Management Strategy Evaluation (MSE) Example for Black Sea Bass

Chengxue Li

August 06, 2025

Contents

Introduction

This document provides a comprehensive example of how to conduct a Management Strategy Evaluation (MSE) for Black Sea Bass using the `wham` and `whamMSE` R packages. This workflow is a complete example, covering:

1. **Operating Model (OM) Setup**: Configuring an OM conditioned on historical data for Black Sea Bass.
2. **Estimation Model (EM) Configuration**: Defining the assessment model that will be used within the MSE loop.
3. **MSE Execution**: Running the simulation to evaluate the performance of the management strategy.
4. **Result Visualization**: Summarizing and plotting the output from the MSE.

1. Installation and Package Loading

First, we need to install and load the necessary R packages. The `wham` package provides the underlying state-space assessment framework, and `whamMSE` provides the functions to perform the management strategy evaluation.

Package Dependency

The `whamMSE` package is an extension of `wham` and relies on its core functions and objects. This dependency ensures that updates to the core modeling framework in `wham` are seamlessly integrated, maintaining consistency and preventing version conflicts.

Installation and Loading

The following commands are for installing the packages. These steps are typically only done once or when updating. Note that the file paths are user-specific and must be adjusted.

```
# Install wham from GitHub
# devtools::install_github("timjmiller/wham@lab", dependencies=FALSE)
```

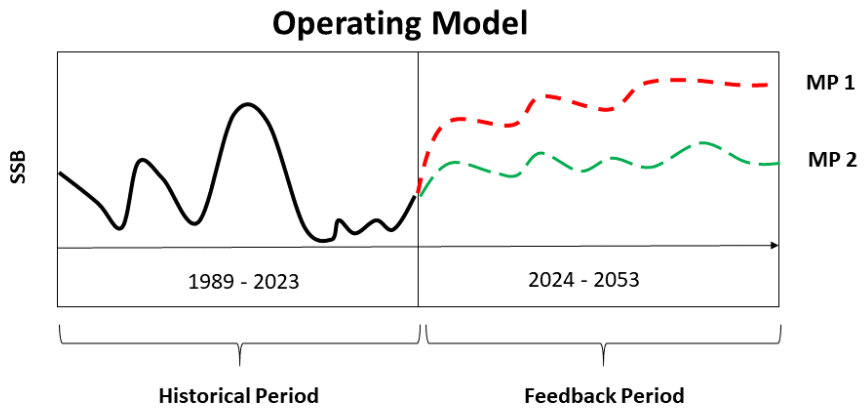


Figure 1: WHAT IS MSE?

```
# Install whamMSE from a local directory (example path)
# devtools::document("C:/Users/chengxue.li/whamMSE")
# devtools::load_all("C:/Users/chengxue.li/whamMSE")
# devtools::install("C:/Users/chengxue.li/whamMSE")
```

Now, we load the packages and set the working directory.

```
# Load required libraries
library(wham)
library(whamMSE)

# Set the working directory
# IMPORTANT: Update this path to your project folder where data files are located.
# setwd("C:/Users/chengxue.li/Desktop/Rutgers-MSE-Setup")
```

2. Defining Model Dimensions and Periods

We begin by defining the core dimensions of our simulation, including the time periods, number of stocks, regions, fleets, and other biological parameters.

```
# Define the historical period for the model
years <- 1989:2023

# Define the MSE feedback (projection) period in years
MSE_years <- 30

# Define the number of seasons and the fraction of the year in each 'season'
n_seasons <- 11
fracyr_seasons <- c(rep(1, 5), 2, rep(1, 5)) / 12
```

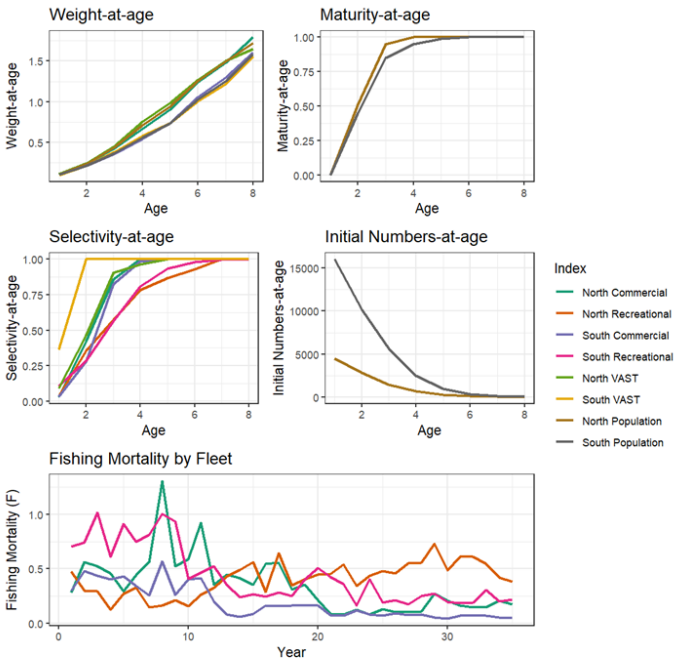


Figure 2: Heterogeneity in biology and fishery for Black Sea Bass.

```
# Define the number of 'stocks' (populations or subpopulations)
n_stocks <- 2

# Define the number of geographic regions
n_regions <- 2

# Define the total number of fishing fleets across all regions
n_fleets <- 4

# Define the total number of surveys across all regions
n_indices <- 2

# Define the number of age classes
n_ages <- 8

# Define the fraction of the year when the population spawns
fracyr_spawn <- 0.5
```

3. Preparing Input Data

This section loads or defines the biological and fishery data used to parameterize the Operating Model.

Weight-at-Age (WAA)

We prepare the weight-at-age data. In this example, we assume time-invariant WAA by replicating a base WAA matrix across all simulation years.

```r
# Ensure "WAA.RDS" is in your working directory
waa <- readRDS("WAA.RDS")
user_waa <- list()
# Dimension: indices x years x ages
user_waa$waa <- array(NA, dim = c(8, length(years) + MSE_years, 8))
for (age in 1:8) {
  for (indx in 1:8) {
    user_waa$waa[indx, , age] <- rep(waa[indx, age], length(years) + MSE_years)
  }
}
# Pointers to assign the correct WAA matrix to different model components
user_waa$waa_pointer_fleets <- 1:4
user_waa$waa_pointer_indices <- 5:6
user_waa$waa_pointer_ssb <- 7:8
user_waa$waa_pointer_M <- 7:8
```

Maturity-at-Age (MAA)

Next, we define time-invariant maturity-at-age for the two stocks.

```r
user_maa <- rbind(
  c(0, 0.51, 0.95, 1, 1, 1, 1, 1),
  c(0, 0.45, 0.85, 0.95, 0.99, 1, 1, 1)
)
```

Fleet and Survey Information

We define which fleets and surveys operate in which regions and specify their associated observation errors.

```r
# Fleet Information
fleet_regions <- c(1, 1, 2, 2)
catch_Neff <- c(50, 50, 50, 50) # Effective sample size for age comps
catch_cv <- c(0.05, 0.15, 0.05, 0.15) # CV for total catch

# Survey Information
index_regions <- c(1, 2)
index_Neff <- c(50, 50) # Effective sample size for age comps
index_cv <- c(0.4, 0.4) # CV for survey index
# Survey catchability (q), using estimates from the recent BSB stock assessment
q <- c(7.370412e-05, 2.832577e-05)
# Fraction of the year when each survey is conducted
fracyr_indices <- c(0.25, 0.25)
```

4. Configuring the Operating Model (OM)

This section builds the components of the Operating Model, which represents the "true" underlying population and fishery dynamics.

Generate Basic Information List

The `generate_basic_info` function consolidates the parameters into a list required for building the OM.

```r
info <- generate_basic_info(
  base.years = years,
  n_feedback_years = MSE_years,
```

```
  n_stocks = n_stocks,
  n_regions = n_regions,
  n_indices = n_indices,
  n_fleets = n_fleets,
  n_ages = n_ages,
  n_seasons = n_seasons,
  catch_info = list(catch_cv = catch_cv, catch_Neff = catch_Neff),
  index_info = list(index_cv = index_cv, index_Neff = index_Neff, q = q, fracyr_indices = fracyr_indices
  fracyr_seasons = fracyr_seasons,
  fracyr_spawn = 0.5,
  user_waa = user_waa,
  user_maturity = user_maa
)
```

```
##
## Survey fraction for index 1 is equal to 0, indicating the survey is happening at the edge of a season
##
## Survey fraction for index 2 is equal to 0, indicating the survey is happening at the edge of a season
```

```
# Extract the generated lists for easier access
basic_info <- info$basic_info
catch_info <- info$catch_info
index_info <- info$index_info
F_info <- info$F
```

Update Historical Fishing Mortality (F)

We replace the default historical fishing mortality rates with estimates from the BSB stock assessment.

```
# Ensure "F_hist.RDS" is in your working directory
F_est <- readRDS("F_hist.RDS")
F_info$F[1:35, ] <- F_est
```

Configure Movement Model

This section defines the movement dynamics between regions. The settings below define where stocks can be, when they can move, and the probability of moving.

