User guidelines for Advanced Model Diagnostics with ss3diags

Hennig Winker (JRC-EC) Felipe Carvalho (NOAA)
Massimiliano Cardinale (SLU) Laurence Kell (Sea++)

25 May, 2021

Contents

1 Getting started

This vignette introduces you to the ss3diags R package, which accompanies the paper "A cookbook for using model diagnostics in integrated stock assessments" by Carvalho, Winker et al. (2021).

The `ss3diags` comprises a explains set of functions for applying advanced model diagnostics for Stock Synthesis models. with the ss3diags R package for model. The ss3diag R package accompanies the paper "A cookbook for using model diagnostics in integrated stock assessments" by Carvalho, Winker et al. (2021). The `ss3diags` package builds on the widely used R package r4ss (Taylor et al. 2021), which is designed to support the use of the Stock Synthesis software modeling framework (Methot and Wetzel, 2013). This vignette is divided into four sections. Section 1 consists of installing `ss3diags` and loading the data from the two case studies presented in the cookbook. Section 2 describes the plotting of various model diagnostics as described in the Cookbook. Section 3 provides a detailed explanation on how to assess model uncertainty using `ss3diags`. In Section 4 we provide a series "cookbook recipes" on how to implement selected model diagnostics in Stock Synthesis.

1.1 Installation

Both `ss3diags` and `r4ss` can be installed from gihtub using library(devtools):

```
installed.packages("devtools")

devtools::install_github("JABBAmodel/r4ss")
```

```
devtools::install_github("JABBAmodel/ss3diags")
```

```
library(r4ss)
library(ss3diags)
```

1.2 Loading built-in example data

The package contains two fully worked examples of Stock Synthesis assessments as presented in the Cookbook.

1.2.1 The Pacif hake (*Merluccius productus*) base case model

```
data("pac.hke")
```

Individual list objects generated using R package r4ss:

- `ss3phk`: output from Stock Synthesis as read by `r4ss::SS_output()`

```
dir.path = "C:/Users/henni/Dropbox/ss3diags_demo/PacificHake"
ss3pke = r4ss::SS_output(file.path(dir.path, "Reference_Run"))
```

- `retro.phk`: list of retrospective runs with `r4ss:SS_doRetro()` as read by `r4ss::SSgetoutput()`

The retrospective runs, produced with `r4ss:SS_doRetro()`, were located in the subfolders `/Retro_Reference_Run` and named "retro0","retro-1",. . . , "retro-7" and so on. To load those at once we use the `r4ss::SSgetoutput()`. Eight retrospective runs were conducted, where "retro0" corresponds to the "Reference_Run" and "retro-1" to "retro-7" are "peels". To assign the model names, specify `start.retro = 0` and `end.retro = 7` below.

```
start.retro = 0
end.retro = 7
retro.runs = "Retro_Reference_Run"
# load models
retro.phk <- r4ss::SSgetoutput(dirvec = file.path(dir.path, retro.runs, paste0("retro",
    start.retro:-end.retro)))
```

The list object `retro.phk[[1]]` ("retro0") corresponds to the reference run ss3phk

- `aspm.phk`: Comprises of two runs the "Reference_Run" and the "ASPM", which can be loaded together using `r4ss::SSgetoutput()` and then summarized with `r4ss::SSsummarize()`

```
asem.phk <- r4ss::SSgetoutput(dirvec = file.path(dir.path, paste0("Reference_Run",
    "APSM")))
asem.phk <- r4ss::SSsummarize(asem.phk)
```

1.2.2 North Atlantic shortfin mako (*Isurus oxyrinchus*)

```
data("natl.sma")
```

Individual list objects generated using R package r4ss:

- `ss3sma`: output from Stock Synthesis as read by `r4ss::SS_output()`
- `retro.sma`: list of retrospective runs with `r4ss:SS_doRetro()` as read by `r4ss::SSgetoutput()`
- `aspm.sma`: list of Stock Synthesis referece and aspm created with `r4ss::SSsummarize()`

2 Model Diagnostics with ss3diags

2.1 Residual diagnostics

The plotting options are kept mainly to those provided by r4ss. Like with r4ss, if, for example, ‘SSplotRunstest() called with no further specifications several windows will open, which depends on the number abundance indices.

```
SSplotRunstest(ss3sma)
```

To visiualize the runs test for multiple indices, it is recommended to use of the function `sspar()`. The option plot `add=TRUE` in `sspar()` facilitates setting the graphic parameters so that they are suitable for `ss3diags` plots. The option `add=TRUE` prevents the plotting functions from over-writing `sspar()`.

```
sspar(mfrow = c(3, 2), plot.cex = 0.8)
rt = SSplotRunstest(ss3sma, add = T, verbose = F)
```

It is also possible to select the indices that should be plotted. For example, we excluded CPUE2 as it was not fitted (zero weight to the likelihood). This creates space to add the joint-residual plot `SSplotJABBAres` to summarize all selected indices.

```
sspar(mfrow = c(3, 2), plot.cex = 0.8)
rt = SSplotRunstest(ss3sma, add = T, indexselect = c(1, 3:6), legendcex = 0.8, verbose = F)
jr = SSplotJABBAres(ss3sma, add = T, indexselect = c(1, 3:6), legendcex = 0.55, verbose = F)
```

The default for `SSplotRunstest()` and `SSplotJABBAres()` is plot the residual runs for the abundance indices. However, it is also possible to plot the composition data by specifying `subplots="len"` (or “age”)

```
sspar(mfrow = c(3, 2), plot.cex = 0.8)
rt = SSplotRunstest(ss3sma, add = T, legendcex = 0.8, subplot = "len", verbose = F)
jr = SSplotJABBAres(ss3sma, add = T, legendcex = 0.55, legendloc = "bottomright",
    subplot = "len", verbose = F)
```

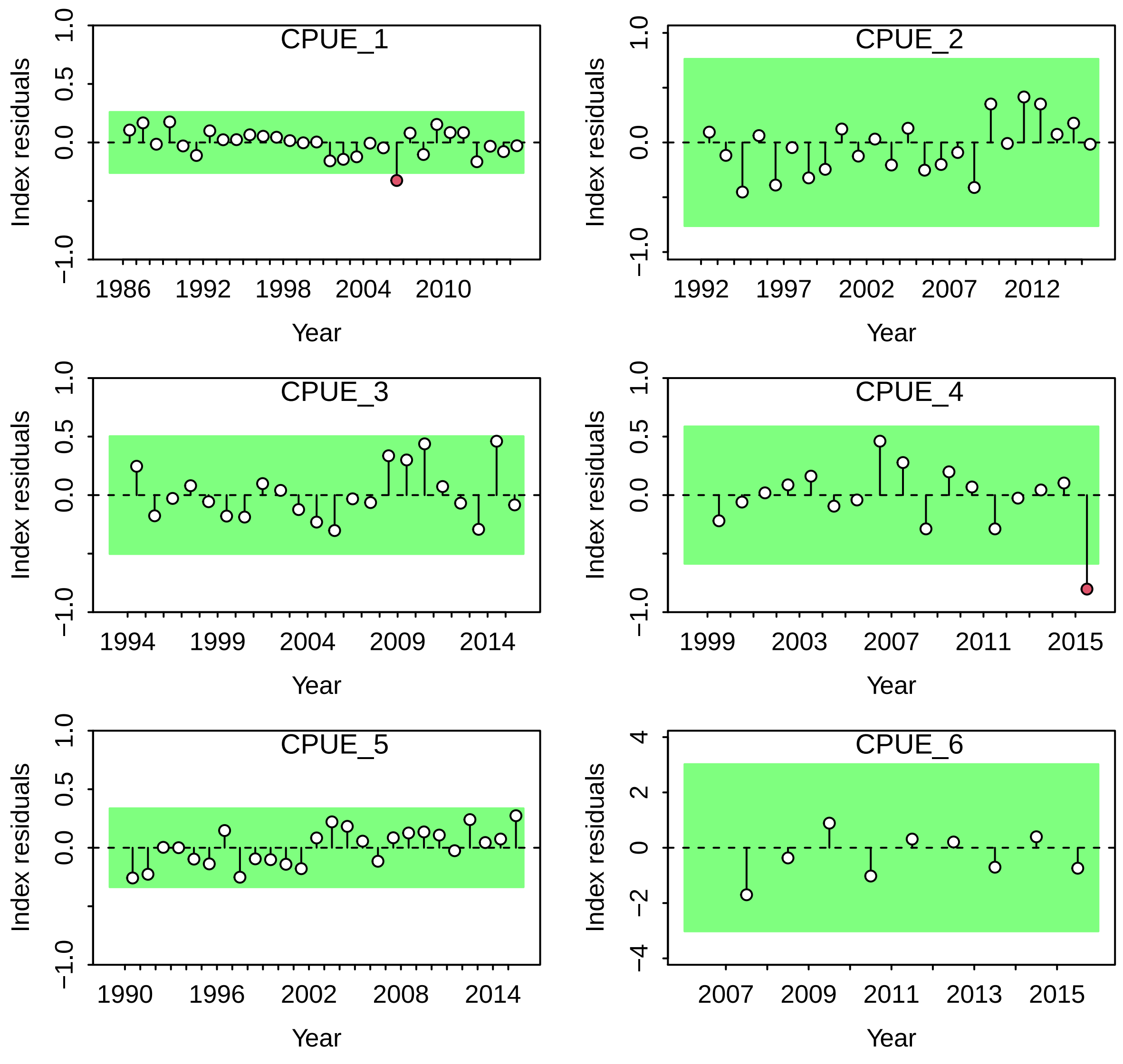


Figure 1: Runs test plots for CPUE index fits. Green shading indicates no evidence ($p = 0.05$) and red shading evidence ($p < 0.05$) to reject the hypothesis of a randomly distributed time-series of residuals, respectively. The shaded (green/red) area spans three residual standard deviations to either side from zero, and the red points outside of the shading violate the 'three-sigma limit' for that series.

