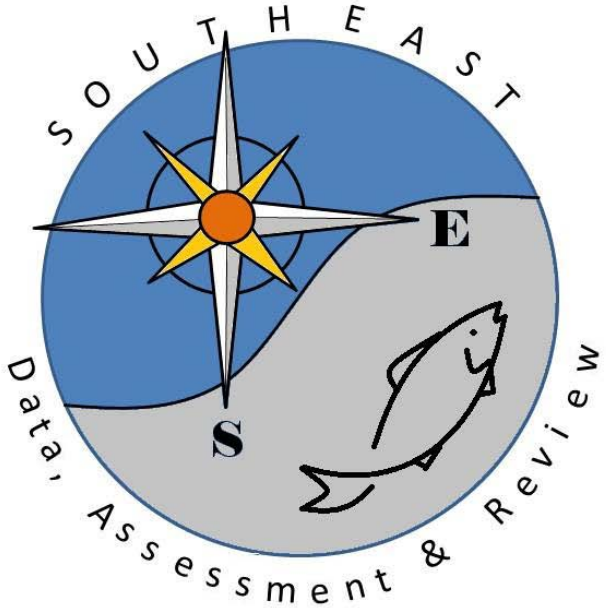
The Beaufort Assessment Model (BAM) with application to Spanish mackerel: mathematical description, implementation details, and computer code

Katie Andrews

SEDAR28-RW03

15 October 2012



Please cite as:

Andrews, K. 2012. The Beaufort Assessment Model (BAM) with application to Spanish mackerel: mathematical description, implementation details, and computer code. SEDAR28-RW03. SEDAR, North Charleston, SC. 47 pp.

The Beaufort Assessment Model (BAM) with application to Spanish mackerel: mathematical description, implementation details, and computer code

Sustainable Fisheries Branch

National Marine Fisheries Service

Southeast Fisheries Science Center

NOAA Beaufort Laboratory

101 Pivers Island Road, Beaufort, NC 28516

1 Overview

The primary model in this assessment was the Beaufort assessment model (BAM), which applies a statistical catch-age formulation as described in the AW report.

2 Model configuration and equations

Model equations are detailed in Table 2.1, and AD Model Builder code is supplied in Appendix A. A general description of the assessment model follows.

Stock dynamics In the assessment model, new biomass was acquired through growth and recruitment, while abundance of existing cohorts experienced exponential decay from fishing and natural mortality. The population was assumed closed to immigration and emigration. The model included age classes $0 - 10^{+}$, where the oldest age class 10^{+} allowed for the accumulation of fish (i.e. a plus group).

Initialization Initial (1950) abundance at age was estimated in the model as follows. First, the equilibrium age structure was computed for ages 1-10+ based on natural and fishing mortality (F), where F was set equal to a value of historical fishing mortality decided by the AW panel (0.2). The value was supported by that of the previous benchmark assessment and seemed reasonable given initial catches and the existence of both a commercial and recreational fishery prior to the start year of the model. Initial recruitment (age-0) was then calculated (as described below) based on spawning biomass associated with the initial equilibrium age structure of the stock.

Natural mortality rate The natural mortality rate (M) was assumed constant over time, but decreasing with age. The form of M as a function of age was based on Lorenzen (1996). The Lorenzen (1996) approach inversely relates the natural mortality at age to mean weight at age W_a by the power function $\mathrm{M}_a{=}\alpha W_a^{\beta}$, where α is a scale parameter and β is a shape parameter. Lorenzen (1996) provided point estimates of α and β for oceanic fishes, which were used for this assessment. As in previous SEDAR assessments, the Lorenzen estimates of M_a were rescaled to provide the same fraction of fish surviving from age-1 through the oldest observed age (12 yr) as would occur with constant $M = 0.35$ from the DW. This approach using cumulative mortality is consistent with the findings of Hoenig (1983) and Hewitt and Hoenig (2005).

Growth and maturity Mean size at age (fork length) was input to the model as three separate length at age vectors derived from different von Bertalanffy growth equations (see AW report). As suggested by the DW, separate growth curves were estimated for males and females to represent differential growth in the population as a whole. In addition, a von Bertalanffy growth curve was estimated using fishery-dependent samples to represent the fished population. Weight at age (whole weight) was input as a function of length, and maturity at age of females was modeled with a logistic equation. Parameters of growth, length-weight conversion, and maturity were estimated by the DW and were treated as input to the assessment model. Females are fully mature at age 2^{+} and the proportion mature at ages 0 and 1 were estimated to be 0.0 and 0.939 respectively.

Spawning stock For Spanish mackerel, peak spawning was considered to occur in June. In cases when reliable estimates of fecundity are unavailable, spawning biomass, and in this case, female weight, is commonly used as a proxy for population fecundity.

Recruitment Recruitment was predicted from spawning biomass using a Beverton–Holt spawner-recruit model. In years when composition data could provide information on year-class strength (1982–2011), estimated recruitment was conditioned on the Beverton–Holt model. In years prior, recruitment followed the Beverton–Holt model precisely (similar to an age-structured production model).

Landings Landings were modeled with the Baranov catch equation (Baranov 1918) and were fitted in units of weight (1000 lb whole weight). The DW provided observed landings back to the first assessment year (1950) for each fleet except general recreational which started in 1955.

Fishing Mortality For each time series of landings and discards the assessment model estimated a separate full fishing mortality rate (F). Age-specific rates were then computed as the product of full F and selectivity at age.

Selectivities Selectivity curves applied to landings and CPUE series were estimated using a parametric approach. This approach applies plausible structure on the shape of the curves, and achieves greater parsimony than occurs with unique parameters for each age. Selectivity of landings from the handline and gill net fleets were modeled as flat-topped, using a two parameter logistic function. Selectivities for the cast net, pound net and recreational fleets were modeled as dome-shaped, with a four parameter double logistic function. Selectivities of the fishery-dependent index was the same as that of their corresponding fishery longline fleet.

Relatively narrow priors were required for most selectivity functions (Normal priors with a CV = 0.15 or CV = 0.25) to reach reasonable values. The pound net fishery was especially difficult to model. All resulting selectivity curves were accepted by the AW panel as reasonable after examination of the size composition data from each fishery. A diffuse prior was used for estimating a parameter of the cast net fishery selectivity function. This prior assumed normal distributions with CV = 1.0 and was intended to provide only weak information to help the optimization routine during model execution. Priors help by steering estimation away from parameter space with no response in the likelihood surface. Without a diffuse priors, it is possible during the optimization search that the parameter could become unimportant, for example if its bounds were set too wide and depending on values of other parameters. When this happens, the likelihood gradient with respect to the aimless parameter approaches zero even if the parameter is not at its globally best value. Diffuse priors help avoid that situation.

Indices of abundance The model was fit to three indices of relative abundance: Florida handline (1986-2011), SEAMAP YOY (1989-2011), and the MRFSS index (1982-2011). Predicted indices were conditional on selectivity of the corresponding fleet or survey and were computed from abundance or biomass (as appropriate) at the midpoint of the year.

Catchability In the BAM, catchability scales indices of relative abundance to estimated population abundance at large. Several options for time-varying catchability were implemented in the BAM following recommendations of the 2009 SEDAR procedural workshop on catchability (SEDAR Procedural Guidance 2009). In particular, the BAM allows for density dependence, linear trends, and random walk, as well as time-invariant catchability. Parameters for these models could be estimated or fixed based on *a priori* considerations. For the base model, the AW assumed time-invariant catchability.

Biological reference points Biological reference points (benchmarks) were calculated based on maximum sustainable yield (MSY) estimates from the Beverton–Holt spawner-recruit model with bias correction (expected values in arithmetic space). The bias correction assumed $\sigma_R = 0.6$, a value supported by Beddington and Cooke (1983) and Mertz and Myers (1996). Computed benchmarks included MSY, fishing mortality rate at MSY (F_{MSY}), and spawning stock at MSY ($\mathrm{SSB}_{\mathrm{MSY}}$). In this assessment, spawning stock measures total weight of mature females. These benchmarks are conditional on the estimated selectivity functions and the relative contributions of each fleet's fishing mortality. The selectivity pattern used here was the effort-weighted selectivities at age, with effort from each fishery estimated as the full F averaged over the last three years of the assessment.

Fitting criterion The fitting criterion was a penalized likelihood approach in which observed landings were fit closely, and observed composition data and abundance indices were fit to the degree that they were compatible. Landings and index data were fitted using lognormal likelihoods. Age composition data were fitted using robust multinomial likelihoods which incorporates the effect of the sample size on the multinomial.

