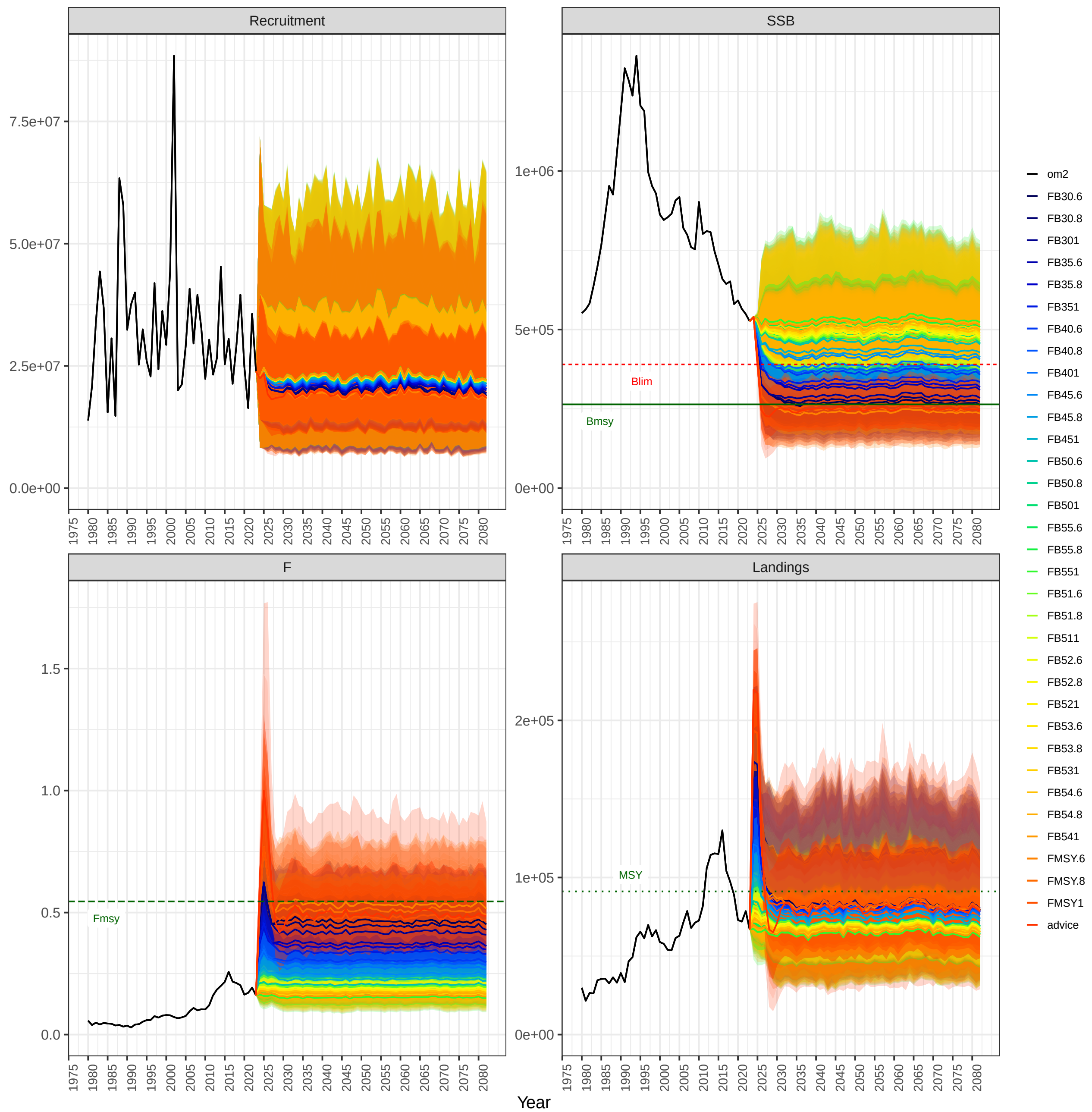

Figure 11: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for OM2, all

5.5 Select best HCR target and trigger combination

```
perfCsv <- read.csv("msePerf_table.csv")
OM <- c("REF", "RED")
OM <- c("REF")

p05 <- subset(perfCsv, om %in% OM & (name == "P3(B<Blim)" & Median <
    0.05))
catchP05 <- subset(perfCsv, om %in% OM & (name == "Catch/MSY" &
    mp %in% p05$mp))
bestCase <- subset(catchP05, Median == max(Median))
bestCase <- bestCase %>%
    select(mp, trigger, target)
bestCase$target <- round(bestCase$target, 3)
names(bestCase)[1] <- "MP"
names(bestCase)[2] <- "Btrigger"
names(bestCase)[3] <- "Ftarget"

knitr::kable(bestCase, "pipe", align = "lc", caption = "Summary of estimated reference
      points for reference case model of Gulf of Bothnian herring")
```

Table 1: Summary of estimated reference points for reference case model of Gulf of Bothnian herring