The conceptual movement pattern is shown in the figure below.

```
# Specify which ages can be in which region on January 1 of each year.
basic_info$NAA_where <- array(1, dim = c(n_stocks, n_regions, n_ages))
basic_info$NAA_where[1, 2, 1] <- 0 # Stock 1, age 1 cannot be in region 2 on Jan 1
basic_info$NAA_where[2, 1, ] <- 0  # Stock 2 does not move into region 1 at any age

# Define movement parameters
move <- list(stock_move = c(TRUE, FALSE), separable = TRUE)
move$must_move <- array(0, dim = c(2, 11, 2))
move$must_move[1, 5, 2] <- 1
move$can_move <- array(0, dim = c(2, 11, 2, 2))
move$can_move[1, 1:4, 2, 1] <- 1
move$can_move[1, 7:11, 1, 2] <- 1
move$can_move[1, 5, 2, ] <- 1
mean_vals <- array(0, dim = c(2, length(fracyr_seasons), 2, 1))
mean_vals[1, 1:11, 1, 1] <- 0.02214863
mean_vals[1, 1:11, 2, 1] <- 0.3130358
```

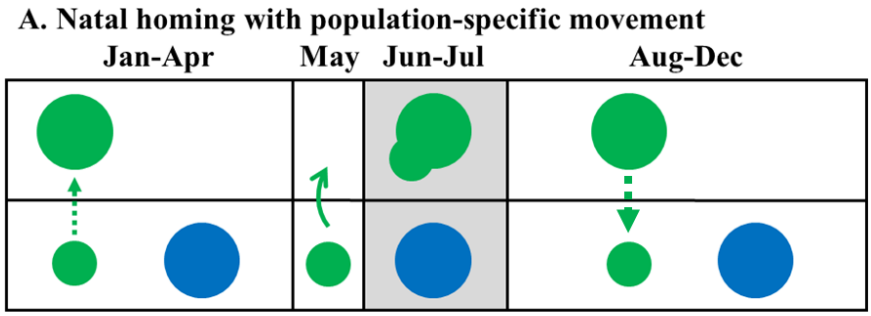


Figure 3: Movement Pattern for Black Sea Bass.

```
move$mean_vals <- mean_vals
move$mean_model <- matrix("stock_constant", 2, 1)
```

Configure Numbers-at-Age (NAA) Dynamics

We define process errors for recruitment and survival. For this example, we assume recruitment is random around a mean (density-independent).

```
sigma <- "rec+1"
re_cor <- "iid"
ini.opt <- "age-specific-fe"
sigma_vals <- array(0.2, dim = c(n_stocks, n_regions, n_ages)) # NAA survival sigma
sigma_vals[, , 1] <- 0.75 # Recruitment sigma

NAA_re <- list(
  N1_model = rep(ini.opt, n_stocks),
  sigma = rep(sigma, n_stocks),
  cor = rep(re_cor, n_stocks),
  recruit_model = 2, # 2 = density-independent
  recruit_pars = list(16000, 21000), # Mean recruitment for each stock
  sigma_vals = sigma_vals
)
```

Configure Natural Mortality (M)

We assume a time-invariant, age-constant natural mortality (M).

```
M <- list(model = "constant", initial_means = array(0.4, dim = c(n_stocks, n_regions, n_ages)))
```

Configure Gear Selectivity

We specify different selectivity patterns for each fleet and survey.

```
model <- c("age-specific", "age-specific", "logistic", "logistic", "age-specific", "age-specific")
initial_pars_f1 <- c(0.031, 0.421, 0.858, 1, 1, 1, 1, 1)
initial_pars_f2 <- c(0.036, 0.355, 0.573, 0.782, 0.868, 0.928, 1, 1)
initial_pars_f3 <- c(2.381, 0.402)
initial_pars_f4 <- c(2.788, 0.849)
initial_pars_s1 <- c(0.091, 0.463, 0.905, 0.962, 1, 1, 1, 1)
initial_pars_s2 <- c(0.362, 1, 1, 1, 1, 1, 1, 1)

sel <- list(
  model = model,
  initial_pars = c(
    list(initial_pars_f1), list(initial_pars_f2),
    list(initial_pars_f3), list(initial_pars_f4),
    list(initial_pars_s1), list(initial_pars_s2)
  ),
  fix_pars = list(4:8, 7:8, NULL, NULL, 5:8, 2:8)
)
```

5. Build the OM Object & Prepare for MSE

This section finalizes the OM setup by consolidating all components into a single `wham` input object.

Prepare Final `wham` Input

This critical step gathers all configurations into a single input list.

```
input <- prepare_wham_input(
  basic_info = basic_info, NAA_re = NAA_re,
  move = move,
  M = M,
  selectivity = sel,
  catch_info = catch_info,
  index_info = index_info,
  F = F_info
)
```

Post-processing Updates

Some inputs are easier to update after the main list has been created.

```
# Update WAA using the previously defined structure
waa_info <- info$par_inputs$user_waa
input <- update_waa(input, waa_info = waa_info)

# Set initial numbers-at-age for the first year of the historical period.
ini.NAA <- matrix(NA, n_ages, n_stocks)
ini.NAA[, 1] <- c(4531.15069, 2876.48828, 1484.19315, 692.67290, 314.73335, 142.50149, 64.41915, 52.581
ini.NAA[, 2] <- c(16023.94518, 10167.14058, 5683.71704, 2523.02265, 972.93721, 353.59712, 125.80803, 68
input$par$log_N1[] <- 0
for (i in 1:n_regions) {
  input$par$log_N1[i, i, ] <- log(ini.NAA[, i])
}
```

Generate the Operating Model Object

We use `fit_wham` with `do.fit = FALSE` to create the OM structure without fitting it to data. This defines our "true" simulated world.

```
random <- input$random # Identify which processes are random effects
input$random <- NULL   # Nullify so inner optimization won't change simulated REs
om <- fit_wham(input, do.fit = FALSE, do.brps = FALSE, MakeADFun.silent = TRUE)
```

6. MSE Configuration and Execution

This section sets up the specifics of the MSE feedback loop, including assessment frequency and the harvest control rule, and then runs the simulation.

Specify MSE Timeline

We define the frequency of assessments and determine which years in the projection period will be assessment years.

```
assess.interval <- 3 # Assessments occur every 3 years
base.years <- years
terminal.year <- tail(base.years, 1)
assess.years <- seq(terminal.year, tail(om$years, 1) - assess.interval, by = assess.interval)

# Print the designated assessment years to verify
print(assess.years)
```

```
##  [1] 2023 2026 2029 2032 2035 2038 2041 2044 2047 2050
```

Run the MSE Simulation Loop

WARNING: The following MSE loop can be computationally intensive and take a long time to run. The chunk option is set to `eval=FALSE` to prevent it from running automatically when you knit the document. Change to `eval=TRUE` to run the simulation.

```
# Generate a single realization of the OM (with one set of random effects)
om_with_data <- update_om_fn(om, seed = 123, random = random)

# Configure the Estimation Model (EM) for use within the loop
move_em <- move
move_em$use_prior <- array(0, dim = c(n_stocks, n_seasons, n_regions, n_regions - 1))
move_em$use_prior[1, 1, , ] <- 1
move_em$prior_sigma <- array(0.2, dim = c(n_stocks, n_seasons, n_regions, n_regions - 1))

# Specify the Harvest Control Rule (HCR)
hcr <- list()
hcr$hcr.type <- 1
hcr$hcr.opts <- list(use_FXSPR = TRUE, percentFXSPR = 75)

# Execute the MSE loop for one realization
mod <- loop_through_fn(
  om = om_with_data,
  em_info = info,
  random = random,
  sel_em = sel,
  M_em = M,
  NAA_re_em = NAA_re,
```

```
  move_em = move_em,
  em.opt = list(
    separate.em = FALSE,
    separate.em.type = 3,
    do.move = TRUE,
    est.move = TRUE
  ),
  assess_years = assess.years,
  assess_interval = assess.interval,
  base_years = base.years,
  year.use = 30,
  add.years = TRUE,
  seed = 123,
  hcr = hcr,
  save.last.em = FALSE
)
```

7. Visualizing MSE Results

After the MSE is complete, the `plot_mse_output` function can be used to summarize and visualize the results. The code below is an example of how to call the function. It is set to `eval=FALSE` to prevent errors, as it requires a `mods` object from a completed MSE run.