The model includes the capability for each component of the likelihood to be weighted by user-supplied values (for instance, to give more influence to stronger data sources). For data components, these weights were applied by either scaling the likelihood components or adjusting effective sample sizes (multinomial components). In this application to Spanish mackerel, CVs of landings and discards (in arithmetic space) were assumed equal to 0.05, to achieve a close fit to these time series yet allowing some imprecision. In practice, the small CVs are a matter of computational convenience, as they help achieve the desired result of close fits to the landings, while avoiding having to solve the Baranov equation iteratively (which is complex when there are multiple fisheries). Weights on other data components (indices, age compositions) were adjusted iteratively, starting from initial weights as follows. The CVs of indices were

set equal to the values estimated by the DW. Effective sample sizes of the multinomial components were assumed equal to the number of fish sampled annually, as the number of trips was unavailable. The initial likelihood weights were set to 0 and then adjusted until standard deviations of normalized residuals were near 1.0 (SEDAR28-RW04) (Francis 2011). The final weights and their corresponding SDNRs are listed in SEDAR28-04.

In addition, the compound objective function included several penalties or prior distributions, applied to selectivity parameters. Penalties or priors were applied to maintain parameter estimates near reasonable values, and to prevent the optimization routine from drifting into parameter space with negligible gradient in the likelihood.

Model testing Experiments with a reduced model structure indicated that parameters estimated from the BAM were unbiased and could be recovered from simulated data. Further, the general model structure has been through multiple SEDAR reviews. As an additional measure of quality control, input data were examined for accuracy by multiple analysts. This combination of testing and verification procedures suggest that the assessment model is implemented correctly and can provide an accurate assessment of Spanish mackerel stock dynamics.

References

Baranov, F. I. 1918. On the question of the biological basis of fisheries. Nauchnye Issledovaniya Ikhtiologicheskii Instituta Izvestiya **1**:81–128.

Beddington, J. R., and J. G. Cooke, 1983. The potential yield of fish stocks. FAO Fish. Tech. Pap. 242, 47 p.

Francis, R. 2011. Data weighting in statistical fisheries stock assessment models. Canadian Journal of Fisheries and Aquatic Sciences **68**:1124–1138.

Hewitt, D. A., and J. M. Hoenig. 2005. Comparison of two approaches for estimating natural mortality based on longevity. Fishery Bulletin **103**:433–437.

Hoenig, J. M. 1983. Empirical use of longevity data to estimate mortality rates. Fishery Bulletin **81**:898–903.

Lorenzen, K. 1996. The relationship between body weight and natural mortality in juvenile and adult fish: a comparison of natural ecosystems and aquaculture. Journal of Fish Biology **49**:627–642.

Mertz, G., and R. Myers. 1996. Influence of fecundity on recruitment variability of marine fish. Canadian Journal of Fisheries and Aquatic Sciences **53**:1618–1625.

SEDAR Procedural Guidance, 2009. SEDAR Procedural Guidance Document 2: Addressing Time-Varying Catchability.

Table 2.1. General definitions, input data, population model, and negative log-likelihood components of the statistical catch-at-age model. Hat notation ($\widehat{*}$) indicates parameters estimated by the assessment model, and breve notation ($\breve{*}$) indicates estimated quantities whose fit to data forms the objective function.

Quantity	Symbol	Description or definition
General Definitions		
Index of years	y	$y \in \{1950 \dots 2011\}$
Index of ages	a	$a \in \{0 \dots A\}$, where $A = 10^+$
Index of fisheries	f	$f \in \{1 \dots 5\}$ where 1=commercial handlines, 2=commercial pound net, 3=commercial gill net, 4=commercial cast net, and 5=general recreational
Index of CPUE	u	$u = 1, 2, 3$ where 1 = Florida handlines index, 2 = MRFSS, 3 =SEAMAP YOY,
Input Data		
Proportion female at age	$\rho_{a,y}$	0.5 for $a = 0$; $N^F_{a,y}/N_{a,y}$ otherwise
Proportion females mature at age	m_a	
Observed age compositions	$p^{\alpha}_{f,a,y}$	Proportional contribution of age class a in year y to fishery f
Age comp. sample sizes	$n^{\alpha}_{f,y}$	Number of age samples collected in year y from fishery f
Observed fishery landings	$L_{f,y}$	Reported landings in year y from fishery f (in numbers for recreational, whole weight for all others)
CVs of landings	$c^L_{f,y}$	Set to 0.05 for all landings since the goal was to fit landings closely.
Observed abundance indices	$U_{u,y}$	$u = 1$, FL handlines, $y \in \{1986 \dots 2011\}$ $u = 2$, MRFSS, $y \in \{1982 \dots 2011\}$ $u = 3$, SEAMAP YOY, $y \in \{1989 \dots 2011\}$
CVs of abundance indices	$c^U_{u,y}$	$u = 1, 2, 3$ as above. Annual values estimated from delta-lognormal GLMs for all indices.
Natural mortality rate	M_a	Function of combined-sex weight at age (w_a): $M_a = \alpha w_a^{\beta}$, with estimates of α and β from Lorenzen (1996). Lorenzen M_a then rescaled based on Hoenig estimate.
Observed total discards	$D'_{f,y}$	Discards (Numbers of fish) in year y from fishery $f = 5$.
Discard mortality rate	δ_f	Proportion discards by fishery f that die. Value from the DW was 0.2 for the recreational fishery.
CV of dead discards	$c^D_{f,y}$	Set to 0.05 for model fitting
Bycatch	K_y	Spanish mackerel bycatch in the shrimp fishery in year y (numbers of fish).
CV of bycatch	c^B_y	Set to 0.05 for model fitting
Male selectivity lag	a_s	Year increment by which males lag behind females in terms of growth. The estimate of 0.2 was applied in all cases.
Population Model		

Table 2.1. (continued)

Quantity	**Symbol**	**Description or definition**
Mean length at age	$l_{a,s}$	vector of fork length at age by sex (1=males, 2=females, and 3=both: $l_{a,1} = [129.09, 300.39, 398.25, 454.14, 486.07, 504.31, 514.72, 520.67, 524.07, 526.01, 527.12]$ $l_{a,2} = [120.81, 298.11, 414.61, 491.15, 541.45, 574.49, 596.20, 610.47, 619.84, 626.00, 630.05]$ $l_{a,3}(fished\,population) = [276.87, 360.88, 421.88, 466.18, 498.35, 521.71, 538.67, 550.98, 559.93, 566.42, 571.14]$
Individual weight at age	$w_{a,s}$	Computed from length at age by $w_{a,s} = \theta_1 l_{a,s}^{\theta_2}$ where θ_1 and θ_2 are parameters estimated by the DW
Fishery selectivity	$s_{f,a,s,y}$	$= \begin{cases} \frac{1}{1+\exp[-\widehat{\eta}_{1,f,y}(b-\widehat{\alpha}_{1,f,y})]} & : \text{for } f = 1,3 \\ \left(\frac{1}{\max s_{f,a,s,y}}\right)\left(\frac{1}{1+\exp[-\widehat{\eta}_{1,f,y}(b-\widehat{\alpha}_{1,f,y})]}\right) & \\ \quad \left(1 - \frac{1}{1+\exp[-\widehat{\eta}_{2,f,y}(b-[\widehat{\alpha}_{1,f,y}+\widehat{\alpha}_{2,f,y}])]}\right) & : \text{for } f = 2,4,5 \end{cases}$ where $\widehat{\eta}_{1,f,y}$, $\widehat{\eta}_{2,f,y}$, $\widehat{\alpha}_{1,f,y}$, and $\widehat{\alpha}_{2,f,y}$ are fishery-specific parameters, and $b = a$ for females and $b = a - a_s$ for males. Note that all parameters were assumed constant over time. Curves were rescaled, if necessary, to have a maximum of one.
Discard selectivity	$s'_{f,a,s}$	Selectivity at age vectors for different fisheries (subscript f) and sexes (1 = females, 2 = males) $f = 5$, $s = 1$: [1.00, 0.34, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0] $f = 5$, $s = 2$: [1.00, 0.41, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
Bycatch selectivity	$s'_{shrimp,a}$	[1.00, 0.2, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
pre-assessment fishing mortality	F_{hist}	Fishing mortality used to initialize population model in first year of the model. Set to 0.2 for the base run, varied in sensitivity runs
pre-assessment selectivity	s_{hist}	Selectivity applied to females to set initial equilibrium population size and structure. Set to [0.05, 0.5, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
Fishing mortality rate of landings	$F_{f,a,s,y}$	$F_{f,a,s,y} = s_{f,a,s,y}\widehat{F}_{f,y}$ where $\widehat{F}_{f,y}$ is an estimated fully selected fishing mortality rate by fishery
Fishing mortality rate of discards	$F^D_{f,a,s,y}$	$F^D_{f,a,s,y} = s'_{f,a,s}\widehat{F}^D_{f,y}$ where $\widehat{F}^D_{f,y}$ is an estimated fully selected fishing mortality rate of discards by fishery
Fishing mortality rate of bycatch	$F^B_{a,y}$	$F^B_{a,y} = s'_{shrimp,a}\widehat{F}^B_y$ Fishing mortality rate of fish age a in year y associated with shrimp fishery
Total fishing mortality rate	F_y	$F_y = \sum_f \left(\widehat{F}_{f,y} + \widehat{F}^D_{f,y}\right) + \widehat{F}^B_y$
Total mortality rate	$Z_{a,y,s}$	$Z_{a,y,s} = M_a + \sum_{f=1}^{5} F_{f,a,s,y} + F^D_{f=5,a,s,y} + F^B_{a,y}$