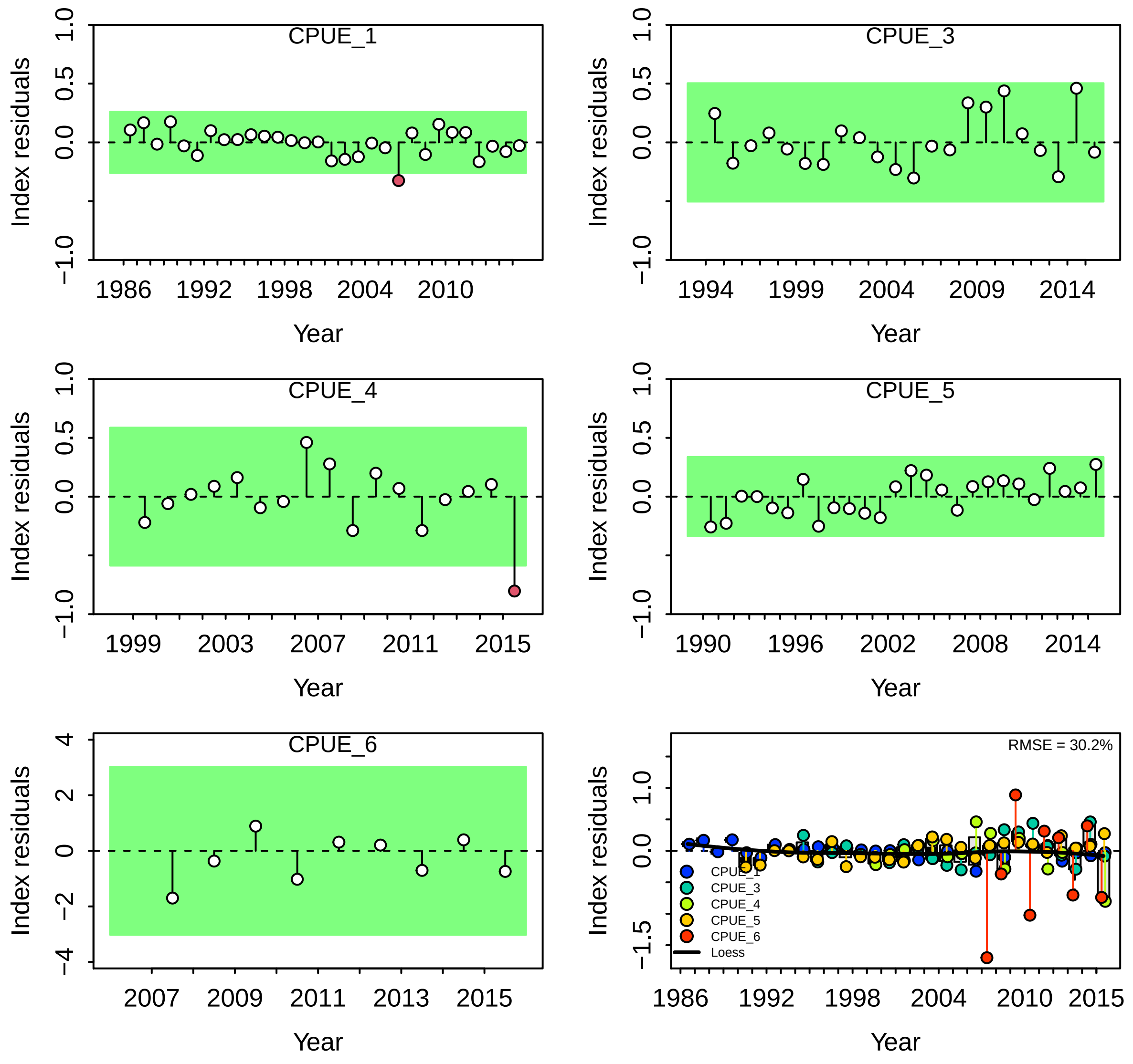


Figure 2: Runs test plot and Joint residual plot for fits to CPUE indices, where the vertical lines with points show the residuals, and solid black lines show loess smoother through all residuals. Boxplots indicate the median and quantiles in cases where residuals from the multiple indices are available for any given year. Root-mean squared errors (RMSE) are included in the upper right-hand corner of each plot.

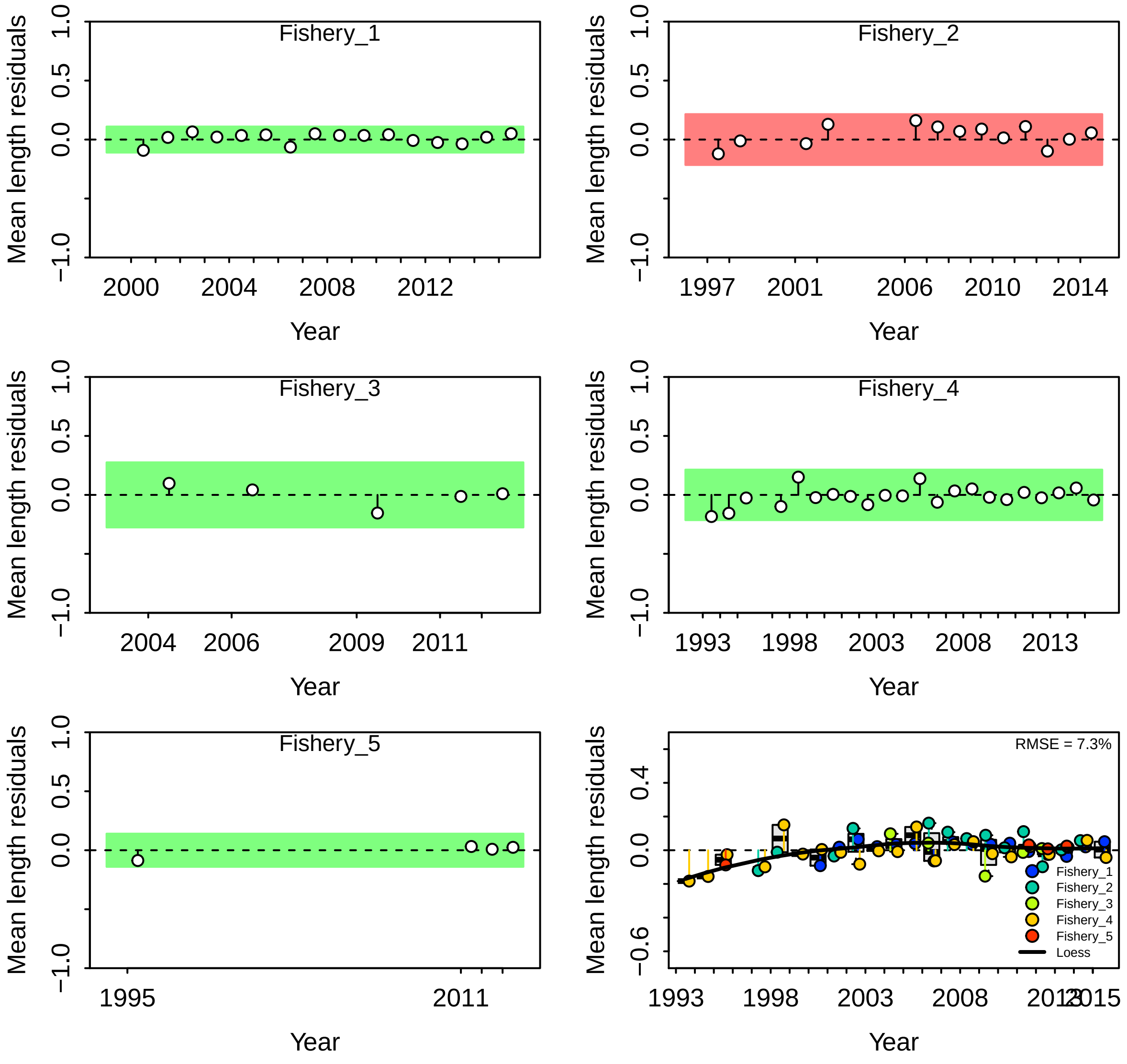


Figure 3: Runs test plot and Joint residual plot for mean lengths from fits length composition data

To facilitate automated processing, results from several diagnostic tests can also be called without plotting

```
rti = SSrunstest(ss3sma, quant = "cpue", verbose = F)
rtl = SSrunstest(ss3sma, quant = "len", verbose = F)
rbind(rti, rtl)
          Index runs.p   test  sigma3.lo sigma3.hi type
   1     CPUE_1  0.069 Passed -0.2605783 0.2605783 cpue
   2     CPUE_2  0.717 Passed -0.7630777 0.7630777 cpue
   3     CPUE_3  0.229 Passed -0.5033328 0.5033328 cpue
   4     CPUE_4  0.406 Passed -0.5868380 0.5868380 cpue
   5     CPUE_5  0.065 Passed -0.3369695 0.3369695 cpue
   6     CPUE_6  0.870 Passed -3.0226356 3.0226356 cpue
   7  Fishery_1  0.127 Passed -0.1103741 0.1103741  len
   8  Fishery_2  0.040 Failed -0.2156036 0.2156036  len
   9  Fishery_3  0.331 Passed -0.2759992 0.2759992  len
   10 Fishery_4  0.806 Passed -0.2141490 0.2141490  len
   11 Fishery_5  0.159 Passed -0.1407228 0.1407228  len
```

The Pacific hake assessment provides an example of fits to age composition instead of length composition data, which can visualized by specifying `subplots="age"`

```
sspar(mfrow = c(2, 2), plot.cex = 0.8)
rti = SSplotRunstest(ss3phk, add = T, legendcex = 0.8, subplot = "cpue", verbose = F)
rta = SSplotRunstest(ss3phk, add = T, legendcex = 0.8, subplot = "age", verbose = F)
jra = SSplotJABBAres(ss3phk, add = T, legendcex = 0.7, subplot = "age", verbose = F)
```