The types of plots that can be generated are shown here:

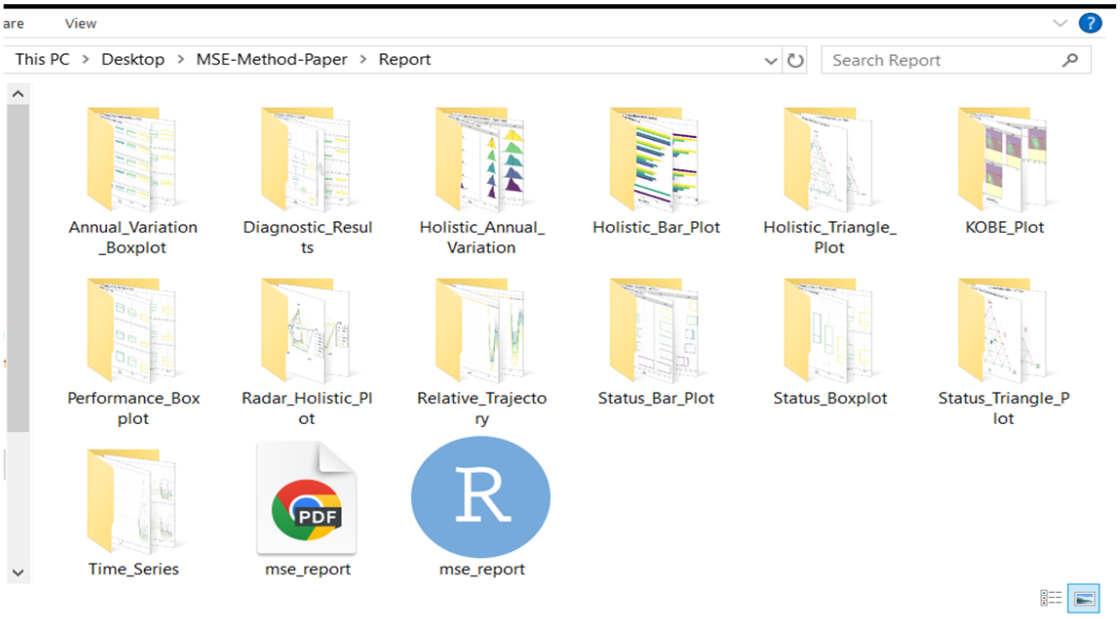


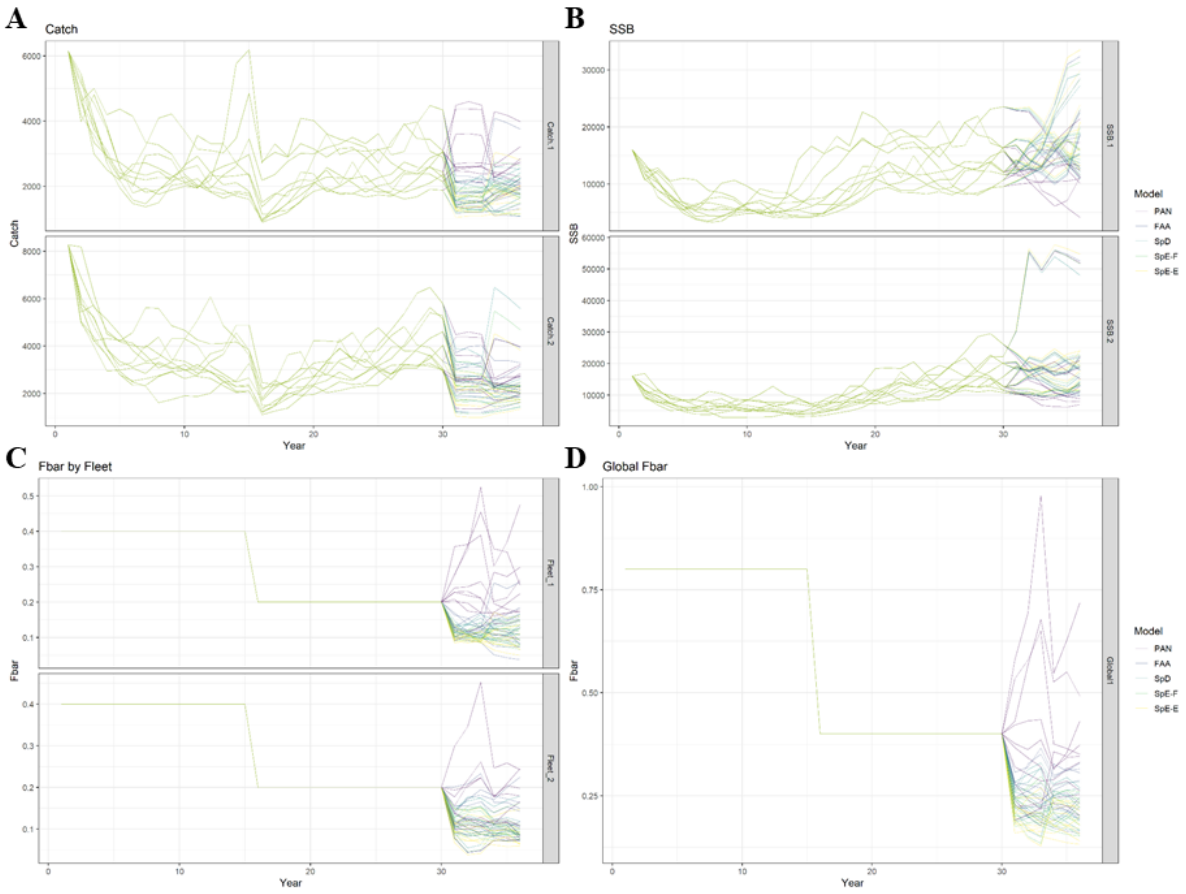
Figure 4: Different types of plots available from the plotting function.

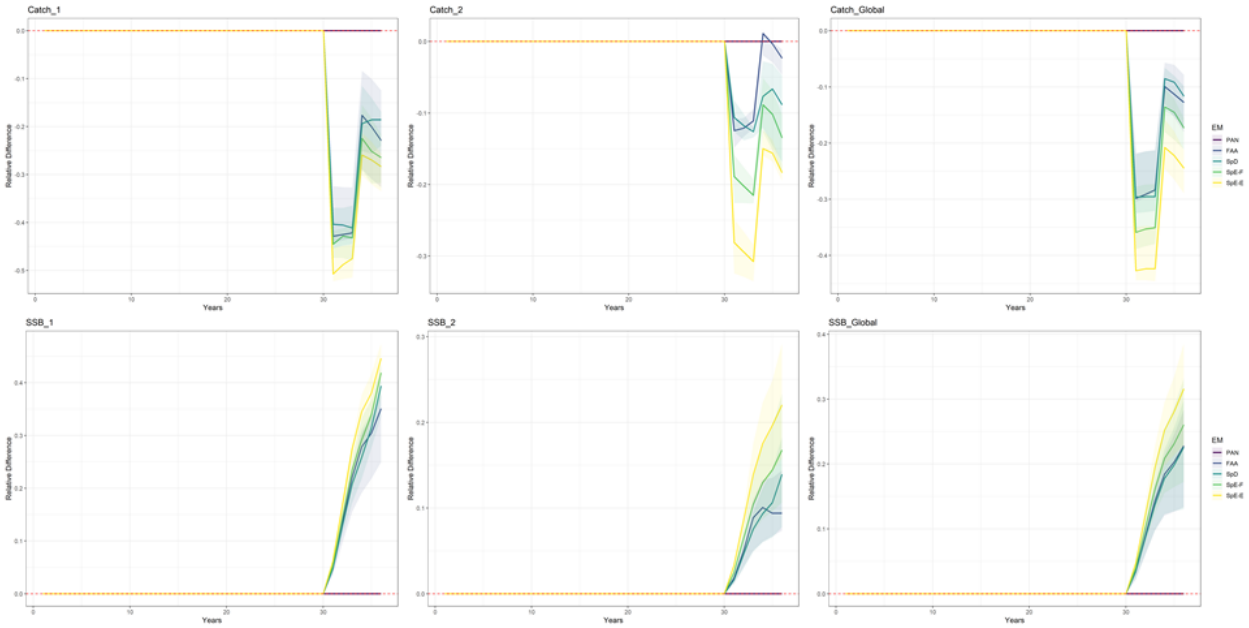
```
# This is an example call. `mods` would be a list of completed MSE runs.
plot_mse_output(
  mods,
  main_dir = getwd(),
```