Table 2.1. (continued)

Quantity	**Symbol**	**Description or definition**
Abundance at age, sex	$N_{a,y}$, $N_{a,y,s}$	Represent pooled sex and sex-specific abundance, respectively. $N_{0,1950} = \frac{\widehat{R}_0[\varsigma 0.8h\Phi_F - 0.2(1-h)]}{(h-0.2)\Phi_F}$ $N_{0,1950,s} = [(2-s)\rho_{a,y} - (1-s)(1-\rho_{a,y})]\, N_{0,1950}$ $N_{a+1,1950,s} = N_{a,1950,s}\exp(-Z_{a,1950,s}) \quad \forall a \in (0 \dots A-1)$ $N_{A,1950,s} = N_{A-1,1950,s}\frac{\exp(-Z_{A,1950,s})}{1-\exp(-Z_{A,1950,s})}$ $N_{0,y+1} = \begin{cases} \frac{0.8\widehat{R}_0 h S_y}{0.2\phi_0\widehat{R}_0(1-h)+(h-0.2)S_y}\varsigma & \text{for } y+1 < 1982 \\ \frac{0.8\widehat{R}_0 h S_y}{0.2\phi_0\widehat{R}_0(1-h)+(h-0.2)S_y}\exp(\widehat{R}_{y+1}) & \text{for } y+1 \geq 1982 \end{cases}$ $N_{0,y+1,s} = [(2-s)\rho_{a,y} - (1-s)(1-\rho_{a,y})]\, N_{0,y+1}$ $N_{a+1,y+1,s} = N_{a,y,s}\exp(-Z_{a,y,s}) \quad \forall a \in (0 \dots A-1)$ $N_{A,y,s} = N_{A-1,y-1,s}\frac{\exp(-Z_{A-1,y-1,s})}{1-\exp(-Z_{A,y-1},s)}$ where 1950 is the initialization year and ϕ_{hist} gives spawning stock biomass per recruit at the assumed pre-assessment fishing level. Parameter $\widehat{R}_0$ (unfished recruitment) is an estimated parameter of the spawner-recruit curve, and $\widehat{R}_y$ are estimated annual recruitment deviations in log space for $y \geq 1982$ and are zero otherwise. Steepness, h, is fixed in the base model at 0.75. The bias correction is $\varsigma = \exp(\sigma^2/2)$, where σ^2 is the variance of recruitment deviations during 1982–2011. Quantities ϕ_0 and S_y are described below.
Abundance at age (mid-year)	$N'_{a,y}$	Used to match indices of abundance $N'_{a,y,s} = N_{a,y,s}\exp(-Z_{a,y,s}/2)$
Abundance at age at time of spawning	$N''_{a,y}$	Assumed mid-year $N''_{a,y,s} = N'_{a,y,s}$
Unfished abundance at age per recruit at time of spawning	NPR_a	$NPR_0 = \exp(-M_0/2)$ $NPR_{a+1} = NPR_a\exp[-(M_a + M_{a+1})/2] \quad \forall a \in (0 \dots A-1)$ $NPR_A = \frac{NPR_{A-1}\exp[-(M_{A-1}+M_A)/2]}{1-\exp(-M_A)}$
Unfished mature biomass per recruit	ϕ_0	$\phi_0 = \sum_a NPR_a w_{a,1}\rho_{a,y}m_a$
Mature biomass	S_y	$S_y = \sum_a N''_{a,y} w_a \rho_{a,y} m_a$ Also referred to as spawning stock biomass (SSB)
Population biomass	B_y	$B_y = \sum_a \sum_s N_{a,y,s} w_{a,s}$
Landed catch at age, sex	$C_{f,a,y,s}$	$C_{f,a,y,s} = \frac{F_{f,a,y,s}}{Z_{a,y,s}} N_{a,y,s}[1 - \exp(-Z_{a,y,s})]$
Discard mortalities at age, sex	$C^D_{f,a,y,s}$	$C^D_{f,a,y,s} = \frac{F^D_{f,a,y,s}}{Z_{a,y,s}} N_{a,y,s}[1 - \exp(-Z_{a,y,s})]$
Bycatch at age, sex	$C^B_{a,y,s}$	$C^B_{a,y,s} = \frac{F^B_{a,y,s}}{Z_{a,y,s}} N_{a,y,s}[1 - \exp(-Z_{a,y,s})]$
Predicted landings in wgt	$\breve{L}_{f,y}$	$\breve{L}_{f,y} = \sum_a \sum_s C_{f,a,y,s} w_{a,s}$ for $f = 1, 2, 3, 4$
Predicted landings in numbers	$\breve{L}_{f,y}$	$\breve{L}_{f,y} = \sum_a \sum_s C_{f,a,y,s}$ for $f = 5$
Predicted discard mortalities	$\breve{D}_{f,y}$	$\breve{D}_{f,y} = \sum_a \sum_s C^D_{f,a,y_s}$
Predicted shrimp bycatch	$\breve{K}_y$	$\breve{K}_y = \sum_a C^B_{a,y}$

Table 2.1. (continued)

Quantity	**Symbol**	**Description or definition**
Predicted age compositions	$\breve{p}^{\alpha}_{f,a,y}$	$\breve{p}^{\alpha}_{f,a,y} = \frac{\sum_a \sum_s C_{f,a,y,s}}{\sum_a \sum_s C_{f,a,y,s}}$
Predicted CPUE	$\breve{U}_{u,y}$	$\breve{U}_{u,y} = \widehat{q}_u \sum_a \sum_s N'_{a,y,s} s_{u,a,y,s}$ where $\widehat{q}_u$ is the estimated catchability coefficient of index u and $s_{u,a,y}$ is the selectivity of the relevant fishery. For SEAMAP trawl survey, the YOY index is assumed to have $s_{u,a,y} = 1$ for $a = 0$ and $s_{u,a,y} = 0$ otherwise.
Objective Function		
Robust multinomial age compositions	Λ_1	$\Lambda_1 = \sum_{f,u} \sum_y 0.5 \log(E') - \log\left[\exp\left(-\frac{(p^{\alpha}_{(f,u),a,y} - \breve{p}^{\alpha}_{(f,u),a,y})^2}{2E' / (n^{\alpha}_{(f,u),y} \omega^{\alpha}_{(f,u)})}\right) + x\right]$ where $E' = \left[(1 - p^{\alpha}_{(f,u),a,y})(p^{\alpha}_{(f,u),a,y}) + \frac{0.1}{mbin}\right]$, $mbin$ is the number of age bins, $\omega^{\alpha}_{(f,u)}$ is a preset weight (selected by iterative re-weighting) and x =1e-5 is an arbitrary value to avoid log zero. The robust multinomial was used for all fisheries except pound net.
Multinomial age compositions	Λ_2	$\Lambda_2 = -\omega_2 \sum_f \sum_y \left[n^{\alpha}_{f,y} \sum_a (p^{\alpha}_{f,a,y} + x) \log\left(\frac{(\breve{p}^{\alpha}_{f,a,y} + x)}{(p^{\alpha}_{f,a,y} + x)}\right)\right]$ where ω_2 is as determined by iterative reweighting and x =1e-5 is an arbitrary value to avoid log zero. The denominator of the log is a scaling term. The multinomial likelihood was only used for the pound net fishery.
Lognormal landings	Λ_3	$\Lambda_3 = \omega_3 \sum_f \sum_y \frac{\left[\log\left((L_{f,y}+x)/(\breve{L}_{f,y}+x)\right)\right]^2}{2(c^L_{f,y})^2}$ where $\omega_3 = 1$ is a preset weight and x =1e-5 is an arbitrary value to avoid log zero or division by zero
Lognormal discard mortalities	Λ_4	$\Lambda_4 = \omega_4 \sum_f \sum_y \frac{\left[\log\left((\delta_f D_{f,y}+x)/(\breve{D}_{f,y}+x)\right)\right]^2}{2(c^D_{f,y})^2}$ for $f = 5$ where $\omega_4 = 1$ is a preset weight and x =1e-5 is an arbitrary value to avoid log zero or division by zero
Lognormal Bycatch	Λ_5	$\Lambda_5 = \omega_5 \sum_f \sum_y \frac{\left[\log\left((K_y+x)/(\breve{K}_y+x)\right)\right]^2}{2(c^B_y)^2}$ where $\omega_5 = 1$ is a preset weight and x =1e-5 is an arbitrary value to avoid log zero or division by zero
Lognormal CPUE	Λ_6	$\Lambda_6 = \sum_{u=1}^{2} \omega_6 \sum_y \frac{\left[\log\left((U_{u,y}+x)/(\breve{U}_{u,y}+x)\right)\right]^2}{2(c^U_{u,y})^2}$ where ω_6 is a preset weight is a preset weight for each index determined by iterative reweighting and x =1e-5 is an arbitrary value to avoid log zero or division by zero.
Constraint on recruitment deviations	Λ_7	$\Lambda_7 = \omega_7 \left[R^2_{1982} + \sum_{y>1982} (R_y - \widehat{\varrho} R_{y-1})^2 \right]$ where R_y are recruitment deviations in log space, $\omega_7 = 1.0$ is a preset weight and $\widehat{\varrho}$ is set to 0 for the base run.
Additional constraint on recruitment deviations	Λ_8	$\Lambda_8 = \omega_8 \left(\sum_{y \geq 2009} R^2_y \right)$ where $\omega_8 = 1$ is a preset weight