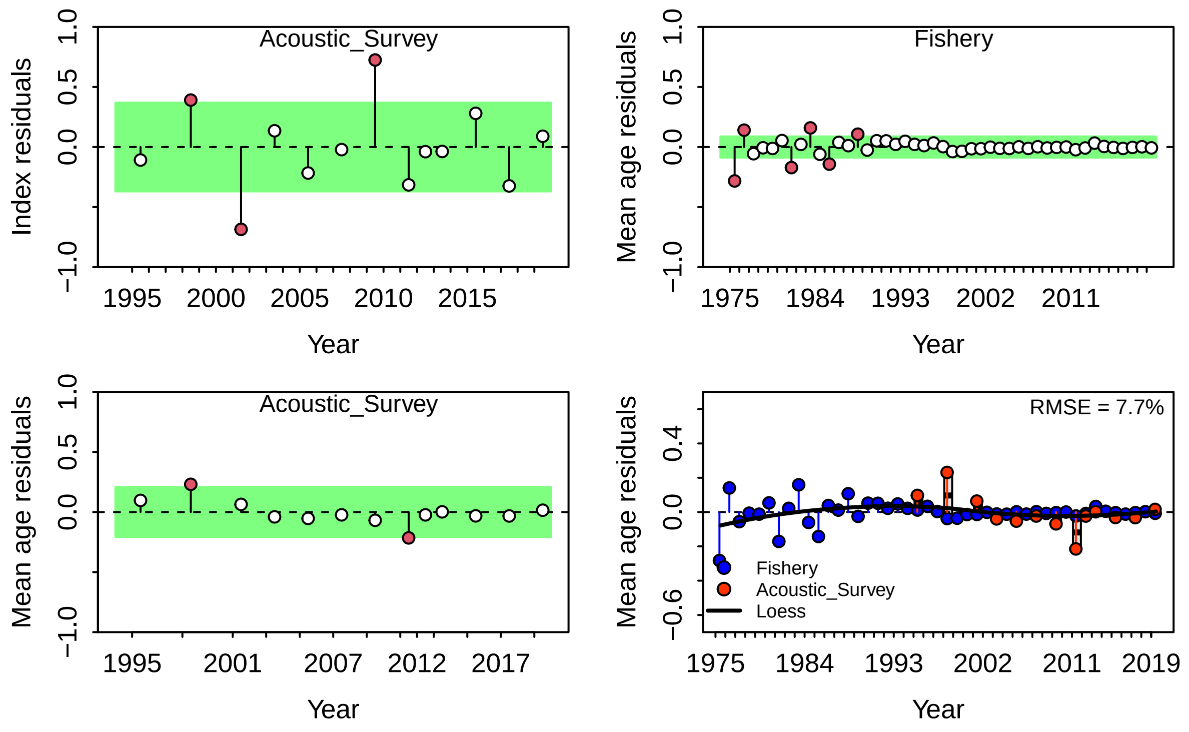


Figure 4: Runs test plot and Joint residual plot for a survey abudance index and mean ages from fits to survey and fisheries dependent age-composition data

2.2 Retrospective and Forecast bias

In Stock Synthesis, retrospective analysis can be routinely using `r4ss:SS_doRetro()` (see Section 4.1). `ss3diags` provides the function `SSplotRetro()` to visualize the retrospective patterns of SBB and F and compute the associated Mohn's rho value (i.e. retrospective bias). This would require first loading the retrospective runs (Section 1.2]), which are already inbuilt into `ss3diags` in this case. The next step is to summarize the list of retrospective runs using `r4ss::SSsummarize()`.

```
retroI.phk <- r4ss::SSsummarize(retro.phk, verbose = F)
```

We use notation "retroI" because `r4ss::SSsummarize()` summarizes the modeled quantities and abundance indices, but not length or age composition data. Using `retroI.phk` it is possible to produce some basic retrospective plots.

```
sspar(mfrow = c(1, 2), plot.cex = 0.8)
rb = SSplotRetro(retroI.phk, add = T, forecast = F, legend = F, verbose = F)
rf = SSplotRetro(retroI.phk, add = T, subplots = "F", ylim = c(0, 0.4), forecast = F,
    legendloc = "topleft", legendcex = 0.8, verbose = F)
```

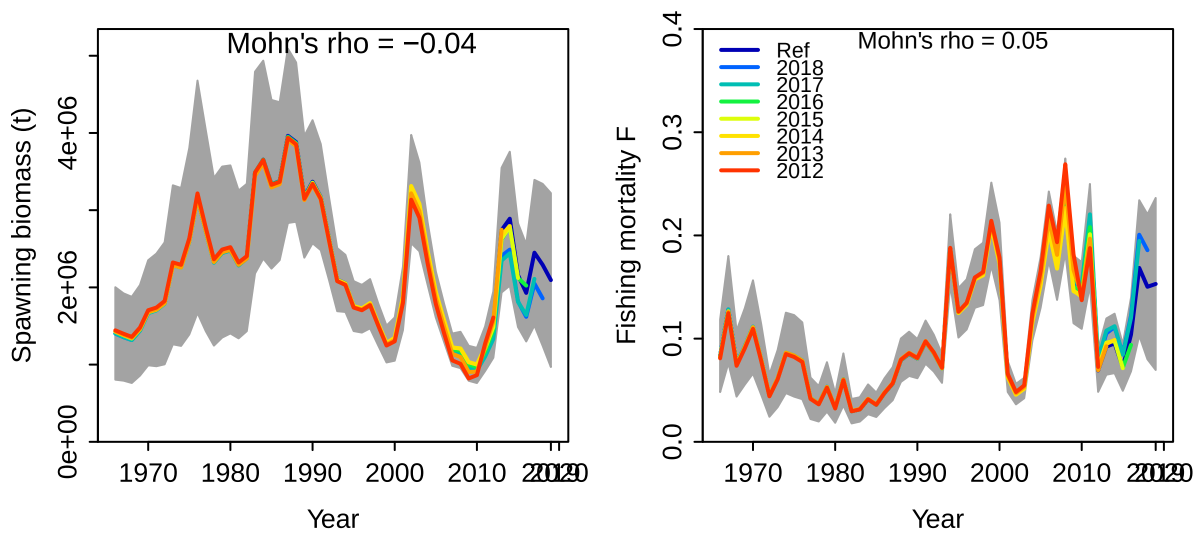


Figure 5: Retrospective analysis of spawning stock biomass (SSB) and fishing mortality estimates for Pacific hake conducted by re-fitting the reference model (Ref) after seven years, one year at a time sequentially. Mohn's rho statistic are denoted on top of the panels. Grey shaded areas are the 95 % confidence intervals from the reference model in cases where the analysis was run with Hessian

An intuitive extension of the retrospective analysis is to assess potential forecast bias by adding the additional step of forward projecting quantities, such as SSB, over the truncated years. In Stock Synthesis the forecasts are automatically done when using `r4ss:SS_doRetro()`.The forecasts are based on the settings specified in 'forecast.ss', which are also evoked when conducting future projections with the same model. The observed catches are used for the retrospective forecasts. Retrospective forecasts with Stock Synthesis are therefore only a matter of visualization, which can be done by setting the `SSplotRetro()` option `forecast=TRUE`.

```
sspar(mfrow = c(1, 2), plot.cex = 0.8)
rb = SSplotRetro(retroI.phk, add = T, forecast = T, legend = F, verbose = F, xmin = 2000)
rf = SSplotRetro(retroI.phk, add = T, subplots = "F", ylim = c(0, 0.4), forecast = T,
    legendloc = "topleft", legendcex = 0.8, verbose = F, xmin = 2000)
```

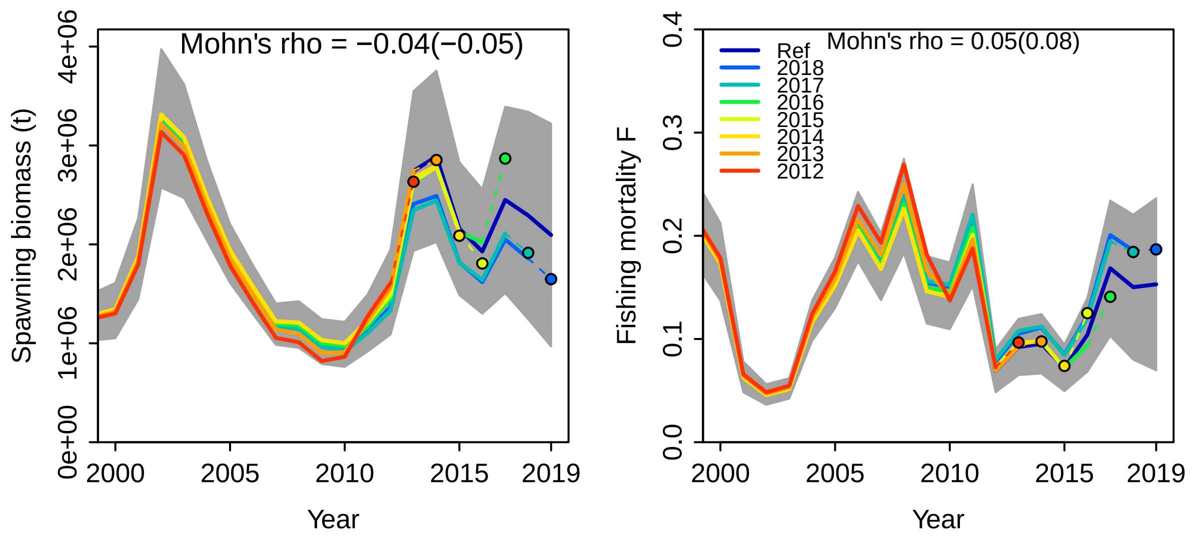


Figure 6: Retrospective results shown for the most recent years only. Mohn's rho statistic and the corresponding 'hindcast rho' values (in brackets) are now printed at the top of the panels. One-year-ahead projections denoted by color-coded dashed lines with terminal points are shown for each model.