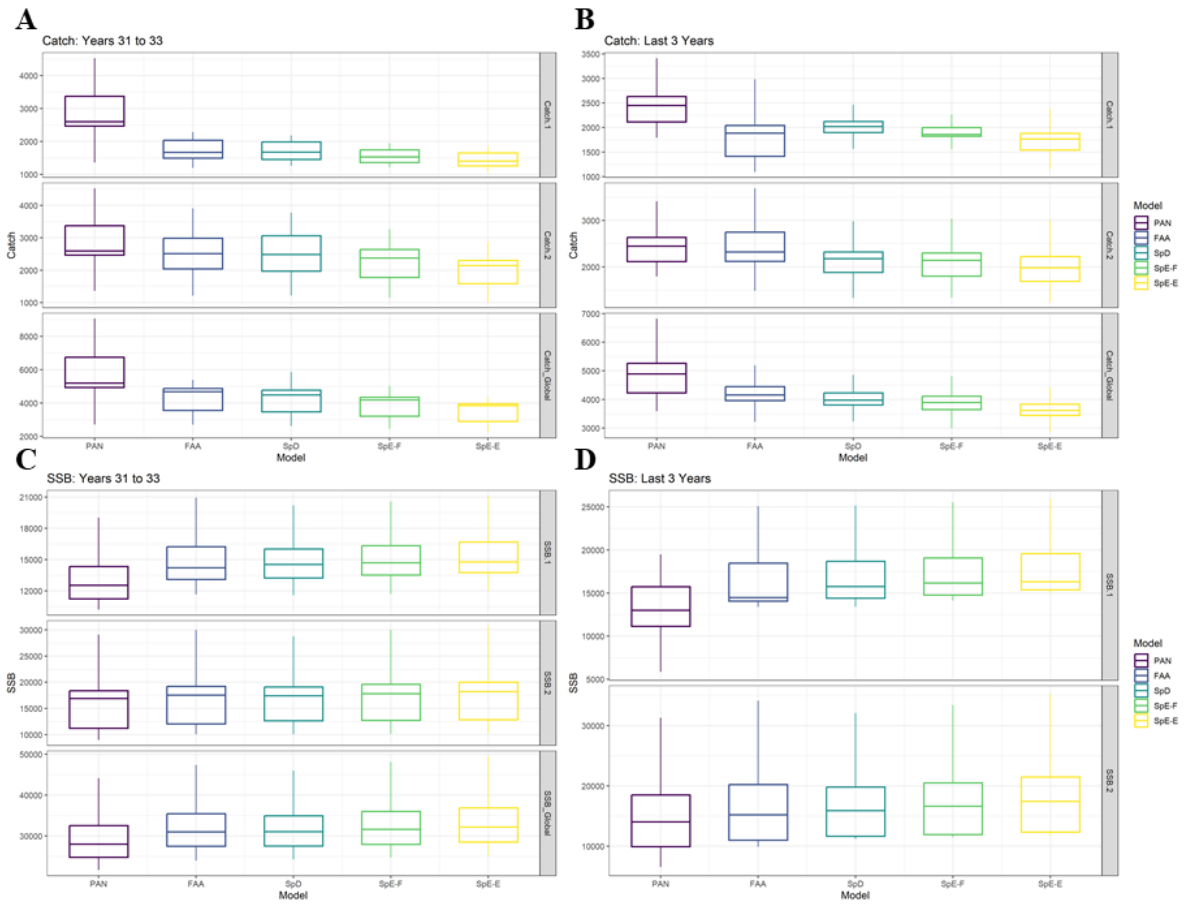
```
  output_dir = "Report",
  output_format = "pdf", # or "html" or "png"
  width = 10, height = 7, dpi = 300,
  col.opt = "D",
  method = "mean",
  outlier.opt = NA,
  f.ymax = 2, # control y-axis scale
  new_model_names = c('PAN', 'FAA', 'SpD', 'SpE-F', "SpE-E"),
  base.model = 'PAN',
  start.years = 31,
  use.n.years.first = 3,
  use.n.years.last = 3
)
```