Table 2.1. (continued)

Quantity	**Symbol**	**Description or definition**
Constraint on F_y	Λ_9	$\Lambda_9 = \omega_9 \sum_y I_y (F_y - \Psi)^2$ where $\omega_9 = 1$ is a preset weight, $\Psi = 3.0$ is the max unconstrained F_y, and $I_y = \begin{cases} 1 & : \text{if } F_y > \Psi \\ 0 & : \text{otherwise} \end{cases}$
Total objective function	Λ	$\Lambda = \sum_{i=1}^{9} \Lambda_i$ Objective function minimized by the assessment model

Appendix A AD Model Builder code to implement the Beaufort Assessment Model

```
//
//
//  SEDAR 28 Assessment: Spanish mackerel, May 2012
//
//  NMFS, Beaufort Lab, Sustainable Fisheries Branch
//
//


//Inputs selectivity for discards (as known) and estimates different time dep. F's for discards & landings
//Uses interpolated landings instead of interpolated F's
//Uses an input value for historical F rather than setting it = F(1950)
//Inputs historical selectivity as well.


DATA_SECTION

!!cout << "Starting Spanish Mackerel Assessment Model" << endl;

// Starting and ending year of the model (year data starts)
init_int styr;
init_int endyr;
//Starting year to estimate recruitment deviation from S-R curve
init_int styr_rec_dev;

//Total number of ages
init_int nages;

//!!cout << "Number of ages" << nages << endl;

// Vector of ages for age bins
init_vector agebins(1,nages);
//!!cout << "Age bins" << agebins << endl;
//number assessment years
//int styrR;
number nyrs;
number nyrs_rec;
//this section MUST BE INDENTED!!!
 LOCAL_CALCS
   nyrs=endyr-styr+1.;
   nyrs_rec=endyr-styr_rec_dev+1.;
 END_CALCS

//Total number of length bins for each matrix
init_int nlenbins;

// Vector of lengths for length bins (cm)(midpoint)
init_vector lenbins(1,nlenbins);

//discard mortality constants
//init_number set_Dmort_HL;
//init_number set_Dmort_GN;
init_number set_Dmort_MRFSS;

//initial proportion female
init_number set_prop_f_a0;
//!!cout << "proportion female" << set_prop_f_a0 << endl;
//Total number of iterations for spr calcs
init_int n_iter_spr;
//!!cout << "spr iterationsn" << n_iter_spr << endl;
//Total number of iterations for msy calcs
init_int n_iter_msy;
//!!cout << "MSY iterations" << n_iter_msy << endl;
//starting index of ages for exploitation rate: if model has age-0s, ages of E are (value-1) to oldest
init_int set_E_age_st;
//!!cout << "Exploitation age" << set_E_age_st << endl;
//bias correction (set to 1.0 for no bias correction or 0.0 to compute from rec variance)
init_number set_BiasCor;
//!!cout << "Bias correction" << set_BiasCor << endl;
// Von Bert parameters for males and females, respectively
//init_number set_Linf_m;
//init_number set_K_m;
//!!cout<<"Linf males"<< set_Linf_m<<endl;
//init_number set_t0_m;
//init_number set_Linf_f;
//!!cout<<"Linf females"<< set_Linf_f<<endl;
//init_number set_K_f;
//init_number set_t0_f;
//!!cout<<"t0 females"<< set_t0_f<<endl;
//CV of length at age
init_number set_len_cv;

//length(mm)-weight(whole weight in g) relationship: W=aL^b
init_number wgtpar_a;
init_number wgtpar_b;
init_number wgtpar_am;
init_number wgtpar_bm;
init_number wgtpar_af;
init_number wgtpar_bf;

//weight-weight relationship:whole weight to gutted weight -- gutted=a*whole
init_vector meanlen_fishery(1,nages);
init_vector meanlen_m(1,nages);
init_vector meanlen_f(1,nages);
//!!cout<<"mean length fishery"<<meanlen_fishery<<endl;
//!!cout<<"mean length males"<<meanlen_m<<endl;
//!!cout<<"mean length females"<<meanlen_f<<endl;
```

```
//Female maturity and proportion female at age
init_vector maturity_f_obs(1,nages);
!!cout << "maturity" << maturity_f_obs(1,nages) << endl;          //total maturity of females
init_vector prop_f_obs(1,nages);                //proportion female at age
//
//Commercial Hand Lines fishery
//CPUE
//FL trip ticket
init_int styr_FL_HL_cpue;
//  !!cout << "start year of FL HL" << styr_FL_HL_cpue << endl;
init_int endyr_FL_HL_cpue;
init_vector obs_FL_HL_cpue(styr_FL_HL_cpue,endyr_FL_HL_cpue);//Observed CPUE
init_vector FL_HL_cpue_cv(styr_FL_HL_cpue,endyr_FL_HL_cpue); //CV of cpue
//  !!cout << "FL HL cpue cvs " << FL_HL_cpue_cv << endl;
// Landings (1000 lb whole weight)
init_int styr_HL_L;
init_int endyr_HL_L;
init_vector obs_HL_L(styr_HL_L,endyr_HL_L);
init_vector HL_L_cv(styr_HL_L,endyr_HL_L);    //vector of CV of landings by year
//  !!cout << "start year of HL Landings" <<styr_HL_L << endl;
// Discards (1000s)
//init_int styr_HL_D;
//init_int endyr_HL_D;
//init_vector obs_HL_released(styr_HL_D,endyr_HL_D); //vector of observed releases by year, multiplied by discard mortality for fitting
//init_vector HL_D_cv(styr_HL_D,endyr_HL_D);    //vector of CV of discards by year
// Length Compositions (1cm bins)
//init_int nyr_HL_lenc;
//init_ivector yrs_HL_lenc(1,nyr_HL_lenc);
//init_vector nsamp_HL_lenc(1,nyr_HL_lenc);
//init_matrix obs_HL_lenc(1,nyr_HL_lenc,1,nlenbins);
  !!cout << "start year of FL HL" <<styr_FL_HL_cpue << endl;
// Age Compositions
init_int nyr_HL_agec;
init_ivector yrs_HL_agec(1,nyr_HL_agec);
init_vector nsamp_HL_agec(1,nyr_HL_agec);
init_matrix obs_HL_agec(1,nyr_HL_agec,1,nages);
//  !!cout << "Handline data" << obs_HL_agec << endl;
//
//Poundnet fishery
// Landings (1000 lb whole weight)
init_int styr_PN_L;
init_int endyr_PN_L;
init_vector obs_PN_L(styr_PN_L,endyr_PN_L);
init_vector PN_L_cv(styr_PN_L,endyr_PN_L);
// Length Compositions (1cm bins)
//init_int nyr_PN_lenc;
//init_ivector yrs_PN_lenc(1,nyr_PN_lenc);
//init_vector nsamp_PN_lenc(1,nyr_PN_lenc);
//init_matrix obs_PN_lenc(1,nyr_PN_lenc,1,nlenbins);
// Age compositions
init_int nyr_PN_agec;
init_ivector yrs_PN_agec(1,nyr_PN_agec);
init_vector nsamp_PN_agec(1,nyr_PN_agec);
init_matrix obs_PN_agec(1,nyr_PN_agec,1,nages);
//  !!cout << "Poundnet data" <<nyr_PN_agec << endl;
//
//Commercial Gillnet fishery
// Landings - (1000 lb whole weight)
init_int styr_GN_L;
init_int endyr_GN_L;
init_vector obs_GN_L(styr_GN_L,endyr_GN_L); //vector of observed landings by year
init_vector GN_L_cv(styr_GN_L,endyr_GN_L);    //vector of CV of landings by year

// Length Compositions (1cm bins)
//init_int nyr_GN_lenc;
//init_ivector yrs_GN_lenc(1,nyr_GN_lenc);
//init_vector nsamp_GN_lenc(1,nyr_GN_lenc);
//init_matrix obs_GN_lenc(1,nyr_GN_lenc,1,nlenbins);
// Age compositions
init_int nyr_GN_agec;
init_ivector yrs_GN_agec(1,nyr_GN_agec);
init_vector nsamp_GN_agec(1,nyr_GN_agec);
init_matrix obs_GN_agec(1,nyr_GN_agec,1,nages);
  !!cout << "Gillnet data" <<obs_GN_agec << endl;
//
//Castnet fishery
// Landings (1000 lb whole weight)
init_int styr_CN_L;
init_int endyr_CN_L;
init_vector obs_CN_L(styr_CN_L,endyr_CN_L);
init_vector CN_L_cv(styr_CN_L,endyr_CN_L);
// Length Compositions (1cm bins)
//init_int nyr_CN_lenc;
//init_ivector yrs_CN_lenc(1,nyr_CN_lenc);
//init_vector nsamp_CN_lenc(1,nyr_CN_lenc);
//init_matrix obs_CN_lenc(1,nyr_CN_lenc,1,nlenbins);
// Age compositions
init_int nyr_CN_agec;
init_ivector yrs_CN_agec(1,nyr_CN_agec);
init_vector nsamp_CN_agec(1,nyr_CN_agec);
init_matrix obs_CN_agec(1,nyr_CN_agec,1,nages);
//  !!cout << "Castnet data" <<obs_CN_agec << endl;
//
//MRFSS landings
//CPUE
init_int styr_MRFSS_cpue;
init_int endyr_MRFSS_cpue;