The statistics from the retrospective analysis with forecasting, mohn's rho and forecast bias, can be called without plotting using the function `SShcbias()`

```
SShcbias(retroI.phk, quant = "SSB", verbose = F)
     type      peel          Rho  ForcastRho
   1  SSB      2018 -0.189511890 -0.21290187
   2  SSB      2017 -0.138534518 -0.16468977
   3  SSB      2016  0.048946154  0.17079324
   4  SSB      2015 -0.017403026 -0.06301299
   5  SSB      2014 -0.032761073 -0.03298098
   6  SSB      2013  0.001623057 -0.01285596
   7  SSB      2012  0.059944948 -0.03946034
   8  SSB Combined -0.038242335 -0.05072981

SShcbias(retroI.phk, quant = "F", verbose = F)
     type      peel          Rho  ForcastRho
   1    F      2018  0.235928805  0.22117119
   2    F      2017  0.154620273  0.22678319
   3    F      2016 -0.096502898 -0.16382344
   4    F      2015 -0.008902238  0.20182834
   5    F      2014  0.034769554  0.02664839
   6    F      2013  0.003690925  0.02342468
   7    F      2012  0.049728943  0.04693738
   8    F Combined  0.053333338  0.08328139
```

2.3 Hindcast Cross-Validation and prediction skill

Implementing the Hindcast Cross-Validation (HCxval) diagnostic in Stock Synthesis requires the same model outputs generated by `r4ss:SS_doRetro()` as described in Section 3.1. Therefore, no additional step is needed for HCxval if conducted in conjunction with retrospective analysis. As a robust measure of prediction skill, we implemented the mean absolute scaled error (MASE). In brief, the MASE score scales the mean absolute error (MAE) of forecasts (i.e., prediction residuals) to MAE of a naïve in-sample prediction, which is realized in the form of a simple ‘persistence algorithm’, i.e. tomorrow’s weather will be the same as today’s (see Eq. 3, p.5 in Carvalho and Winker et al. 2021). A MASE score > 1 indicates that the average model forecasts are worse than a random walk. Conversely, a MASE score of 0.5 indicates that the model forecasts twice as accurately as a naïve baseline prediction; thus, the model has prediction skill.

HCxval is implemented using function `SSplotHCxval()`, which produces the novel HCxval diagnostic plot and computes the MASE scores for CPUE indices, mean lengths or mean ages that have observations falling within the hindcast evaluation period.

Plotting HCxval for abundance indices requires the same step of summarizing the list of retrospective runs as for the retrospective analysis, which therefore only needs be done once. Below is a summary of the retrospective runs for shortfin mako.

```r
retroI.sma <- r4ss::SSsummarize(retro.sma, verbose = F)
```

```r
sspar(mfrow = c(3, 2), plot.cex = 0.8)
hci = SSplotHCxval(retroI.sma, add = T, verbose = F, ylimAdj = 1.3, legendcex = 0.7)
```

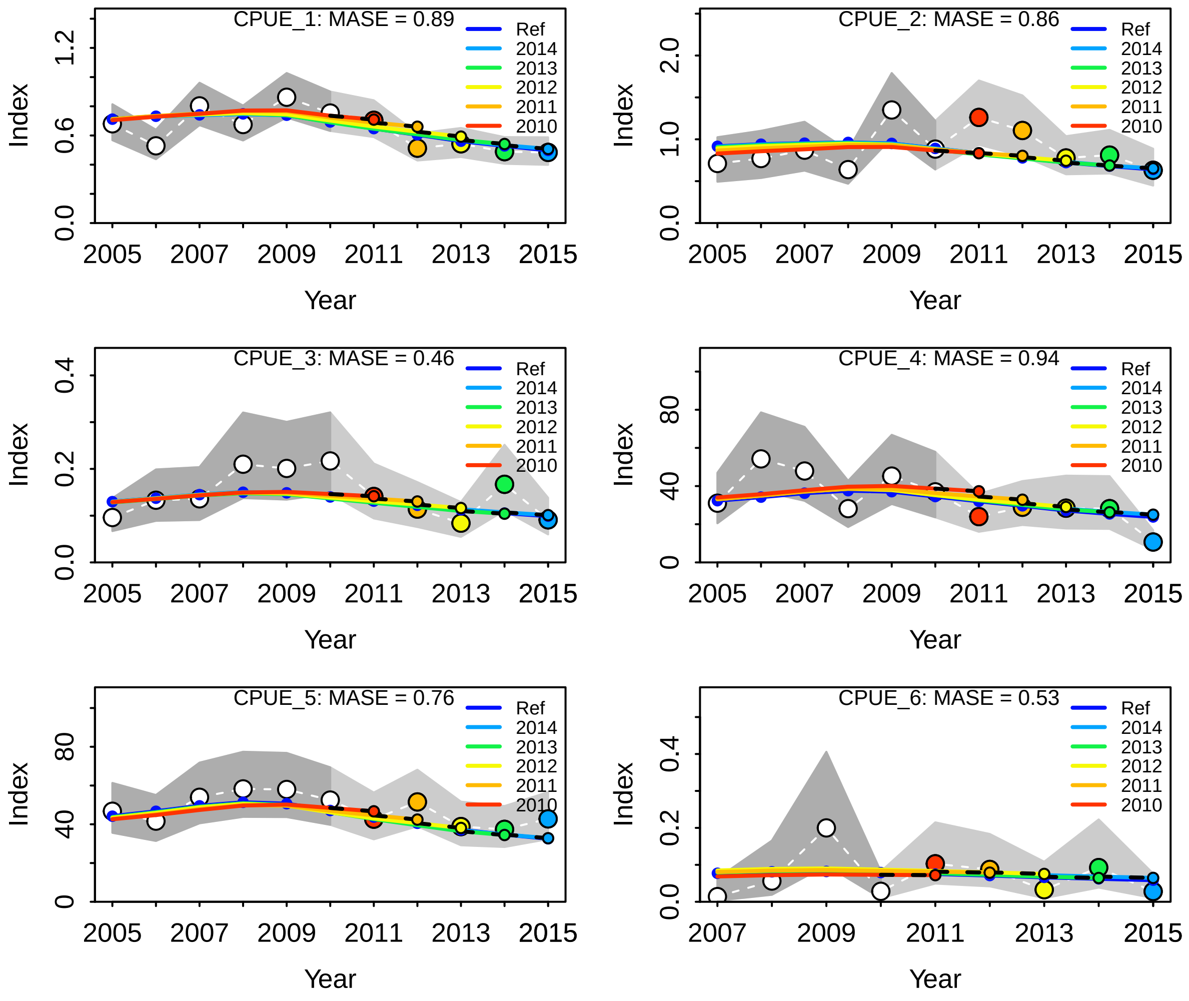


Figure 7: Hindcasting cross-validation (HCxval) results from CPUE fits, showing observed (large points connected with dashed line), fitted (solid lines) and one-yearahead forecast values (small terminal points).HCxval was performed using one reference model (Ref) and five hindcast model runs (solid lines) relative to the expected CPUE. The observations used for crossvalidation are highlighted as color-coded solid circles with associated 95 % confidence intervals. The model reference year refers to the endpoints of each one-year-ahead forecast and the corresponding observation (i.e., year of peel + 1). The mean absolute scaled error (MASE) score associated with each CPUE

The forecast length- and age-composition are located in the Stock Synthesis report.sso as “ghost files”. To extract and summarize the composition data in the form of observed and expected mean lengths and age `ss3diags` provides the function `SSretroComps()`.

```
retroC.sma = SSretroComps(retro.sma)
```

```
sspar(mfrow = c(1, 2), plot.cex = 0.8)
hcl = SSplotHCxval(retroC.sma, subplots = "len", add = T, verbose = F, ylimAdj = 1.3,
    legendcex = 0.7, indexselect = c(1, 2))
```

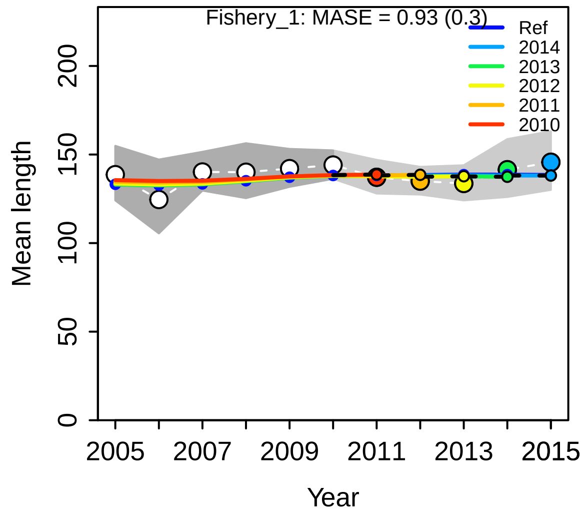


Figure 8: Hindcasting cross-validation (HCxval) results for mean lengths. Note that MASE values in breakets are adjusted MASE values for cases where naive predictions have a Mean-Absolute-Error below 0.1