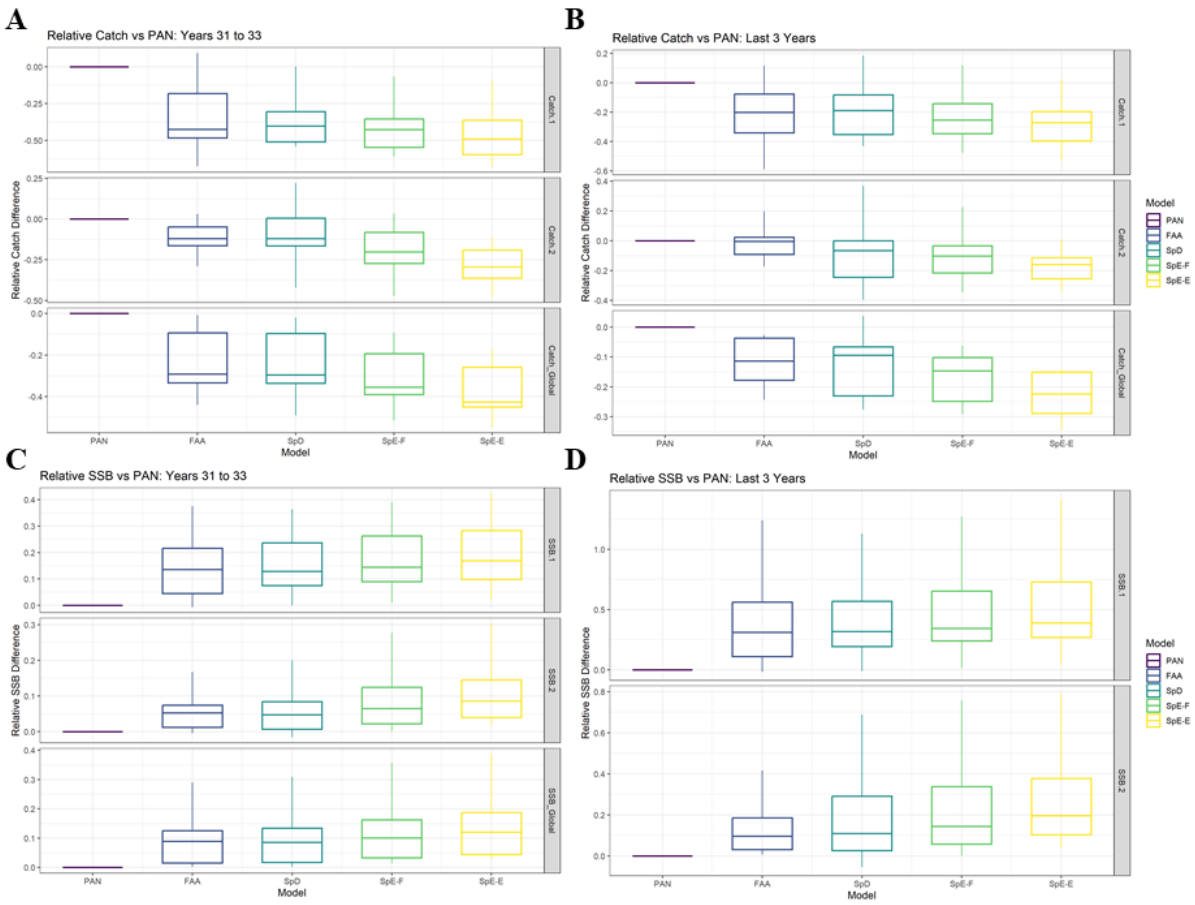
Example Output Figures

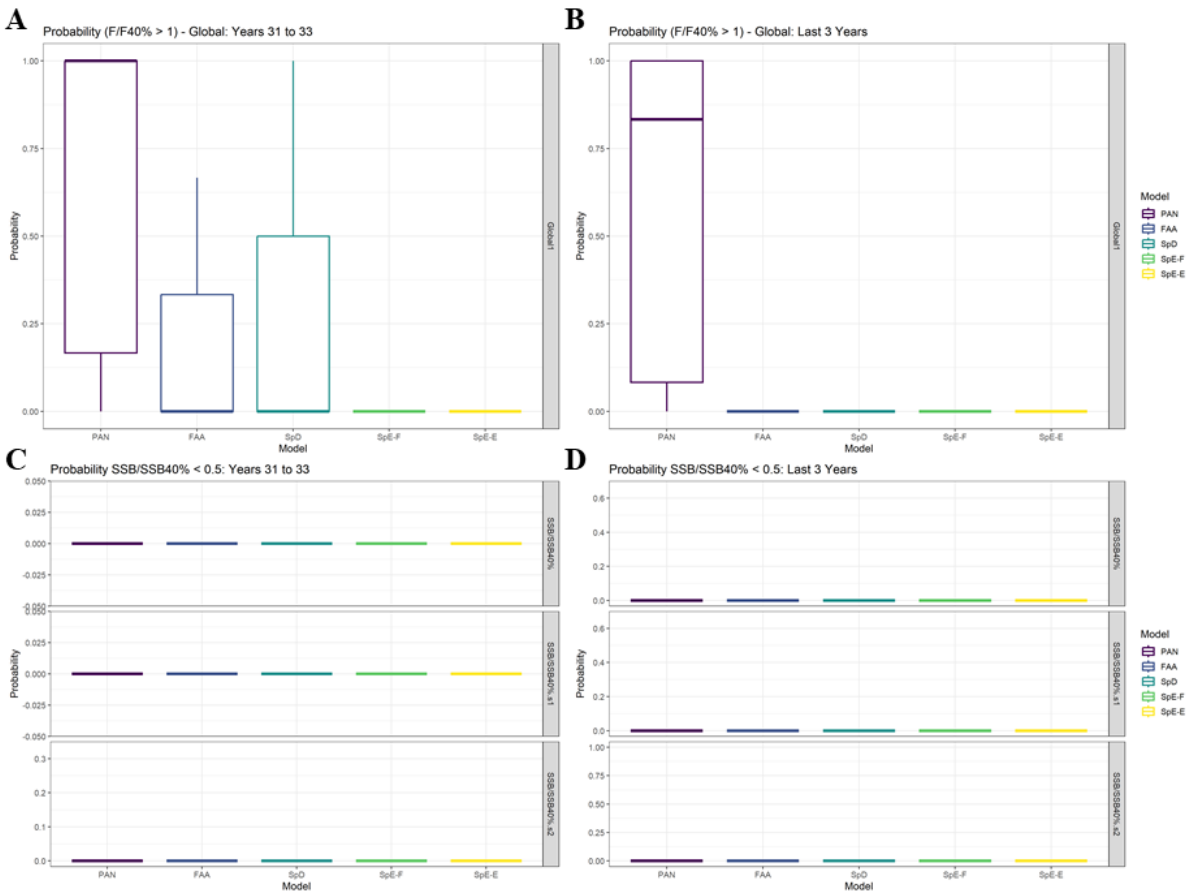
The following plots show the various types of output that can be generated by the `plot_mse_output` function. For this to work, ensure you have a subfolder named `Example_Figures` in your project directory containing `Slide1.PNG` through `Slide13.PNG`.