init_vector obs_MRFSS_cpue(styr_MRFSS_cpue,endyr_MRFSS_cpue);//Observed CPUE
init_vector MRFSS_cpue_cv(styr_MRFSS_cpue,endyr_MRFSS_cpue); //CV of cpue
//  !!cout << "MRFSS cpue" <<MRFSS_cpue_cv << endl;
// Landings (1000 of fish)
init_int styr_MRFSS_L;
init_int endyr_MRFSS_L;
```

```
init_vector obs_MRFSS_L(styr_MRFSS_L,endyr_MRFSS_L);
init_vector MRFSS_L_cv(styr_MRFSS_L,endyr_MRFSS_L);
  !!cout << "MRFSS Landings cv" <<MRFSS_L_cv << endl;
// Discards (1000s)
init_int styr_MRFSS_D;
init_int endyr_MRFSS_D;
init_vector obs_MRFSS_released(styr_MRFSS_D,endyr_MRFSS_D); //vector of observed releases by year, multiplied by discard mortality for fitting
init_vector MRFSS_D_cv(styr_MRFSS_D,endyr_MRFSS_D);    //vector of CV of discards by year
// Length Compositions (1cm bins)
//init_int styr_MRFSS_lenc;
//init_int endyr_MRFSS_lenc;
//init_int nyr_MRFSS_lenc;
//init_vector nsamp_MRFSS_lenc(styr_MRFSS_lenc,endyr_MRFSS_lenc);
//init_matrix obs_MRFSS_lenc(styr_MRFSS_lenc,endyr_MRFSS_lenc,1,nlenbins);
// !!cout << "MRFSS length comps" <<obs_MRFSS_lenc << endl;
// Age Compositions
init_int styr_MRFSS_agec;
init_int endyr_MRFSS_agec;
init_int nyr_MRFSS_agec;
init_ivector yrs_MRFSS_agec(1,nyr_MRFSS_agec);
init_vector nsamp_MRFSS_agec(1,nyr_MRFSS_agec);
init_matrix obs_MRFSS_agec(1,nyr_MRFSS_agec,1,nages);
//  !!cout << "MRFSS" <<obs_MRFSS_agec << endl;
!!cout<<"nyrs mrfss"<< nyr_MRFSS_agec<<endl;
//
//Shrimp Bycatch
// Bycatch (1000s of fish)
init_int styr_shrimp_B;
init_int endyr_shrimp_B;
init_vector obs_shrimp_B(styr_shrimp_B,endyr_shrimp_B);
init_vector shrimp_B_cv(styr_shrimp_B,endyr_shrimp_B);
//init_vector obs_shrimp_B_effort(styr_shrimp_B,endyr_shrimp_B);
//init_vector shrimp_B_cv(styr_shrimp_B,endyr_shrimp_B);
//init_int styr_shrimp_B_fit;  //start year for computing geometric mean of shrimp bycatch for fitting
//init_int endyr_shrimp_B_fit; //end year for computing geometric mean of shrimp bycatch for fitting

//   !!cout << "Shrimp bycatch" <<shrimp_B_cv << endl;
//
//SEAMAP
//YOY CPUE
init_int styr_SMAP_YOY_cpue;
init_int endyr_SMAP_YOY_cpue;
init_vector obs_SMAP_YOY_cpue(styr_SMAP_YOY_cpue,endyr_SMAP_YOY_cpue);//Observed CPUE
init_vector SMAP_YOY_cpue_cv(styr_SMAP_YOY_cpue,endyr_SMAP_YOY_cpue); //CV of cpue
//1-yr-old CPUE
//init_int styr_SMAP_1YR_cpue;
//init_int endyr_SMAP_1YR_cpue;
//init_vector obs_SMAP_1YR_cpue(styr_SMAP_1YR_cpue,endyr_SMAP_1YR_cpue);//Observed CPUE
//init_vector SMAP_1YR_cpue_cv(styr_SMAP_1YR_cpue,endyr_SMAP_1YR_cpue); //CV of cpue
//   !!cout << "SEAMAP 1-yr" <<obs_SMAP_1YR_cpue << endl;
//
//Parameter values and initial guesses
//weights for likelihood components
init_number set_w_L;
init_number set_w_D;
//init_number set_w_shrimp;
//init_number set_w_lc;
//init_number set_w_lc_HL;
//init_number set_w_lc_PN;
//init_number set_w_lc_GN;
//init_number set_w_lc_CN;
//init_number set_w_lc_MRFSS;
//init_number set_w_ac;
init_number set_w_ac_HL;
init_number set_w_ac_PN;
init_number set_w_ac_GN;
init_number set_w_ac_CN;
init_number set_w_ac_MRFSS;
init_number set_w_I_FL_HL;
init_number set_w_I_MRFSS;
init_number set_w_I_SMAP_YOY;
//init_number set_w_I_SMAP_1YR;
init_number set_w_R;
init_number set_w_R_init;
init_number set_w_R_end;
init_number set_w_F;
init_number set_w_B1dBO;         // weight on B1/BO
init_number set_w_fullF;         //penalty for any fullF>5
init_number set_w_cvlen_dev;         //penalty on cv deviations at age
init_number set_w_cvlen_diff;       //penalty on first difference of cv deviations at age

//Initial guesses or fixed values
init_number set_steep;
init_number set_steep_se;      //SE of recruitment steepness
//init_number set_M;
init_vector set_M(1,nages); //age-dependent: used in model
init_number set_M_constant; //age-independent: used only for MSST
init_number set_rec_sigma;     //recruitment standard deviation in log space
init_number set_rec_sigma_se;  //SE of recruitment standard deviation in log space

//--index catchability-----------------------------------------------------------------------------------------------------

init_number set_logq_FL_HL;         //catchability coefficient (log) for FL hand lines
init_number set_logq_MRFSS;
init_number set_logq_SMAP_YOY;
//init_number set_logq_SMAP_1YR;
//init_number set_logq_shrimp_B;
//--F's-------------------------------
init_number set_F_hist;
  !!cout << "historic F" <<set_F_hist << endl;
init_number set_log_avg_F_HL;  //hand lines
init_vector set_log_F_dev_HL(styr_HL_L,endyr_HL_L);
init_number set_log_avg_F_PN;  //pound nets
init_vector set_log_F_dev_PN(styr_PN_L,endyr_PN_L);
```

```
init_number set_log_avg_F_GN;  //gill nets
  !!cout << "AVG F GN" <<set_log_avg_F_GN << endl;
init_vector set_log_F_dev_GN(styr_GN_L,endyr_GN_L);
init_number set_log_avg_F_CN;  //cast nets
init_vector set_log_F_dev_CN(styr_CN_L,endyr_CN_L);
//  !!cout << "F devs CN" <<set_log_F_dev_CN << endl;
init_number set_log_avg_F_MRFSS;  //mrfss
init_vector set_log_F_dev_MRFSS(styr_MRFSS_L,endyr_MRFSS_L);
init_number set_log_avg_F_MRFSS_D;  //mrfss discards
  !!cout << "AVG F MRFSS_D" <<set_log_avg_F_MRFSS_D << endl;
init_vector set_log_F_dev_MRFSS_D(styr_MRFSS_D,endyr_MRFSS_D);
  !!cout << "F devs MRFSS _D" <<set_log_F_dev_MRFSS_D << endl;
init_number set_log_avg_F_shrimp;  //shrimp
init_vector set_log_F_dev_shrimp(styr_shrimp_B,endyr_shrimp_B);

//Set some more initial guesses of estimated parameters
init_number set_log_R0;
init_number set_R_autocorr;
  !!cout << "R autocorr" <<set_R_autocorr << endl;

//Initial guesses of estimated selectivity parameters
init_number set_selpar_L50_HL_keep;
//  !!cout << "parm L50 for HL" <<set_selpar_L50_HL_keep << endl;
init_number set_selpar_slope_HL;
// !!cout << "parm slope for HL"<<set_selpar_slope_HL<<endl;
init_number set_selpar_L501_PN;
// !!cout<<"PN a501"<<set_selpar_L501_PN<<endl;
init_number set_selpar_slope1_PN;
// !!cout<<"PN slope1"<<set_selpar_slope1_PN<<endl;
init_number set_selpar_L502_PN;
// !!cout<<"PN a502"<<set_selpar_L502_PN<<endl;
init_number set_selpar_slope2_PN;
// !!cout<<"PN slope2"<<set_selpar_slope2_PN<<endl;
init_number set_selpar_L50_GN_keep;
//!!cout<<"GN a50"<<set_selpar_L50_GN_keep<<endl;
init_number set_selpar_slope_GN;
init_number set_selpar_L501_CN;
init_number set_selpar_slope1_CN;
init_number set_selpar_L502_CN;
init_number set_selpar_slope2_CN;
//init_number set_selpar_sigma_CN;
//init_number set_selpar_afull_CN;
init_number set_selpar_L501_MRFSS_keep;
init_number set_selpar_slope1_MRFSS_keep;
init_number set_selpar_L502_MRFSS_keep;
init_number set_selpar_slope2_MRFSS_keep;
//init_number set_selpar_sigma_MRFSS;
//init_number set_selpar_afull_MRFSS;
init_vector set_sel_historical_F(1,nages);
//   !!cout << "historical F selex" <<set_sel_historical_F << endl;
init_vector set_sel_MRFSS_D_F(1,nages);  //selectivity vectors for female, male discards input directly
init_vector set_sel_MRFSS_D_M(1,nages);
init_vector set_sel_shrimp_B(1,nages); //shrimp bycatch selectivity input directly
//  !!cout << "shrimp bycatch selex" <<set_sel_shrimp_B << endl;

init_number set_L50_diff; //difference between males and females

//aging error matrix