The Figure above provides some additional, so called adjusted MASE values, in brackets. This gets invoked in cases where the inter-annual variation in the observed values is very small (default MAE < 0.1 for naive predictions log(y[t+1])-log(y[t])). The reasoning is that prediction residuals must be already very accurate to fall below this threshold. The adjusted MASE essential keep the naive prediction MAE denominator of the MASE to a maximum. Below we show the effect of changing adjustment threshold from the default `MAE.base.adj = 0.1`

```
mase1 = SSmase(retroC.sma, quant = "len", MAE.base.adj = 0.1, indexselect = c(1:2))

Computing MASE with all 5 of 5 prediction residuals for
Index Fishery_1

Computing MASE with only 4 of 5 prediction residuals for
Index Fishery_2

Warning: Unequal spacing of naive predictions residuals may
influence the interpretation of MASE

MASE stats by Index:
mase1
Index Season MASE MAE.PR MAE.base MASE.adj n.eval
1 Fishery_1 1 0.9265301 0.02981727 0.03218165 0.2981727 5
2 Fishery_2 1 0.6504563 0.07615571 0.11708045 0.6504563 4
```

to a larger value `MAE.base.adj = 0.15`

```
SSmase(retroC.sma, quant = "len", MAE.base.adj = 0.15, indexselect = c(1:2))

Computing MASE with all 5 of 5 prediction residuals for
Index Fishery_1

Computing MASE with only 4 of 5 prediction residuals for
Index Fishery_2

Warning: Unequal spacing of naive predictions residuals may
influence the interpretation of MASE

MASE stats by Index:
Index Season MASE MAE.PR MAE.base MASE.adj n.eval
1 Fishery_1 1 0.9265301 0.02981727 0.03218165 0.1987818 5
2 Fishery_2 1 0.6504563 0.07615571 0.11708045 0.5077048 4
```

where `MASE` is the ratio of the mean absolute error of the prediction residuals `MAE.PR` to the residuals of the naive predictions `MAE.base`

```
mase1$MAE.PR/mase1$MAE.base
   [1] 0.9265301 0.6504563
mase1$MASE
   [1] 0.9265301 0.6504563
```

and MASE.adj

```
mase1$MAE.PR/pmax(mase1$MAE.base, 0.1)
   [1] 0.2981727 0.6504563
mase1$MASE.adj
   [1] 0.2981727 0.6504563
```

Applying HCxval for composition data requires correctly specifying the composition data type fitted in the model. For example, age composition data need to be specified as "age" in `SSplotHCxval` and `SSmase`, as shown below for the Pacific hake model.

```
retroC.phk = SSretroComps(retro.phk)  # summarize comps

sspar(mfrow = c(1, 2), plot.cex = 0.8)
hcl = SSplotHCxval(retroC.phk, subplots = "age", add = T, verbose = F, ylimAdj = 1.3,
    legendcex = 0.7, indexselect = c(1, 2))
```

```
SSmase(retroC.phk, quants = "age")

    Computing MASE with all 7 of 7  prediction residuals for Index Fishery

    Computing MASE with only 4 of 7  prediction residuals for Index Acoustic_Survey

    Warning:  Unequal spacing of naive predictions residuals may influence the interpretation of MASE

   MASE stats by Index:
               Index Season      MASE     MAE.PR  MAE.base  MASE.adj n.eval
   1         Fishery      1 0.6319721 0.09063054 0.1434091 0.6319721      7
   2 Acoustic_Survey      1 0.3556384 0.05181084 0.1456841 0.3556384      4
```

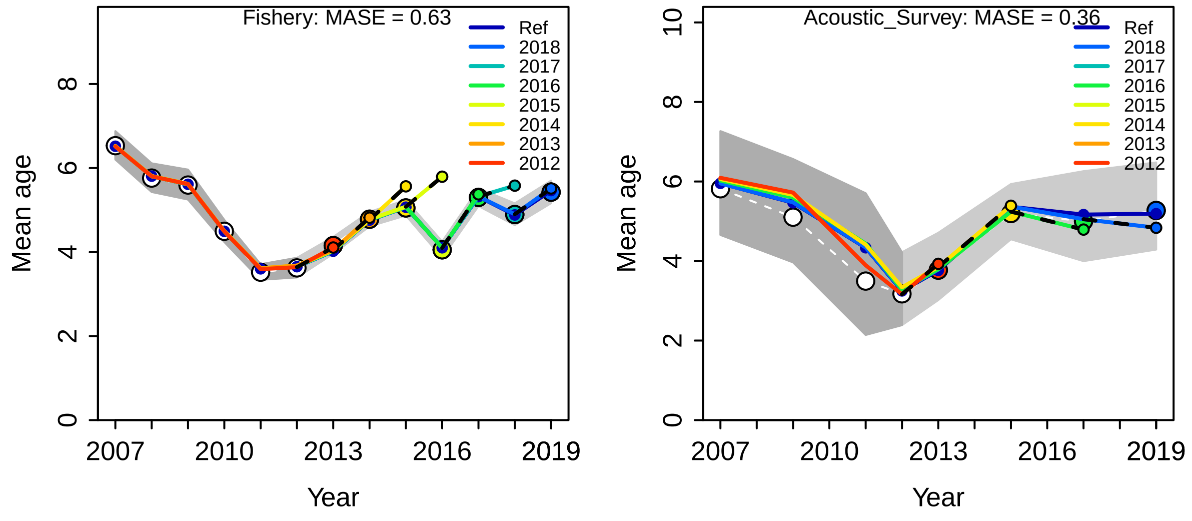


Figure 9: Hindcasting cross-validation (HCxval) results for mean lengths. Note that MASE values in brackets are adjusted MASE values for cases where naive predictions have a Mean-Absolute-Error below 0.1

3 Model uncertainty

The management advice frameworks increasingly require translating the estimated uncertainty about the stock status into probabilistic statements (Kell et al. 2016). A classical example is the Kobe framework used in tuna Regional Fisheries Management Organisations (tRFMOs) around the world. The key quantities of interest are typically the ratios SSB/SSB_{MSY} and F/F_{MSY}. It is reasonably straight forward in Stock Synthesis to approximate uncertainty of individual quantities (e.g. SSB) from the asymptotic standard errors (SE) derived from the Hessian matrix using the delta method. However, the joint distribution of SSB/SSB_{MSY} and F/F_{MSY}
requires to adequately account for the covariance structure between these two derived quantities. Joint distributions were typically constructed using bootstrap or Markov Chain Monte-Carlo (MCMC) methods. However, these methods can be computationally intense and time-consuming in integrated assessments.

As an alternative, `ss3diags` implements a rapid delta-Multivariate lognormal approximation with `SSdeltaMVLN()` to generate joint error distributions for SSB/SSB_{ref} and F/F_{ref},where the ref may refer to MSY, but also other reference points (e.g., SSB_{40} and F_{40}). In Stock Synthesis, these ratios are determined by the derived quantities `Bratio` and `F`, where either can take the form of ratios (e.g. F/F_{ref}) or absolute value (e.g. `absF`) depending on settings in the `starter.ss` file.

Let `Bratio` be $u = SSB/SSB_{ref}$, `F` be $v = F$, and $w = F_{ref}$ be the F reference point of interest (e.g. F_{MSY}), with $x = \log(u)$, $y = \log(v)$ and $z = \log(w)$, then the variance-covariance matrix VCM has the form:

$$VCM_{x,y,z} = \begin{pmatrix} \sigma_x^2 & cov_{x,y} & cov_{x,y} \\ cov_{x,y} & \sigma_y^2 & cov_{y,z} \\ cov_{x,z} & cov_{y,z} & \sigma_z^2 \end{pmatrix}$$

where, e.g., σ_x^2 is the variance of x and $cov_{x,y}$ is the covariance of x and y. Deriving those requires conducting a few normal to lognormal transformations. First, the variances are approximated as:

$$\sigma_x^2 = \log\left(1 + \left(\frac{SE_u}{u}\right)^2\right)$$

where SE_u, SE_v and SE_z are the asymptotic standard error estimates for $u = SSB/SSB_{ref}$, $v = F$ and $z = F_{ref}$.

The corresponding covariance for x and y, can then be approximated on the log-scale by:

$$COV_{x,y} = \log\left(1 + \rho_{u,v}\sqrt{\sigma_x^2 \sigma_y^2}\right)$$

where $rho_{u,v}$ denotes the correlation of u and v.

To generate a joint distribution of $\tilde{u} = SSB/SSB_{ref}$, $\tilde{v} = F$ and $\tilde{z} = F_{ref}$, a multivariate random generator is used, which is available in the R package 'mvtnorm', to obtain a large number (e.g. nsim = 10,000) iterations, such that

$$JD(\tilde{u}, \tilde{v}, \tilde{w}) = \exp(MVN(\mu_{x,y,z}, VCM_{x,y,z}))$$

so that

$$S\tilde{S}B/S\tilde{S}B_{MSY} = \tilde{u}$$

and

$$\tilde{F}/\tilde{F}_{MSY} = \tilde{v}/\tilde{w}$$

The reference points depend on the settings in the `starter.ss` file that determine the derived quantities `Bratio` and `Fvalue`.