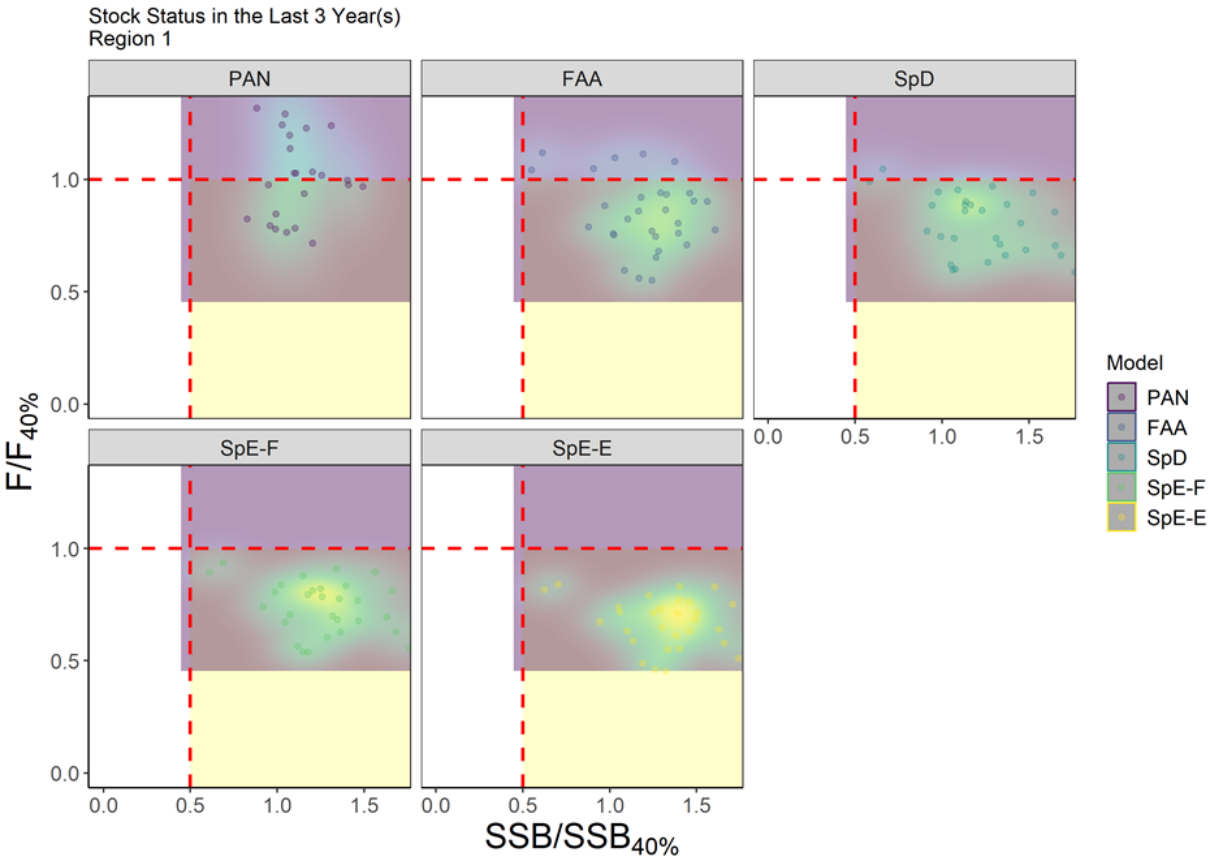


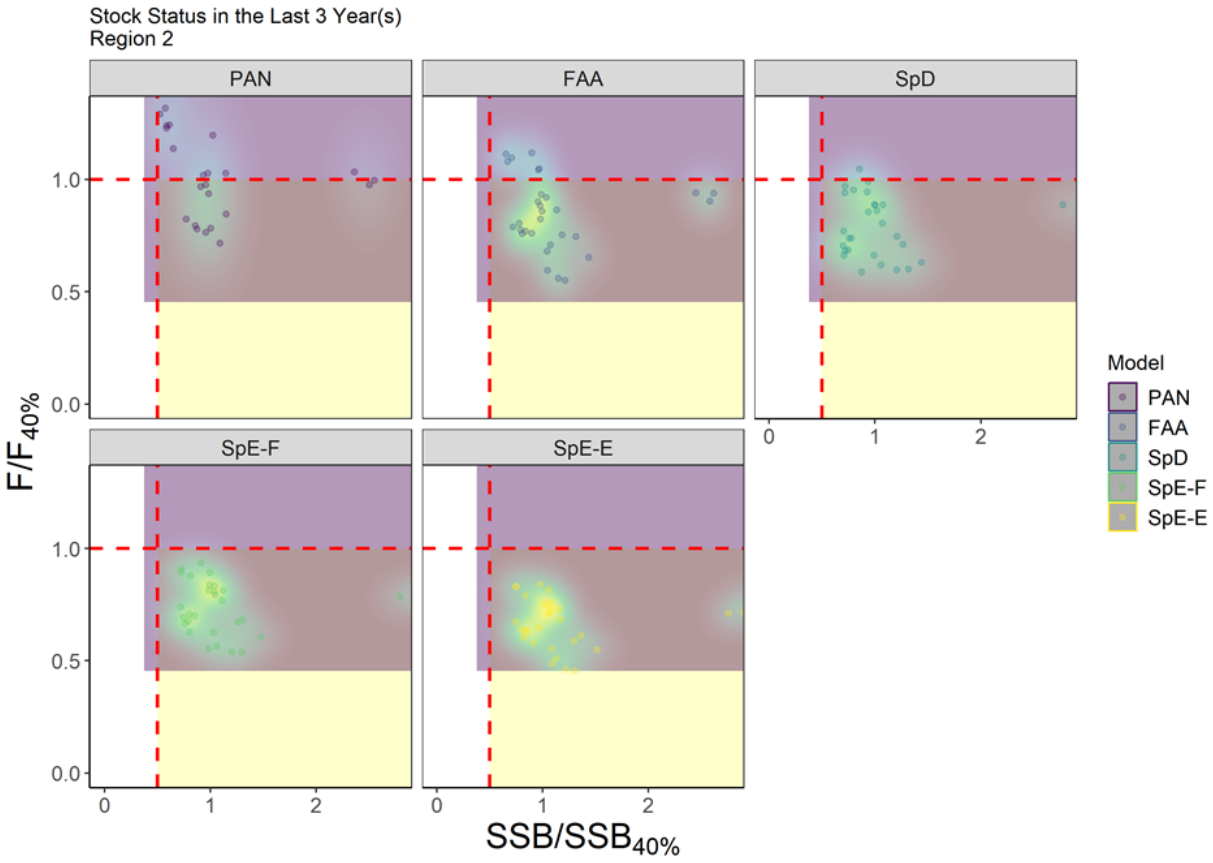

Catch_1
Catch_2
Catch_Global
SSB_1
SSB_2
SSB_Global
Relative Difference
Years
EM
PAN
FAA
SpD
SpE-F
SpE-E

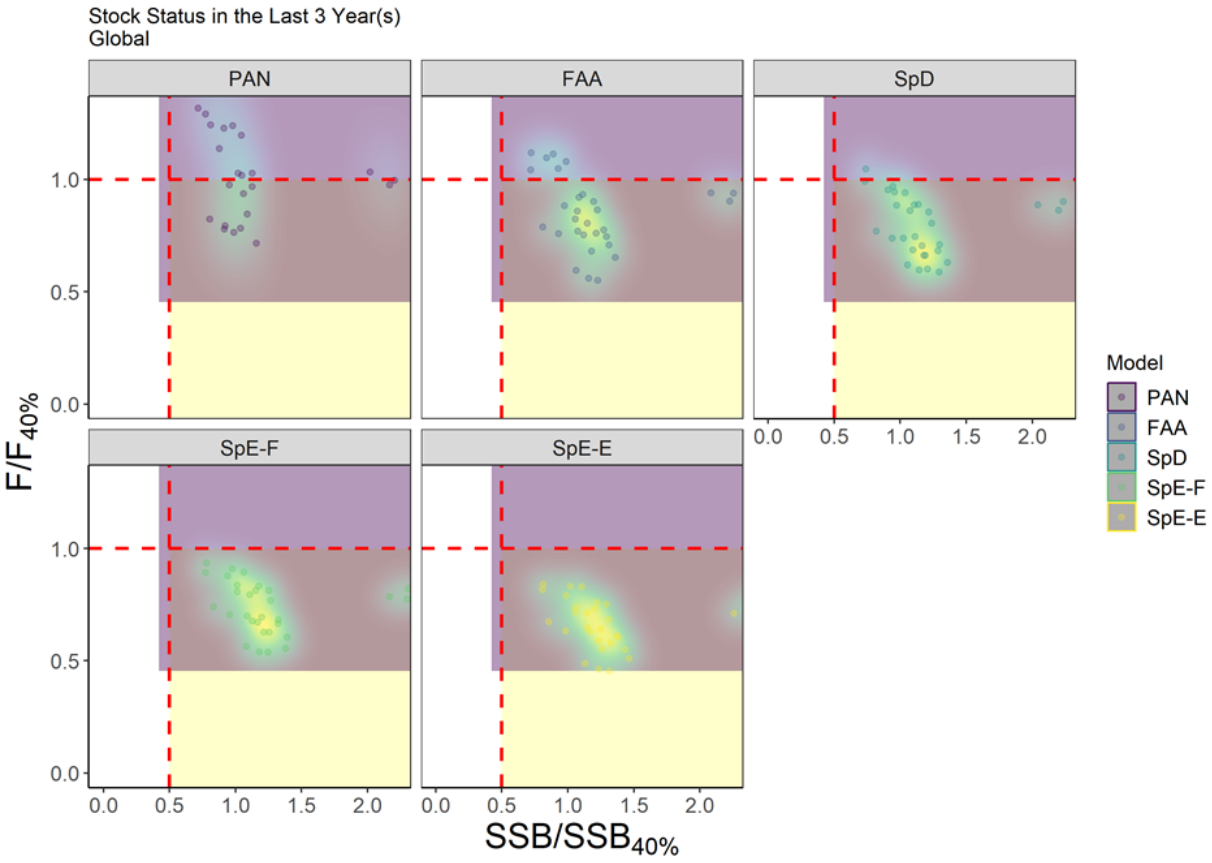

A
Catch: Years 31 to 33
B
Catch: Last 3 Years
C
SSB: Years 31 to 33
D
SSB: Last 3 Years
Catch
SSB
Model
PAN
FAA
SpD
SpE-F
SpE-E
Catch_1
Catch_2
Catch_Global
SSB_1
SSB_2
SSB_Global

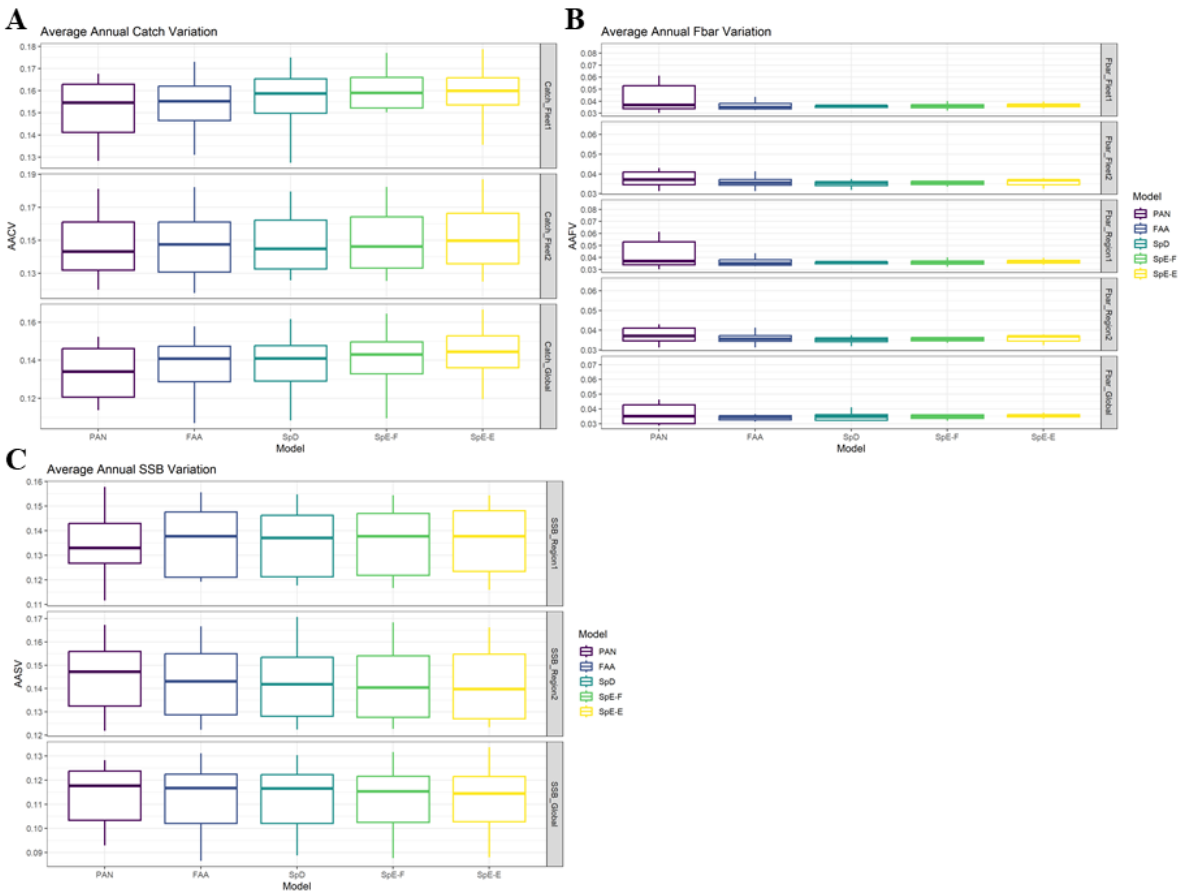

A
Relative Catch vs PAN: Years 31 to 33
Relative Catch Difference
Catch.1
Catch.2
Catch_Global
PAN
FAA
SpD
SpE-F
SpE-E
Model
B
Relative Catch vs PAN: Last 3 Years
Model
C
Relative SSB vs PAN: Years 31 to 33
Relative SSB Difference
SSB.1
SSB.2
SSB_Global
D
Relative SSB vs PAN: Last 3 Years