init_matrix set_age_error_matrix(1,nages,1,nages);

//historic recreational landings multiplier
init_number L_rec_multiply;

//threshold sample sizes for length comps
//init_number minSS_HL_lenc;
//init_number minSS_PN_lenc;
//init_number minSS_GN_lenc;
//init_number minSS_CN_lenc;
//init_number minSS_MRFSS_lenc;
 //cout << "min SS MRFSS" <<minSS_MRFSS_lenc<< endl;
//threshold sample sizes for age comps
init_number minSS_HL_agec;
init_number minSS_PN_agec;
init_number minSS_GN_agec;
init_number minSS_CN_agec;
init_number minSS_MRFSS_agec;
// !!cout<<"minSS for MRFSS"<<minSS_MRFSS_agec<<endl;

// Indices for year(iyear), age(iage),length(ilen)
int iyear;
int iyear2;
int iage;
int ilen;
int E_age_st;   //starting age for exploitation rate: (value-1) to oldest
int ff;

init_number end_of_data_file;
//this section MUST BE INDENTED!!!
 LOCAL_CALCS
   if(end_of_data_file!=999)
   {
       cout << "*** WARNING: Data File NOT READ CORRECTLY ****" << endl;
       cout << "" <<endl;
       exit(0);

   }
   else
   {
    cout << "Data File read correctly" << endl;
   }
 END_CALCS

PARAMETER_SECTION
//number Linf_m;
```

```
//number K_m;
//number t0_m;
//number Linf_f;
//number K_f;
//number t0_f;
//init_bounded_number Linf_m(300,1000,-3);
//init_bounded_number K_m(0.15,1.0,-3);
//init_bounded_number t0_m(-5,0,-3);
//init_bounded_number Linf_f(300,1000,-3);
//init_bounded_number K_f(0.15,1.0,-3);
//init_bounded_number t0_f(-5,0,-3);


//vector meanlen_fishery(1,nages);
//vector meanlen_m(1,nages);
//vector meanlen_f(1,nages);

vector wgt_g_f(1,nages);        //whole wgt in g - females
vector wgt_kg_f(1,nages);       //whole wgt in kg
vector wgt_f(1,nages);          //whole wgt in mt
vector wgt_klb_f(1,nages); //whole wgt in 1000 lb
vector wgt_g_land(1,nages);        //whole wgt in g - females
vector wgt_kg_land(1,nages);       //whole wgt in kg
vector wgt_land(1,nages);          //whole wgt in mt
vector wgt_klb_land(1,nages); //whole wgt in 1000 lb
vector wgt_g_m(1,nages);        //whole wgt in g - males
vector wgt_kg_m(1,nages);       //whole wgt in kg
vector wgt_m(1,nages); //whole wgt in mt
vector wgt_klb_m(1,nages);      //whole wgt in 1000 lb
//vector meanlen_m(1,nages);        //mean length at age -males
//vector meanlen_f(1,nages);        //mean length at age -females
number sqrt2pi;
number g2mt;                    //conversion of grams to metric tons
number g2kg; //conversion of grams to kg
number mt2klb;  //conversion of metric tons to 1000 lb

number nlenbins2;
  matrix lenprob_m(1,nages,1,nlenbins);          //distn of size at age (age-length key, 1cm bins) - males
  matrix lenprob_f(1,nages,1,nlenbins);          //distn of size at age (age-length key, 1cm bins) - females
  matrix lenprob_m2(1,nages,1,nlenbins+20);          //distn of size at age (age-length key, 1cm bins) - males
  matrix lenprob_f2(1,nages,1,nlenbins+20);          //distn of size at age (age-length key, 1cm bins) - females
  vector lenbins2(1,nlenbins+20);

  init_bounded_number log_len_cv(-5,-0.3,-4);  //KWS add a fourth phase
  //init_bounded_number log_len_cv(-4.6,-0.7,2) //cv expressed in log-space, bounds correspond to 0.01, 0.5
  //init_bounded_dev_vector log_len_cv_dev(1,nages,-2,2,3)
  vector len_cv(1,nages);

//----Predicted length and age compositions
  number prop_f_a0; //proportion female at age 0

  //matrix pred_HL_lenc(1,nyr_HL_lenc,1,nlenbins);
  //matrix pred_PN_lenc(1,nyr_PN_lenc,1,nlenbins);
  //matrix pred_GN_lenc(1,nyr_GN_lenc,1,nlenbins);
  //matrix pred_CN_lenc(1,nyr_CN_lenc,1,nlenbins);
  //matrix pred_MRFSS_lenc(styr_MRFSS_lenc,endyr_MRFSS_lenc,1,nlenbins);

  matrix pred_HL_agec(1,nyr_HL_agec,1,nages);
  matrix pred_PN_agec(1,nyr_PN_agec,1,nages);
  matrix pred_GN_agec(1,nyr_GN_agec,1,nages);
  matrix pred_CN_agec(1,nyr_CN_agec,1,nages);
  matrix pred_MRFSS_agec(1,nyr_MRFSS_agec,1,nages);

  //nsamp_X_allyr vectors used only for R output of comps with nonconsecutive yrs
  //vector nsamp_HL_lenc_allyr(styr,endyr);
  vector nsamp_HL_agec_allyr(styr,endyr);
  //vector nsamp_PN_lenc_allyr(styr,endyr);
  vector nsamp_PN_agec_allyr(styr,endyr);
  //vector nsamp_GN_lenc_allyr(styr,endyr);
  vector nsamp_GN_agec_allyr(styr,endyr);
  //vector nsamp_CN_lenc_allyr(styr,endyr);
  vector nsamp_CN_agec_allyr(styr,endyr);
  //vector nsamp_MRFSS_lenc_allyr(styr,endyr);
  vector nsamp_MRFSS_agec_allyr(styr,endyr);

//----Aging error
  matrix age_error_matrix(1,nages,1,nages);

//-----Population---------------------------------------------------------------------------
  matrix N_F(styr,endyr+1,1,nages);           //Population numbers for females by year and age at start of yr
  matrix N_M(styr,endyr+1,1,nages);           //Population numbers for males by year and age at start of yr
  matrix N_F_mdyr(styr,endyr+1,1,nages);        //Population numbers by year and age at mdpt of yr: used for comps and SSB
  matrix N_M_mdyr(styr,endyr+1,1,nages);
  matrix B(styr,endyr+1,1,nages);             //Biomass by year and age - sexes combined
  vector totB(styr,endyr+1);                //Total biomass by year
  vector SSB(styr,endyr);       //Spawning biomass by year
  number SSB_extra;
  vector rec(styr,endyr+1);       //Recruits by year
  matrix prop_f(styr,endyr+1,1,nages);        //Proportion female by year and age
  vector prop_f_F0(1,nages);                //proportion of females in unexploited pop
  vector maturity_f(1,nages);               //Proportion of female mature at age
  vector reprod(1,nages);                 //recruitment calcs

//---Stock-Recruit Function (Beverton-Holt, steepness parameterization)----------
  init_bounded_number log_R0(5,100,1);       //log(virgin Recruitment)
  //number log_R0;
  number R0;
  init_bounded_number steep(0.25,0.9,-3);     //steepness
  //number steep;  //uncomment to fix steepness, comment line directly above
  init_bounded_dev_vector log_rec_dev(styr_rec_dev,endyr,-5,5,3); //log recruitment deviations
  //vector log_dev_N_rec(styr_rec_dev,endyr);
  vector log_dev_R(styr,endyr+1);           //used in output. equals zero except for yrs in log_dev_N_rec
  number var_rec_dev;
  number sigma_rec_dev;                     //variance of log recruitment deviations.
```

```
  init_bounded_number rec_sigma(0.5,1.2,-4);  //sd recruitment residuals
  number rec_sigma_sq;                        //square of rec_sigma
  number rec_logL_add;                        //additive term in -logL term
                                                 //Estimate from yrs with unconstrainted S-R(XXXX-XXXX)
  number BiasCor;                            //Bias correction in equilibrium recruits
  init_bounded_number R_autocorr(0,1.0,-3);  //autocorrelation in SR  //KWS turned off
  //number R_autocorr;
  number R_autocorr_sd;
  number steep_sd;                 //steepness for stdev report
  number S0;                               //equal to spr_F0*R0 = virgin SSB - for males = 0
  number B0;                              //equal to bpr_F0*R0 = virgin Biomass (combined sex)
  number R1;                               //Recruits in styr

  number S1S0;                     //SSB(styr) / virgin SSB