In the first example, we consider the `ss3sma` Stock Synthesis model, which was run with `starter.ss` settings that are common to produce target Kobe plot estimates of SSB/SSB_{MSY} and F/F_{MSY} in tRFMO assessments:

```
2 # Depletion basis: 2=rel SPBmsy; 3=rel X*SPB_styr; 4=rel X*SPB_endyr
```

```
2 # F_report_basis: 0=raw_F_report; 1=F/Fspr; 2=F/Fmsy ; 3=F/Fbtgt
```

To generate a joint MVLN of SSB/SSB_{MSY} and F/F_{MSY}, the default options can be used.

```
mvln = SSdeltaMVLN(ss3sma, run = "SMA")

    starter.sso with Bratio: SSB/SSBMSY and F: (F)/(Fmsy)
```

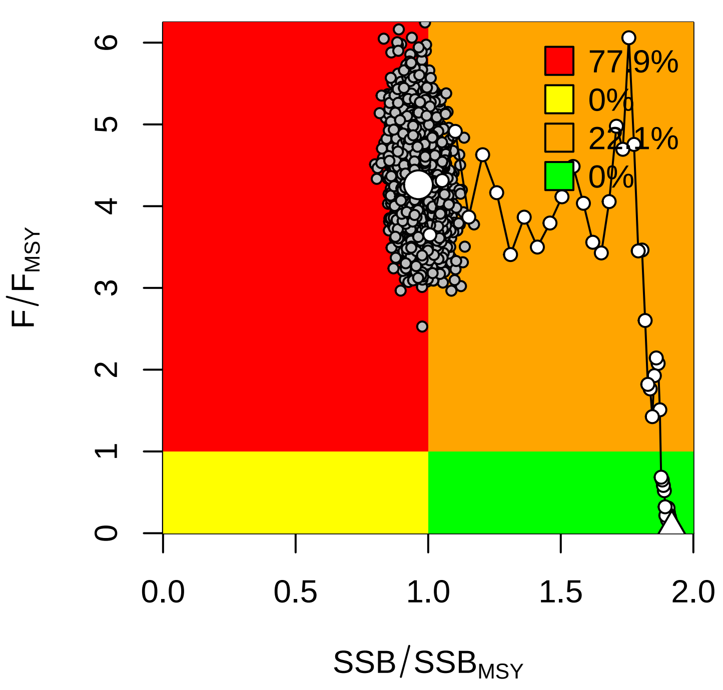


Figure 10: Kobe phase plot showing MVLN Kobe probability distributions of SSB/SSB_{MSY} and F/F_{MSY} for North Atlantic shortfin mako Stock model.

We provide the function `SSsettingsBratioF(ss3sma)` to the `starter.ss` settings:

```
SSsettingsBratioF(ss3sma)
   $Bratio
   [1] "SSB/SSBMSY"

   $F
   [1] "(F)/(Fmsy)"
```

```
$Bref
[1] 0.4
```

This function is also inbuilt in the `SSdeltaMVLN()` to prevent misleading results. The `SSdeltaMVLN()` output include the maximum likelihood estimates (mles) and the MVLN monte-Carlo distributions `$kb` of SSB/SSB_{MSY}, F/F_{MSY} and F. Note the additional quantities SSB and Rec are generated independently from lognormal distributions for practical reasons. These can be plotted by

```
sspar(mfrow = c(3, 2), plot.cex = 0.7)
SSplotEnsemble(mvln$kb, ylabs = mvln$labels, add = T, verbose = F)
```

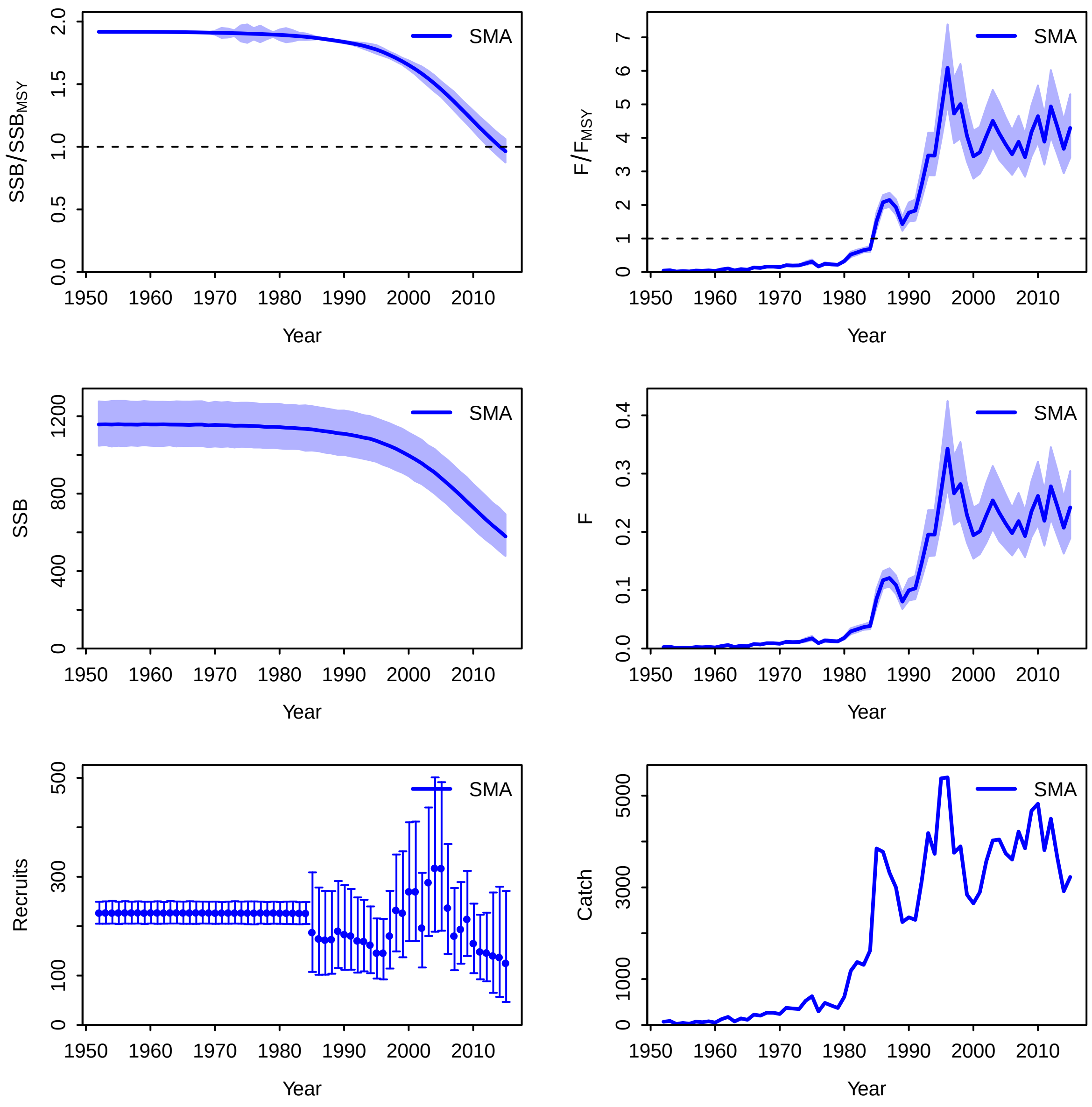


Figure 11: Distributions of SSB/SSB_{MSY}, F/F_{MSY}, SSB, F ,Recruitment and Catch trajectories for the North Atlantic shortfin mako SS3 model

The `SSdeltaMVLN()` provides the option to set alternative `Fref` values, but this is only possible for the recommended `starter.ss` option 0 for `F_report_basis`. For option 2, `SSdeltaMVLN()` prompts an error if `Fref` is changed.

By comparison, the Pacific Hake base case model `ss3phk` is run with settings that are common in NOAA assessments, with `Bratio` set SSB/SSB_0 and `F` is typically kept at absolute quantity.

`1 # Depletion basis: 2=rel SPBmsy; 3=rel X*SPB_styr; 4=rel X*SPB_endyr`

`0 # F_report_basis: 0=raw_F_report; 1=F/Fspr; 2=F/Fmsy ; 3=F/Fbtgt`

The management quantities in this case are SSB/SSB_{40} and F/F_{spr40}, where the target of 40% is specified in the `forecast.ss` file.

`0.4 # SPR target (e.g. 0.40) 0.4 # Biomass target (e.g. 0.40)`

```
mvln = SSdeltaMVLN(ss3phk, run = "Pac.Hake", Fref = "SPR", plot = TRUE)

    starter.sso with Bratio: SSB/SSB0 and F: _abs_F
```

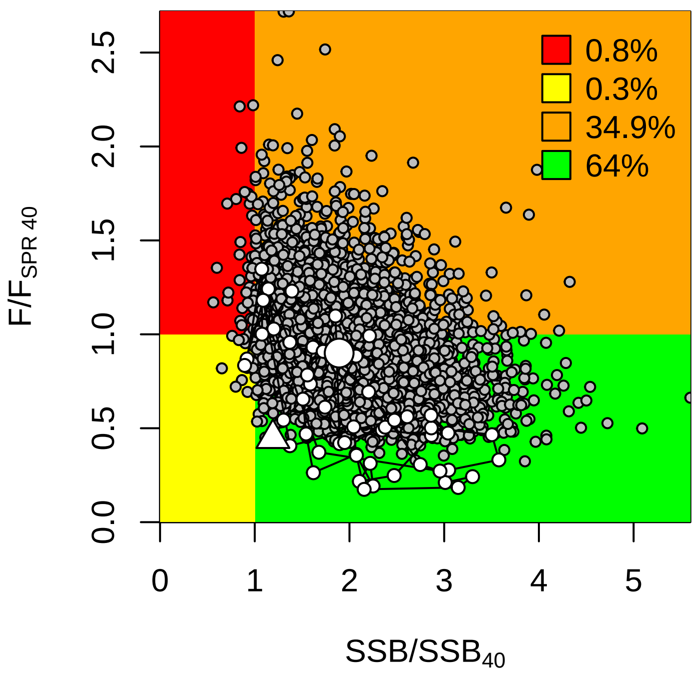


Figure 12: Kobe phase plots showing MVLN Kobe probability distributions of SSB/SSB_{40} and F/F_{SPR40} for North Atlantic shortfin mako Stock model.

```
sspar(mfrow = c(3, 2), plot.cex = 0.7)
SSplotEnsemble(mvln$kb, ylabs = mvln$labels, add = T, verbose = F)
```

It is important to note that MVLN approximation differs notably from MCMC posterior for this model as documented in Stewart et al. (2013) and Taylor et al. (2021). Such differences may be more likely to occur in