	MP	Btrigger	Ftarget
846	FB501	613355	0.218

6 Estimate Blim with different methods as the minimum SSB level that resulted in a recruitment higher that the median and fractions of B0

```
library(StockRecruitSET)
   Loading required package: TMB
   Loading required package: RcppEigen
   Loading required package: bbmle
   Loading required package: stats4

   Attaching package: 'bbmle'
   The following object is masked from 'package:dplyr':

       slice

flsr <- as.FLSR(stk_single)
S <- an(ssb(flsr))
R <- an(rec(flsr))

Bpaemp <- calcBlim(S, R, quant = 0.5, type = 1, nmin = 1, AIC = TRUE)
Blim_emp = round(Bpaemp/(exp(1.645 * SSBcv)), 0)
Blim_emp_defCV = round(Bpaemp/(exp(1.645 * 0.2)), 0)
Blim24 <- round(B0 * 0.24, 0)
Blim30 <- round(B0 * 0.3, 0)
```

```r
### Bloss calculations
Bloss = min(ssb(stk_single))
BpaBloss = Bloss
BlimBloss = BpaBloss/(exp(1.645 * SSBcv))


Bpaemp
   [1] 548671
Blim_emp
   [1] 414043
Blim_emp_defCV
   [1] 394847
Blim24
   [1] 294410
Blim30
   [1] 368013
Bloss
   [1] 516727.1
BpaBloss
   [1] 516727.1
BlimBloss
   [1] 389937.5
```

7 Calculate ABI MSY for the selected scenario

```r
library(FLCore)
stk = plans[["REF"]][["FB501"]]@om@stock
stk <- window(stk, start = 2025, end = 2082)
source("~/Max/Commitees/ICES/WGBFAS/2024/GBH/abi.R")
age = abiAge(brps$REF, ref = "msy", p = 0.9)
pmsy = abiMsy(brps$REF, ref = "msy", p = 0.9)
pt = abistock(stk, age)
abi_stk = abi(stk, brps$REF)
gg = plot(abi_stk)
gg = gg + labs(title = "", subtitle = "", x = "Year", y = "ABI MSY")
```

```r
load("~/Max/Commitees/ICES/WGBFAS/2024/GBH/plot.Rdata")
gg
```

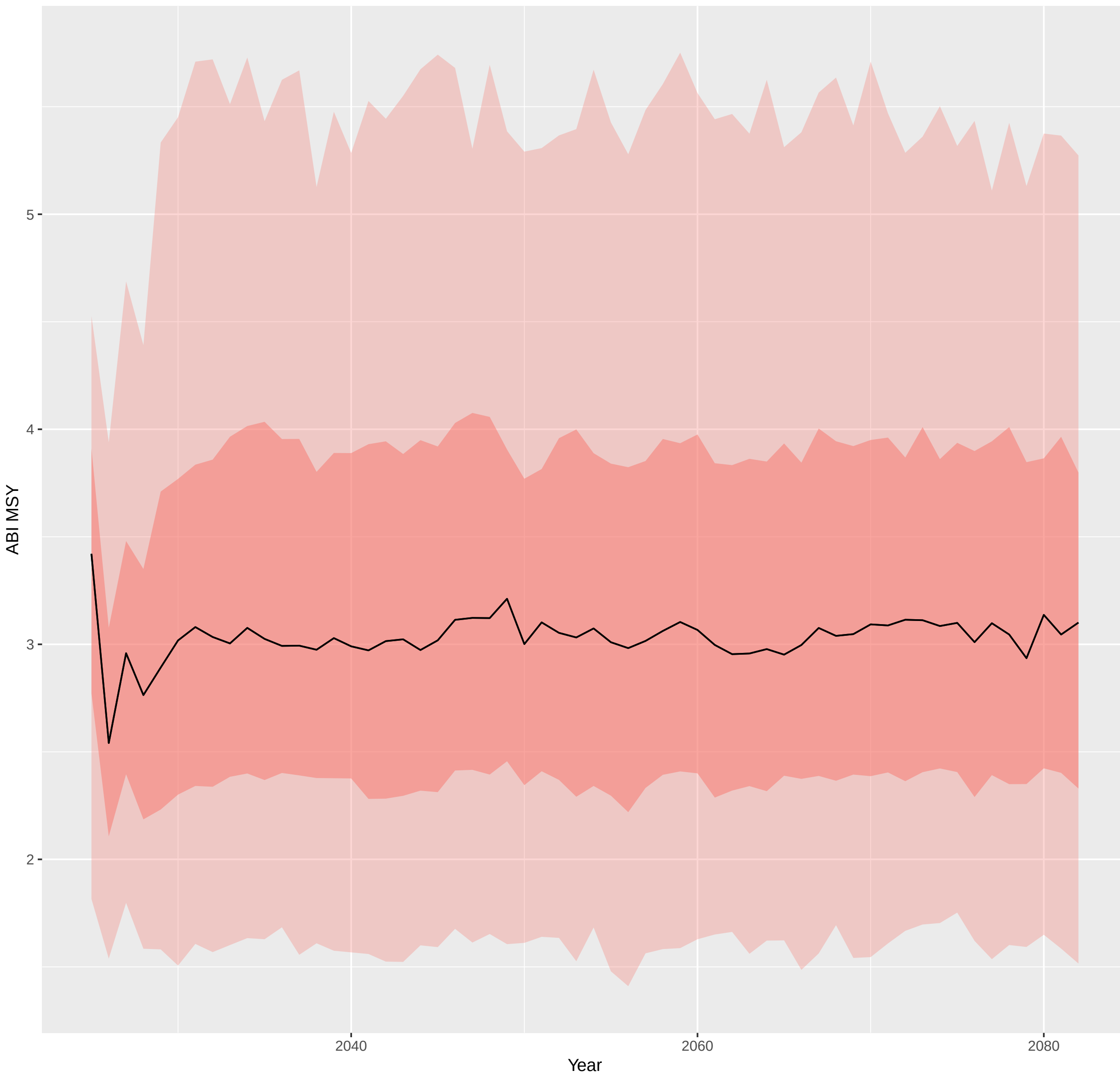

Figure 12: Herring (Clupea harengus) in subdivisions 30 and 31 (Gulf of Bothnia). Trajectories plot for ABI MSE for the selected scenario FB501; values over 1 indicates an age structure in line with BMSY