  number popstatus;                //SSB(endyr) / virgin SSB

//---Selectivity-------------------------------------------------------------------------

  number L50_diff; //shift in selectivity b/w males and females (same for all gears)
  vector sel_historical_F(1,nages);

   //Commercial handline
  vector sel_HL_keep_M(1,nages);  //time invariant - fish that are kept - males
  vector sel_HL_keep_F(1,nages);  //time invariant - fish that are kept - females
  init_bounded_number selpar_slope_HL(0.1,10,3);
  init_bounded_number selpar_L50_HL_keep(0.5,8,2);

  //Poundnets
  vector sel_PN_M(1,nages);  //time invariant - males
  vector sel_PN_F(1,nages);  //females
  init_bounded_number selpar_slope1_PN(0.1,10,3);
  init_bounded_number selpar_L501_PN(0.4,8,2);
  init_bounded_number selpar_slope2_PN(0.1,10,3);
  init_bounded_number selpar_L502_PN(-2.0,8,2);

  //Commercial gillnet
  vector sel_GN_keep_M(1,nages);  //pre-1995 - fish that are kept - males
  vector sel_GN_keep_F(1,nages);  //pre-1995 - fish that are kept - females
  init_bounded_number selpar_slope_GN(0.5,10.0,3);
  init_bounded_number selpar_L50_GN_keep(1.0,8,2);

  //Castnets
  vector sel_CN_M(1,nages);  //time invariant - males
  vector sel_CN_F(1,nages);  //females
  init_bounded_number selpar_slope1_CN(0.5,10.0,3);
  init_bounded_number selpar_L501_CN(2.0,8,2);
  init_bounded_number selpar_slope2_CN(0.5,10.0,3);
  init_bounded_number selpar_L502_CN(2.0,10.0,2);
  //init_bounded_number selpar_sigma_CN(0.1,10000.0,-3);
  //number selpar_afull_CN;

  //MRFSS
  vector sel_MRFSS_keep_M(1,nages);  //time invariant - fish that are kept - males
  vector sel_MRFSS_keep_F(1,nages);  //time invariant - fish that are kept - females
  vector sel_MRFSS_D_M(1,nages);  //selectivity for discards - males
  vector sel_MRFSS_D_F(1,nages);  //selectivity for discards - females
  init_bounded_number selpar_slope1_MRFSS_keep(0.1,15,3);
  init_bounded_number selpar_L501_MRFSS_keep(0.5,8,2);
init_bounded_number selpar_slope2_MRFSS_keep(0.1,15,3);
init_bounded_number selpar_L502_MRFSS_keep(-2.0,8,2);
  //init_bounded_number selpar_sigma_MRFSS(0.1,10000.0,3);
  //number selpar_afull_MRFSS;

  //shrimp
  vector sel_shrimp(1,nages);

  //effort-weighted, recent selectivities
  vector sel_wgted_L_F(1,nages);  //toward landings,females
  vector sel_wgted_D_F(1,nages);  //toward discards
  vector sel_wgted_tot_F(1,nages);//toward Z, landings plus dead discards
  number max_sel_wgted_tot_F;
  vector sel_wgted_L_M(1,nages);  //toward landings,males
  vector sel_wgted_D_M(1,nages);  //toward discards
  vector sel_wgted_tot_M(1,nages);//toward Z, landings plus dead discards
  number max_sel_wgted_tot_M;

//-------CPUE Predictions--------------------------------
  vector pred_FL_HL_cpue(styr_FL_HL_cpue,endyr_FL_HL_cpue);            //predicted FL handline index (pounds/trip)
  matrix N_FL_HL(styr_FL_HL_cpue,endyr_FL_HL_cpue,1,nages);            //used to compute above index
  vector pred_MRFSS_cpue(styr_MRFSS_cpue,endyr_MRFSS_cpue);            //predicted MRFSS index (number per angler-trip)
  matrix N_MRFSS(styr_MRFSS_cpue,endyr_MRFSS_cpue,1,nages);            //used to compute above index
  vector pred_SMAP_YOY_cpue(styr_SMAP_YOY_cpue,endyr_SMAP_YOY_cpue);            //predicted SMAP_YOY index (number per tow)
  //vector pred_SMAP_1YR_cpue(styr_SMAP_1YR_cpue,endyr_SMAP_1YR_cpue);            //predicted SMAP_1YR index (number per tow)

//---Catchability (CPUE q's)-------------------------------------------------------------

  init_bounded_number log_q_FL_HL(-20,-5,1);
  init_bounded_number log_q_MRFSS(-40,-5,1);  //KWs adjusted lower bound
  init_bounded_number log_q_SMAP_YOY(-40,-5,1);
  //init_bounded_number log_q_SMAP_1YR(-40,-5,1);
  //init_bounded_number log_q_shrimp_B(-30,-5,1);

  //init_bounded_number q_rate(-0.1,0.1,-3);
  number q_rate;

//---Landings Bias-----------------------------------------------------------------------
  //init_bounded_number L_early_bias(0.1,10.0,3);
  //number L_early_bias;
  //number L_commHAL_bias;

//---C is landings in (numbers), L is landings in wgt (mt)-------------------------------------------------

  matrix C_HL_M(styr,endyr,1,nages);              //landings (numbers) at age
  matrix C_HL_F(styr,endyr,1,nages);              //landings (numbers) at age
```

```
  matrix L_HL(styr,endyr,1,nages);                //landings (mt) at age
  matrix L_HL_klb(styr,endyr,1,nages);            //landings (1000 lb whole weight) at age
  vector pred_HL_L(styr_HL_L,endyr_HL_L);         //yearly landings (klb) summed over ages

  matrix C_PN_M(styr,endyr,1,nages);              //landings (numbers) at age
  matrix C_PN_F(styr,endyr,1,nages);              //landings (numbers) at age
  matrix L_PN(styr,endyr,1,nages);                //landings (mt) at age
  matrix L_PN_klb(styr,endyr,1,nages);            //landings (1000 lb whole weight) at age
  vector pred_PN_L(styr_PN_L,endyr_PN_L);         //yearly landings (klb) summed over ages

  matrix C_GN_M(styr,endyr,1,nages);              //landings (numbers) at age
  matrix C_GN_F(styr,endyr,1,nages);              //landings (numbers) at age
  matrix L_GN(styr,endyr,1,nages);                //landings (mt) at age
  matrix L_GN_klb(styr,endyr,1,nages);            //landings (1000 lb whole weight) at age
  vector pred_GN_L(styr_GN_L,endyr_GN_L);         //yearly landings (klb) summed over ages

  matrix C_total(styr,endyr,1,nages);             //catch in number
  matrix L_total(styr,endyr,1,nages);             //landings in klb
  vector L_total_yr(styr,endyr);                  //total landings (klb) by yr summed over ages
  matrix D_total(styr,endyr,1,nages);             //discards in klb
  vector D_total_yr(styr,endyr);                  //total discards (klb) by yr summed over ages
  matrix B_total(styr,endyr,1,nages);             //bycatch in klb
  vector B_total_yr(styr,endyr);                  //total bycatch (klb) by yr summed over ages

  matrix C_CN_M(styr,endyr,1,nages);              //landings (numbers) at age
  matrix C_CN_F(styr,endyr,1,nages);              //landings (numbers) at age
  matrix L_CN(styr,endyr,1,nages);                //landings (mt) at age
  matrix L_CN_klb(styr,endyr,1,nages);            //landings (1000 lb whole weight) at age
  vector pred_CN_L(styr_CN_L,endyr_CN_L);         //yearly landings (klb) summed over ages

  matrix C_MRFSS_M(styr,endyr,1,nages);           //landings (numbers) at age
  matrix C_MRFSS_F(styr,endyr,1,nages);           //landings (numbers) at age
  matrix L_MRFSS(styr,endyr,1,nages);             //landings (mt) at age
  matrix L_MRFSS_klb(styr,endyr,1,nages);         //landings (1000 lb whole weight) at age
  vector pred_MRFSS_L(styr_MRFSS_L,endyr_MRFSS_L); //yearly landings (numbers in 1000's) summed over ages

//---Discards (number dead fish) ------------------------------------------------


  matrix C_MRFSS_D_M(styr_MRFSS_D,endyr_MRFSS_D,1,nages);    //discards (numbers) at age
  matrix C_MRFSS_D_F(styr_MRFSS_D,endyr_MRFSS_D,1,nages);    //discards (numbers) at age