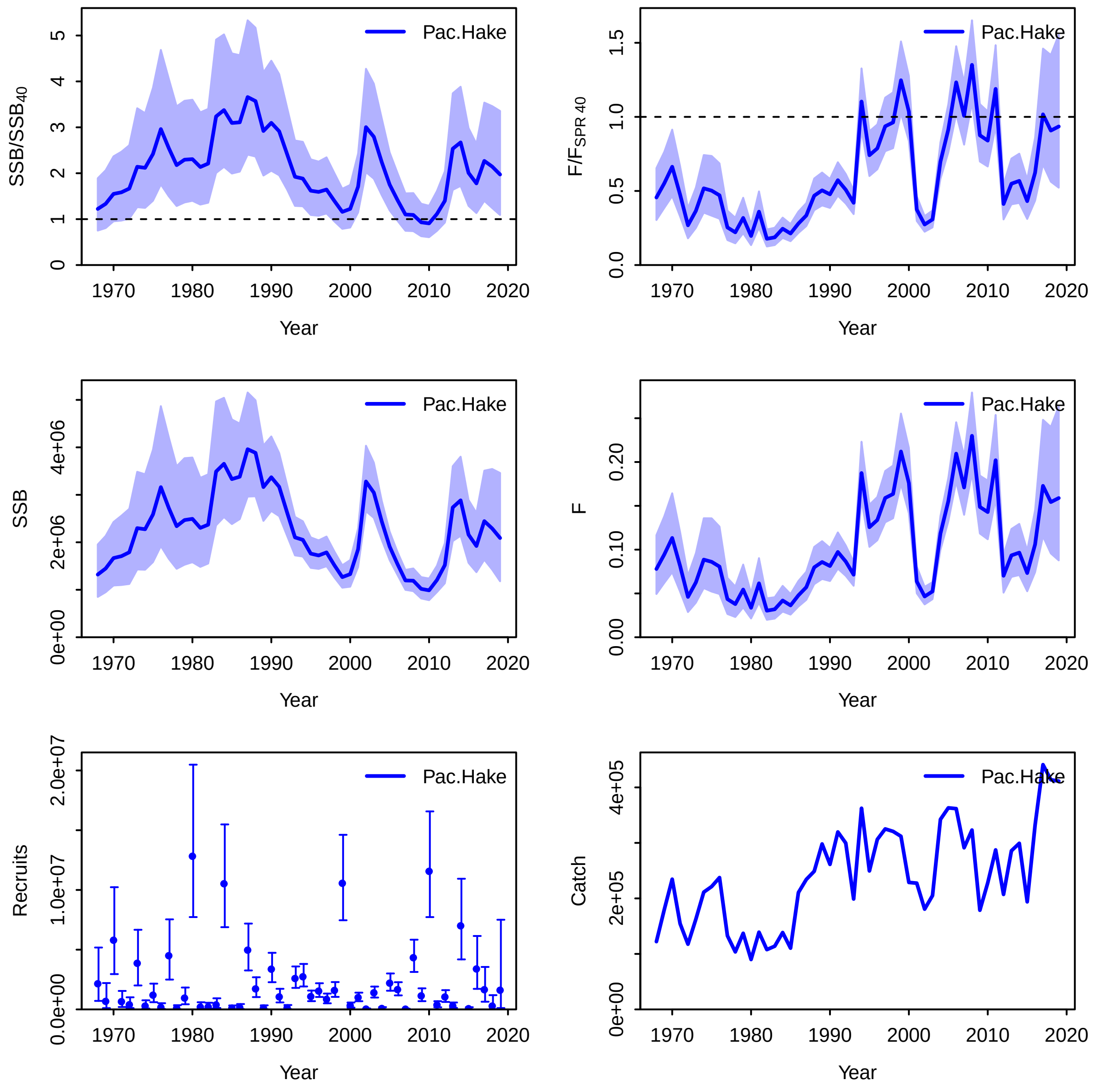


Figure 13: Distributions for SSB/SSB_{40}, F/F_{SPR40}, SSB, F, Recruitment and Catch trajectories for the Pacific Hake SS3 model

cases where key parameters such as steepness h or natural M are estimated using informative priors, which can result in left skewed (non-lognormal) distributions of the benchmarks F_{ref} and B_{ref}.

In other instances, mismatches between the`SSdeltaMVLN` and MCMC may also be caused by the latter's poor performance due to poor regularization (Monnahan et al., 2019).

To facilitate a comparison between the `SSdeltaMLVN()` and MCMC outputs, we provide the function `SSdiagsMCMC()`, which is illustrated on the example of the Stock Synthesis model for the ICES Gulf of Bothian Herring stock (ICES, 2021).

`SSdiagsMCMC()` requires loading both the `report.sso` and MCMC output in the `posterior.sso` file, where the MCMC was in this case run in the subfolder of the assessment file `/mcmc`.

```
# Alread loaded to ss3diags
ss3her = SS_output("gob_her")
mcmc.her = SSreadMCMC("gob_her/mcmc")
```

The options and output of `SSdiagsMCMC()` are largely identical to `SSdeltaMVLN`. In this case, the `starter.ss` is the same as for `ss3phk`, only we use Fref = "Btgt" for illustration with SB_{50} and F_{SB50} for this illustration.

```
mvln = SSdeltaMVLN(ss3her, Fref = "Btgt", plot = F, run = "mvln")

    starter.sso with Bratio: SSB/SSB0 and F: _abs_F

mcmc = SSdiagsMCMC(mcmc.her, ss3her, Fref = "Btgt", plot = F, run = "mcmc")

    starter.sso with Bratio: SSB/SSB0 and F: _abs_F
```

Comparing delta-MVLN with MCMC simply requires combining the `$kb` outputs by, e.g.,

```
sspar(mfrow = c(1, 1), plot.cex = 0.8)
SSplotKobe(rbind(mvln$kb, mcmc$kb), joint = F, xlab = mvln$labels[1], ylab = mvln$labels[2],
    fill = F)
```

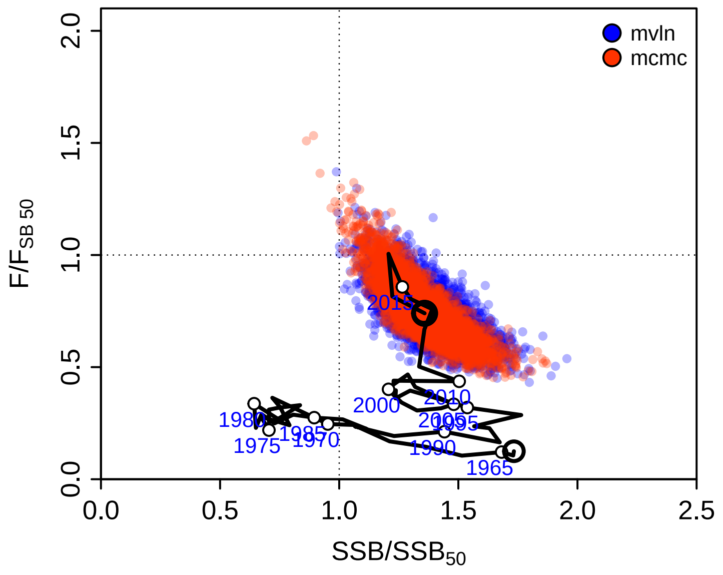


Figure 14: Kobe phase plot comparing MVLN and MCMC posterior distributions of SSB/SSB_{50} and F/F_{50} for the ICES Gulf of Bothia Herring SS3 model

```
  Quadrant     Percent
1      Red  0.08002401
2   Orange  3.06091828
3   Yellow  0.00000000
4    Green 96.85905772
```

```
sspar(mfrow = c(2, 2), plot.cex = 0.7)
SSplotEnsemble(rbind(mvln$kb, mcmc$kb), ylabs = mvln$labels, add = T, subplots = c("stock",
    "harvest", "SSB", "F"), verbose = F)
```

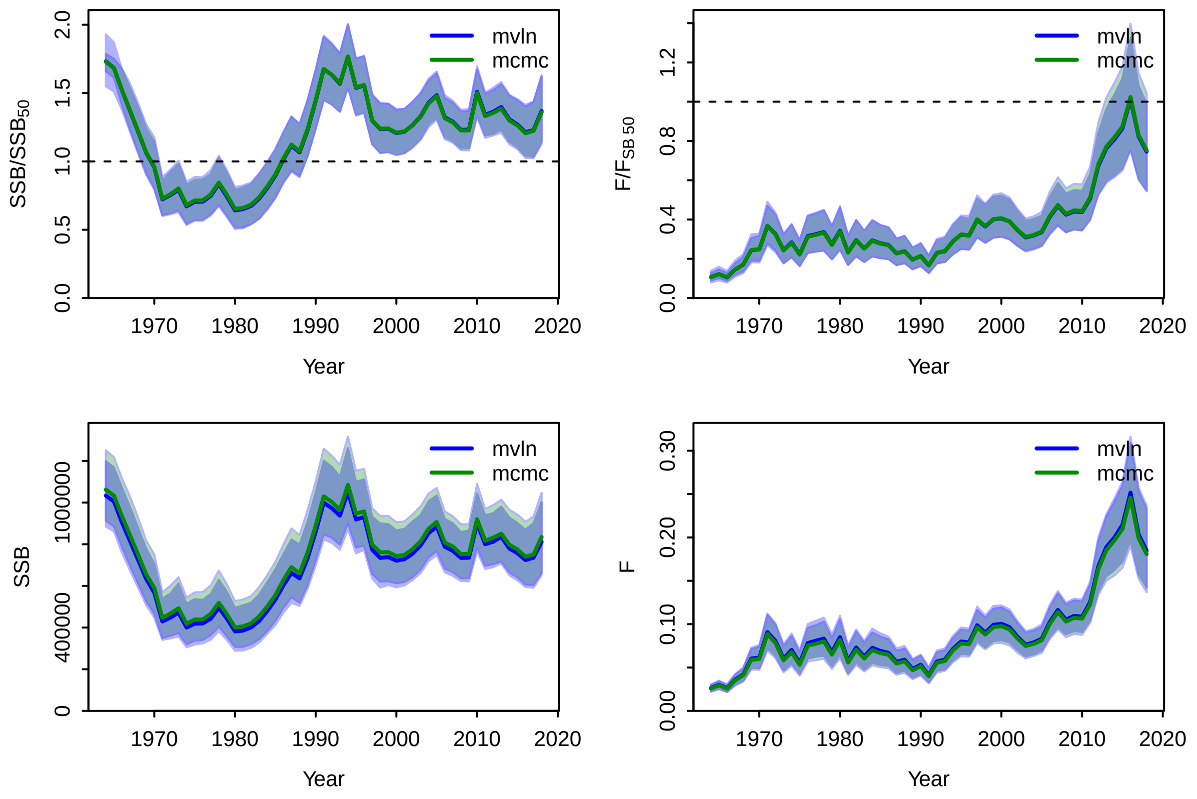


Figure 15: Comparison of MVLN and MCMC posterior distributions for SSB/SSB_{40}, F/F_{SB40}, SSB and F for the ICES Gulf of Bothia Herring SS3 model

This works equally for joining a model ensemble.