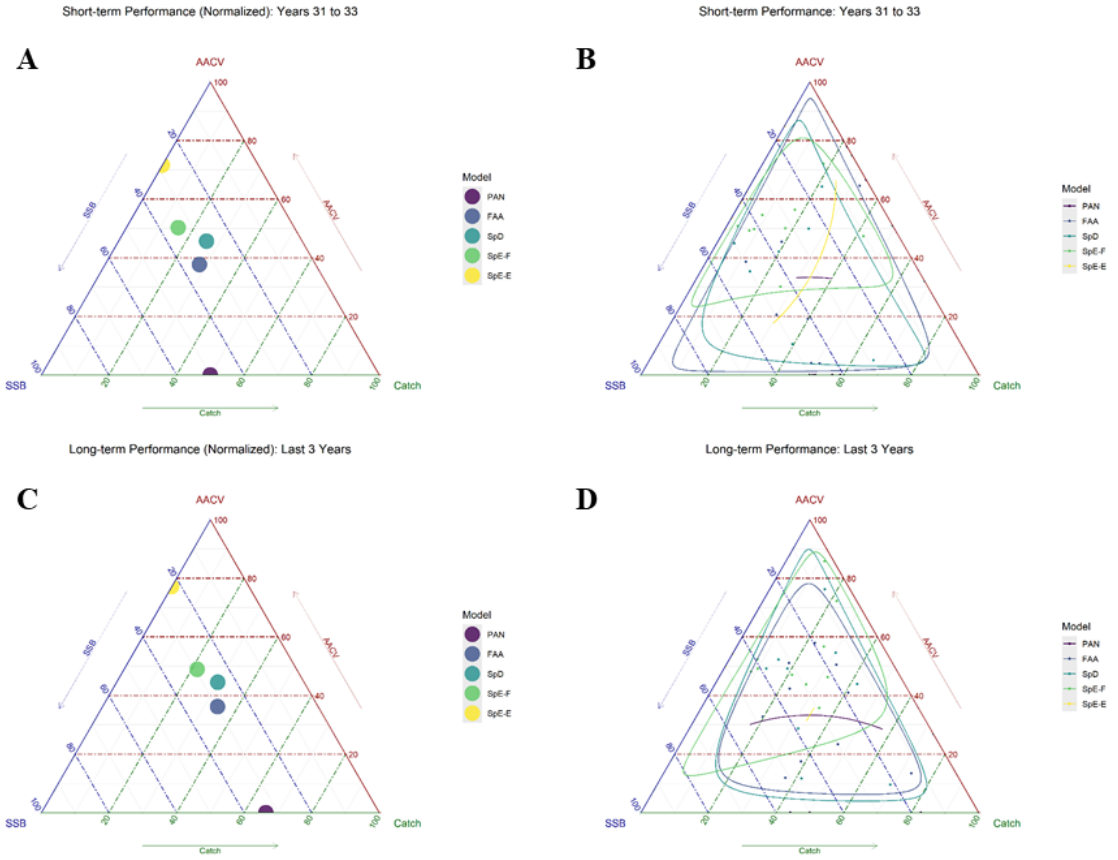

A
Probability (F/F40% > 1) - Global: Years 31 to 33
B
Probability (F/F40% > 1) - Global: Last 3 Years
C
Probability SSB/SSB40% < 0.5: Years 31 to 33
D
Probability SSB/SSB40% < 0.5: Last 3 Years
Probability
Model
PAN
FAA
SpD
SpE-F
SpE-E
Global1
SSB/SSB40%
SSB/SSB40%.x1
SSB/SSB40%.x2

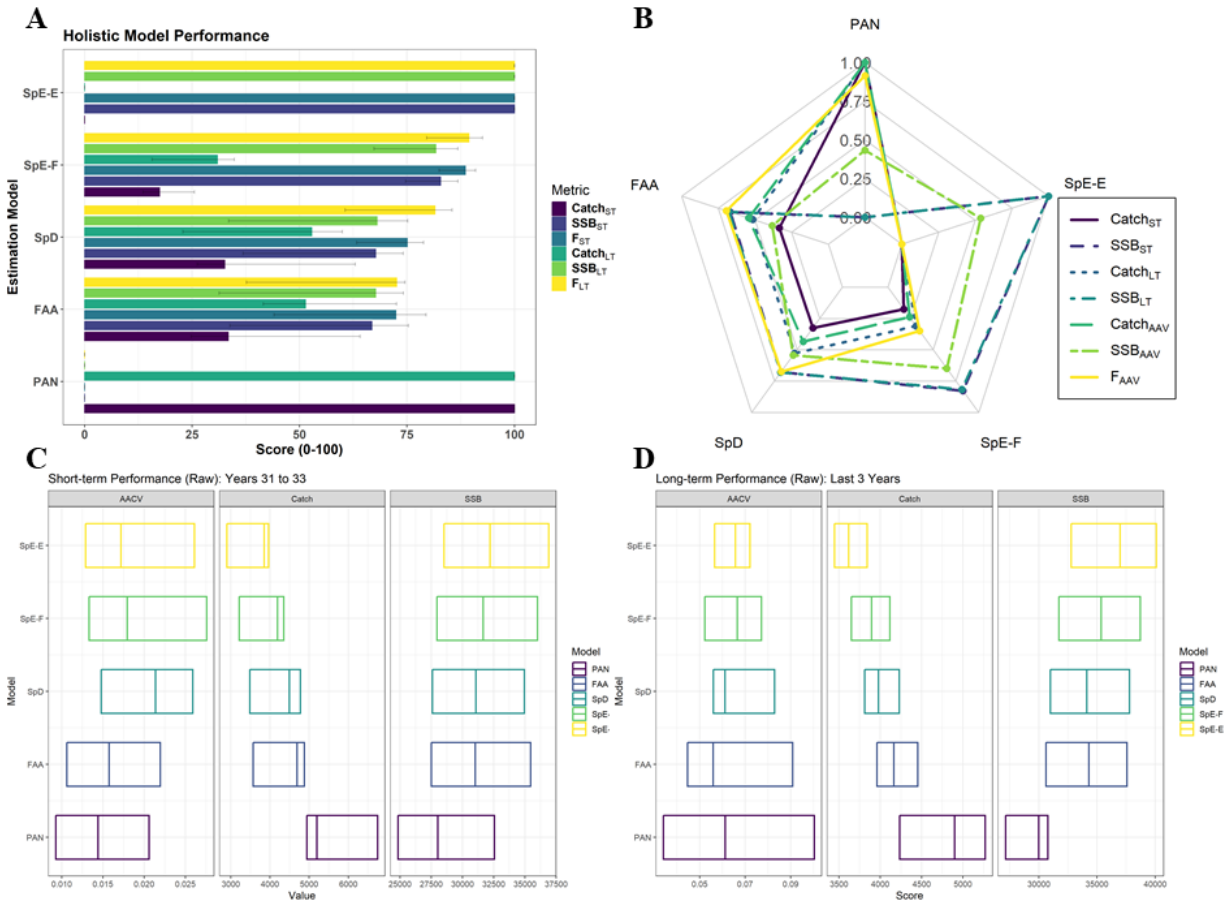

Stock Status in the Last 3 Year(s)
Region 1
PAN
FAA
SpD
SpE-F
SpE-E
F/F40%
SSB/SSB40%
0.0
0.5
1.0
1.5
Model
PAN
FAA
SpD
SpE-F
SpE-E