4 Cookbook Recipies

4.1 Retrospectives with hindcasts

Retrospective analysis can be run for Stock Synthesis using the function `r4ss::SS_doRetro()` available in r4ss. This setup of the retrospective analysis has the advantage that forecasts are conducted automatically given the catch. This makes it possible to apply retrospective forecasting and hindcast cross-validations of observations based on the same output.

```
library(r4ss)
```

Below is a step-by-step cookbook recipe for retrospective analysis in Stock Synthesis.

4.1.1 Step1: Identify restrospective period

Specify the range pf peels that will then determine the `end.yr.vec` of runs in `r4ss::SS_doRetro()`

```
start.retro <- 0  # end year of reference year
end.retro <- 7  # number of years for retrospective e.g.,
```

4.1.2 Step 2: Identify the base directory

Specify the path directory that holds the folder with the base case run. In this case the Pacific Hake folder with a model folder 'Reference_Run"

```
dirname.base = "C:/Users/henni/Dropbox/ss3diags_demo/PacificHake"
run = "Reference_Run"

model.run <- file.path(dirname.base, run)

model.run
```

### Step 3: `DAT` and `CONTROL` files

Specify the names of the data and control files. Note these files are named differently from the `DATA.ss` and `CONTROL.ss`. In this case

```
DAT = "phk.dat"
CTL = "phk.ctl"
```

The names of the DAT and CONTROL are declared on the top of the 'starter.ss', e.g.

`#C Hake starter file phk.dat phk.ctl`

4.1.3 Step 4: Create a subdirectory for the Retrospectives

There are several ways to organize the retrospective output structure. First create a new subfolder for the retrospective runs output

```
dir.retro <- paste0(dirname.base, "/Retro_", run)

dir.create(path = dir.retro, showWarnings = F)
```

Also create a subdirectory for the retrospective model folders

```
dir.create(path = file.path(dir.retro, "retros"), showWarnings = F)
```

then copy model run files to the new retrospective folder

```
file.copy(file.path(model.run, "starter.ss_new"), file.path(dir.retro, "starter.ss"))

file.copy(file.path(model.run, "control.ss_new"), file.path(dir.retro, CTL))

file.copy(file.path(model.run, "data.ss_new"), file.path(dir.retro, DAT))

file.copy(file.path(model.run, "forecast.ss"), file.path(dir.retro, "forecast.ss"))

file.copy(file.path(model.run, "SS.exe"), file.path(dir.retro, "SS.exe"))

# Automatically ignored for models without wtatage.ss
file.copy(file.path(model.run, "wtatage.ss"), file.path(dir.retro, "wtatage.ss"))
```

4.1.4 Step 5: Modify Starter.ss file

Modifying the Starter File helps to speed up model runs

```
starter <- readLines(paste0(dir.retro, "/starter.ss"))

# [8] '2 # run display detail (0,1,2)'
linen <- grep("# run display detail", starter)
starter[linen] <- paste0(1, " # run display detail (0,1,2)")
# write modified starter.ss
write(starter, file.path(dir.retro, "starter.ss"))
```

### Step 6: Execute retrospective runs

Run the retrospective analyses using r4SS function `r4ss::SS_doRetro` Ideally, the runs should be done with Hessian to evaluate the retrospective trajectories with respect to the confidence interval coverage of the reference model.

```
r4ss::SS_doRetro(masterdir = dir.retro, oldsubdir = "", newsubdir = "", years = start.retro:-end.retro)
```

However, for larger models it might be desirable to shorten run times, by not inverting the hessian, using the option `extras = "-nohess"` (much faster)

```
r4ss::SS_doRetro(masterdir = dir.retro, oldsubdir = "", newsubdir = "", years = start.retro:-end.retro,
    extras = "-nohess")
```

4.1.5 Step 7: Read `SS_doRetro()` output

```
retro.phk <- r4ss::SSgetoutput(dirvec = file.path(dir.retro, paste0("retro", start.retro:-end.retro)))
```

It is often useful to save the retro model runs as `.rdata` file for further processing with `ss3diags`, considering that reading the models with `r4ss::SSgetoutput()` can be quite time-consuming for more complex models.

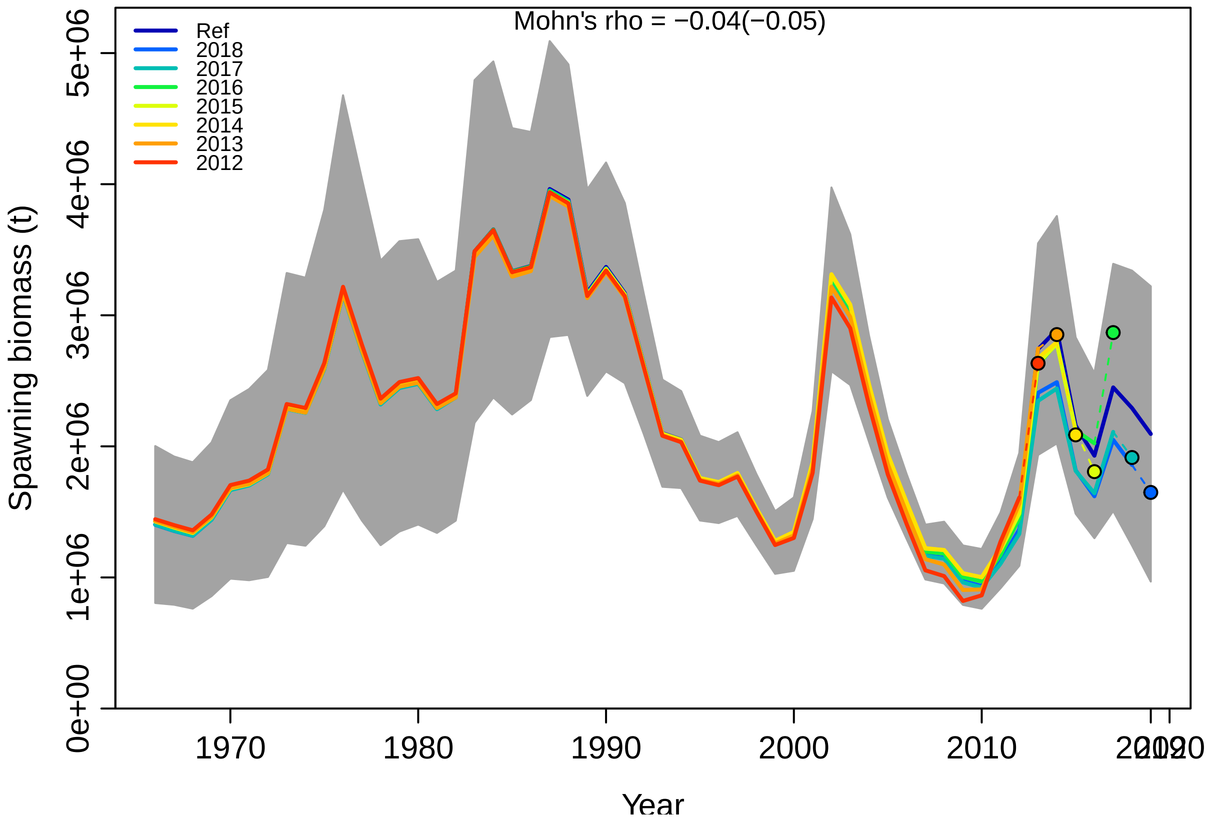
```
save(retro.phk, file = file.path(dir.retro, "retro.phk.rdata"))
```

4.1.6 Step 8: Check

```
library(ss3diags)

check.retro = r4ss::SSsummarize(retro.phk)
   Summarizing 8 models:
   imodel=1/8
```

```
      N active pars = 237
    imodel=2/8
      N active pars = 237
    imodel=3/8
      N active pars = 237
    imodel=4/8
      N active pars = 237
    imodel=5/8
      N active pars = 237
    imodel=6/8
      N active pars = 237
    imodel=7/8
      N active pars = 237
    imodel=8/8
      N active pars = 237
    Summary finished. To avoid printing details above, use 'verbose = FALSE'.
sspar(mfrow = c(1, 1))
SSplotRetro(check.retro, forecast = T, add = T, legendcex = 0.7, legendloc = "topleft")
    Plotting Retrospective pattern
```



```
    Mohn's Rho stats, including one step ahead forecasts:
      type     peel          Rho ForecastRho
    1  SSB     2018 -0.189511890 -0.21290187
    2  SSB     2017 -0.138534518 -0.16468977
    3  SSB     2016  0.048946154  0.17079324
```

```
  4  SSB     2015 -0.017403026 -0.06301299
  5  SSB     2014 -0.032761073 -0.03298098
  6  SSB     2013  0.001623057 -0.01285596
  7  SSB     2012  0.059944948 -0.03946034
  8  SSB Combined -0.038242335 -0.05072981
```

4.2 R0 profiling

4.3 ASPM diagnostic

4.4 Jittering