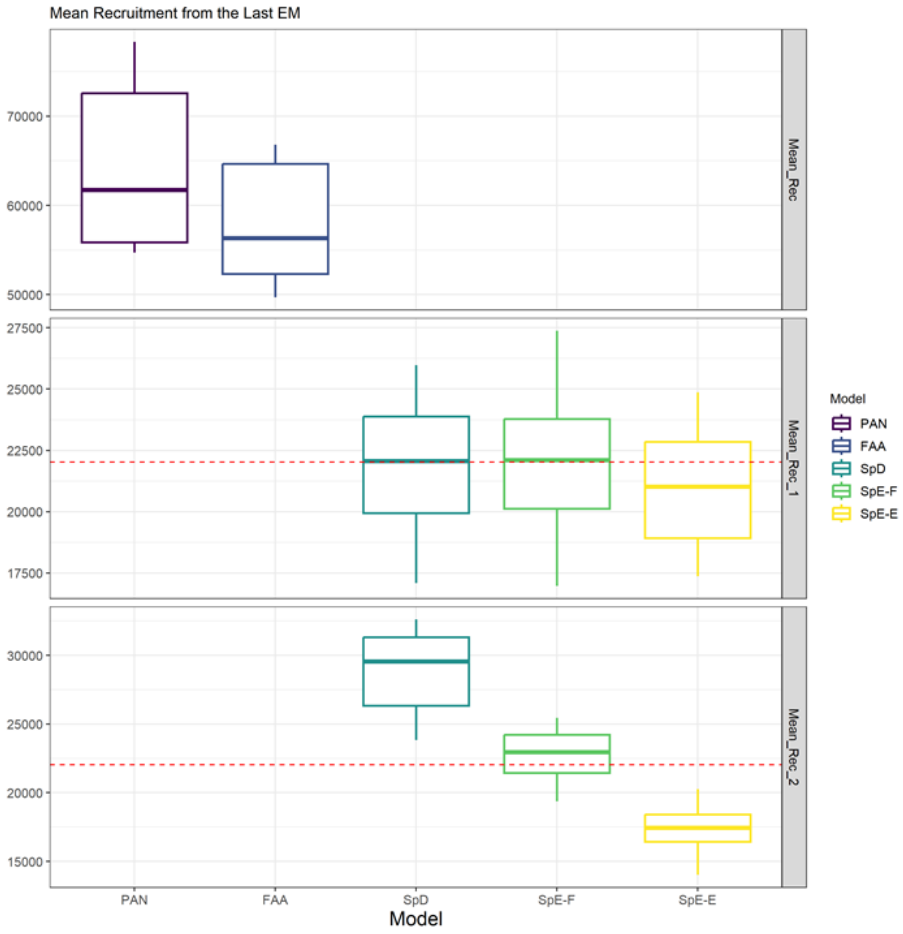

Stock Status in the Last 3 Year(s)
Region 2
PAN
FAA
SpD
SpE-F
SpE-E
1.0
0.5
0.0
0
1
2
F/F40%
SSB/SSB40%
Model
PAN
FAA
SpD
SpE-F
SpE-E

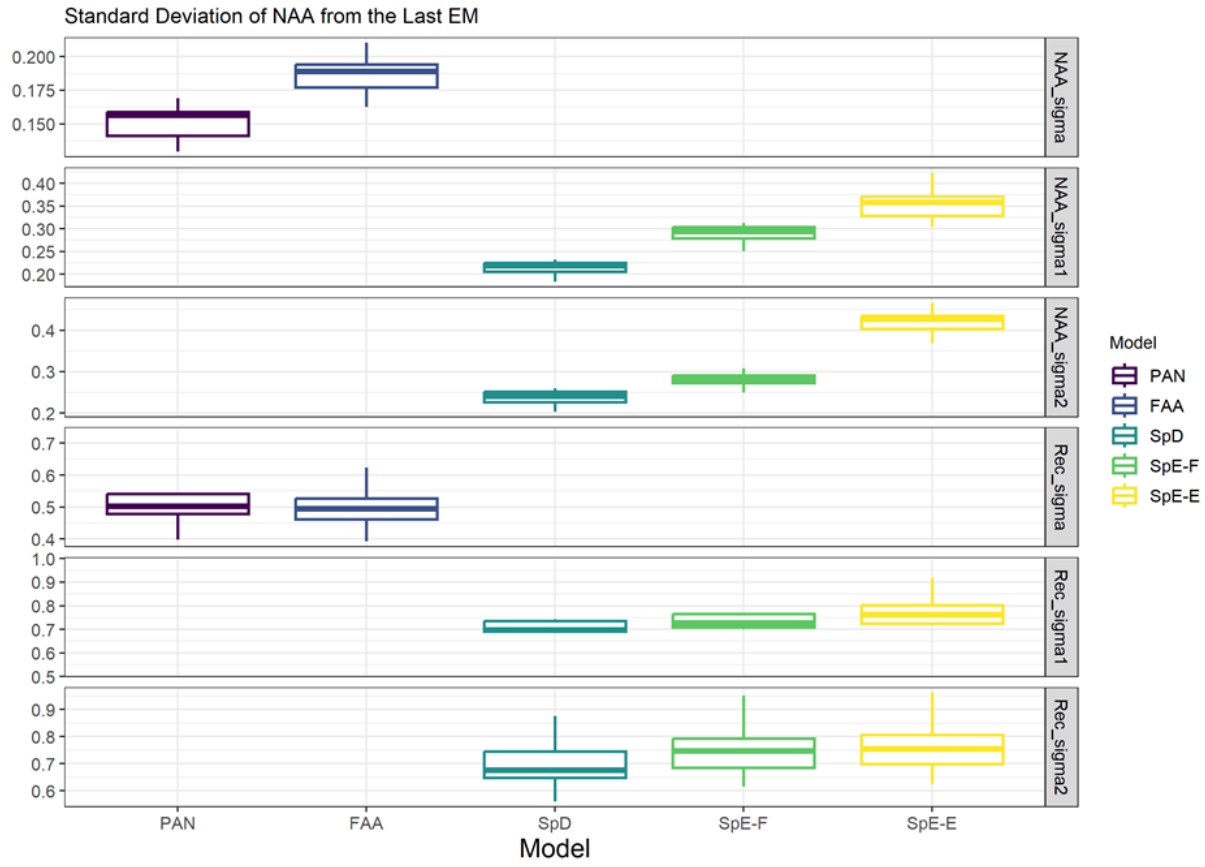

Stock Status in the Last 3 Year(s)
Global
PAN
FAA
SpD
SpE-F
SpE-E
F/F40%
SSB/SSB40%
0.0
0.5
1.0
1.5
2.0
Model
PAN
FAA
SpD
SpE-F
SpE-E


A
Average Annual Catch Variation
AACV
Catch_Fleet1
Catch_Fleet2
Catch_Global
PAN
FAA
SpD
SpE-F
SpE-E
Model
B
Average Annual Fbar Variation
AAFV
Fbar_Fleet1
Fbar_Fleet2
Fbar_Region1
Fbar_Region2
Fbar_Global
Model
PAN
FAA
SpD
SpE-F
SpE-E
C
Average Annual SSB Variation
AASV
SSB_Region1
SSB_Region2
SSB_Global
Model
PAN
FAA
SpD
SpE-F
SpE-E


Short-term Performance (Normalized): Years 31 to 33
A
AACV
SSB
Catch
Model
PAN
FAA
SpD
SpE-F
SpE-E
Short-term Performance: Years 31 to 33
B
Long-term Performance (Normalized): Last 3 Years
C
Long-term Performance: Last 3 Years
D


A
Holistic Model Performance
SpE-E
SpE-F
SpD
FAA
PAN
Estimation Model
0
25
50
75
100
Score (0-100)
Metric
Catch_ST
SSB_ST
F_ST
Catch_LT
SSB_LT
F_LT
B
PAN
1.00
0.75
0.50
0.25
0.00
FAA
SpE-E
SpD
SpE-F
Catch_ST
SSB_ST
Catch_LT
SSB_LT
Catch_AAV
SSB_AAV
F_AAV
C
Short-term Performance (Raw): Years 31 to 33
AACV
Catch
SSB
Model
Value
D
Long-term Performance (Raw): Last 3 Years
Score
Model
PAN
FAA
SpD
SpE-F
SpE-E


Mean Recruitment from the Last EM
70000
60000
50000
Mean_Rec
27500
25000
22500
20000
17500
Mean_Rec_1
30000
25000
20000
15000
Mean_Rec_2
PAN
FAA
SpD
SpE-F
SpE-E
Model
Model
PAN
FAA
SpD
SpE-F
SpE-E


Standard Deviation of NAA from the Last EM
NAA_sigma
NAA_sigma1
NAA_sigma2
Rec_sigma
Rec_sigma1
Rec_sigma2
PAN
FAA
SpD
SpE-F
SpE-E
Model