  matrix L_MRFSS_D(styr,endyr,1,nages);                      //discards in mt whole weight
  vector pred_MRFSS_D(styr_MRFSS_D,endyr_MRFSS_D);           //yearly discards summed over ages
  vector obs_MRFSS_D(styr_MRFSS_D,endyr_MRFSS_D);            //observed releases multiplied by discard mortality

//---Bycatch  -------------------------------------------------------------------
  matrix C_shrimp_M(styr,endyr,1,nages);
  matrix C_shrimp_F(styr,endyr,1,nages);
  matrix L_shrimp(styr,endyr,1,nages);       //bycatch in mt whole weight
  vector pred_shrimp_B(styr,endyr);
 // matrix C_shrimp_M(styr_shrimp_B,endyr_shrimp_B,1,nages);
 // matrix C_shrimp_F(styr_shrimp_B,endyr_shrimp_B,1,nages);
 // matrix L_shrimp(styr,endyr,1,nages);       //bycatch in mt whole weight
 // vector pred_shrimp_B(styr_shrimp_B,endyr_shrimp_B);

  //vector log_obs_shrimp_B(styr_shrimp_B_fit,endyr_shrimp_B_fit);
  //vector log_pred_shrimp_B(styr_shrimp_B_fit,endyr_shrimp_B_fit);
  //number obs_shrimp_geomean;
  //number pred_shrimp_geomean;
//---MSY calcs-------------------------------------------------------------------------

  number F_HL_prop;       //proportion of F_full attributable to hand lines, last three yrs
  number F_PN_prop;
  number F_GN_prop;
  number F_CN_prop;
  number F_MRFSS_prop;
  number F_MRFSS_D_prop;
  number F_shrimp_prop;
  number F_temp_sum;      //sum of geom mean full Fs in last yrs, used to compute F_fishery_prop

  number SSB_msy_out;           //SSB at msy
  number F_msy_out;             //F at msy
  number msy_out;               //max sustainable yield
  number B_msy_out;             //total biomass at MSY
  number E_msy_out;             //exploitation rate at MSY (ages E_age_st plus)
  number R_msy_out;             //equilibrium recruitment at F=Fmsy
  number D_msy_out;             //equilibrium dead discards at F=Fmsy
  number spr_msy_out;           //spr at F=Fmsy

  vector N_age_msy_F(1,nages);         //numbers at age for MSY calculations: beginning of yr - Females
  vector N_age_msy_mdyr_F(1,nages);    //numbers at age for MSY calculations: mdpt of yr
  vector C_age_msy_F(1,nages);         //catch at age for MSY calculations
  vector Z_age_msy_F(1,nages);         //total mortality at age for MSY calculations
  vector D_age_msy_F(1,nages);         //discard mortality (dead discards) at age for MSY calculations
  vector F_L_age_msy_F(1,nages);       //fishing mortality (landings, not discards) at age for MSY calculations
  vector F_D_age_msy_F(1,nages);
  vector N_age_msy_M(1,nages);         //numbers at age for MSY calculations: beginning of yr - Males
  vector N_age_msy_mdyr_M(1,nages);    //numbers at age for MSY calculations: mdpt of yr
  vector C_age_msy_M(1,nages);         //catch at age for MSY calculations
  vector Z_age_msy_M(1,nages);         //total mortality at age for MSY calculations
  vector D_age_msy_M(1,nages);         //discard mortality (dead discards) at age for MSY calculations
  vector F_L_age_msy_M(1,nages);       //fishing mortality (landings, not discards) at age for MSY calculations
  vector F_D_age_msy_M(1,nages);
  vector F_msy(1,n_iter_msy);         //values of full F to be used in per-recruit and equilibrium calculations
  vector spr_msy(1,n_iter_msy);       //reproductive capacity-per-recruit values corresponding to F values in F_msy
  vector R_eq(1,n_iter_msy);      //equilibrium recruitment values corresponding to F values in F_msy
  vector L_eq(1,n_iter_msy);      //equilibrium landings(mt) values corresponding to F values in F_msy
  vector SSB_eq(1,n_iter_msy);    //equilibrium reproductive capacity values corresponding to F values in F_msy
  vector B_eq(1,n_iter_msy);      //equilibrium biomass values corresponding to F values in F_msy
  vector E_eq(1,n_iter_msy);      //equilibrium exploitation rates corresponding to F values in F_msy
  vector D_eq(1,n_iter_msy);      //equilibrium discards (1000s) corresponding to F values in F_msy

  vector FdF_msy(styr,endyr);
  vector EdE_msy(styr,endyr);
```

```
  vector SdSSB_msy(styr,endyr);
  number SdSSB_msy_end;
  number FdF_msy_end;
  number EdE_msy_end;

//--------Mortality------------------------------------------------------------------------
  vector M(1,nages);                        //age-dependent natural mortality
  number M_constant;                        //age-indpendent: used only for MSST
  matrix F_M(styr,endyr,1,nages);           //males
  matrix F_F(styr,endyr,1,nages);           //females

  number F_hist;                       //historical F input into assessment model
  vector F_F_hist(1,nages);                          //historical F used to get stable age dist in 1st year
  vector Z_hist(1,nages);
  vector fullF(styr,endyr);                  //Fishing mortality rate by year
  vector E(styr,endyr);                      //Exploitation rate by year
  vector fullF_sd(styr,endyr);
  vector E_sd(styr,endyr);
  matrix Z_M(styr,endyr,1,nages);            //males
  matrix Z_F(styr,endyr,1,nages);            //females

  init_bounded_number log_avg_F_HL(-10,1,1);
  init_bounded_dev_vector log_F_dev_HL(styr_HL_L,endyr_HL_L,-20,5,2);
  matrix F_HL_M(styr,endyr,1,nages);
  matrix F_HL_F(styr,endyr,1,nages);
  vector F_HL_out(styr,endyr); //used for intermediate calculations in fcn get_mortality
  //number log_F_init_commDV;

  init_bounded_number log_avg_F_PN(-10,0,1);
  init_bounded_dev_vector log_F_dev_PN(styr_PN_L,endyr_PN_L,-20,5,2);
  matrix F_PN_M(styr,endyr,1,nages);
  matrix F_PN_F(styr,endyr,1,nages);
  vector F_PN_out(styr,endyr);        //used for intermediate calculations in fcn get_mortality

  init_bounded_number log_avg_F_GN(-10,0,1);
  init_bounded_dev_vector log_F_dev_GN(styr_GN_L,endyr_GN_L,-20,5,2);
  matrix F_GN_M(styr,endyr,1,nages);
  matrix F_GN_F(styr,endyr,1,nages);
  vector F_GN_out(styr,endyr); //used for intermediate calculations in fcn get_mortality

  init_bounded_number log_avg_F_CN(-10,0,1);
  init_bounded_dev_vector log_F_dev_CN(styr_CN_L,endyr_CN_L,-20,5,2);
  matrix F_CN_M(styr,endyr,1,nages);
  matrix F_CN_F(styr,endyr,1,nages);
  vector F_CN_out(styr,endyr);        //used for intermediate calculations in fcn get_mortality

  init_bounded_number log_avg_F_MRFSS(-10,0,1);
  init_bounded_dev_vector log_F_dev_MRFSS(styr_MRFSS_L,endyr_MRFSS_L,-20,5,2);
  matrix F_MRFSS_M(styr,endyr,1,nages);
  matrix F_MRFSS_F(styr,endyr,1,nages);
  vector F_MRFSS_out(styr,endyr); //used for intermediate calculations in fcn get_mortality
  number MRFSS_ratio;  //ratio of F_MRFSS/F_GN for saltwater report years to apply to missing years;
                       //straight interpolation used from last angler report to first 'real' year of landings

//--Discard mortality stuff-------------------------------------------------------------------------

  init_bounded_number log_avg_F_MRFSS_D(-15,0,1);
  init_bounded_dev_vector log_F_dev_MRFSS_D(styr_MRFSS_D,endyr_MRFSS_D,-20,5,2);
  matrix F_MRFSS_D_M(styr,endyr,1,nages);
  matrix F_MRFSS_D_F(styr,endyr,1,nages);
  vector F_MRFSS_D_out(styr,endyr); //used for intermediate calculations in fMRFSS get_mortality
  number F_MRFSS_D_ratio;  //ratio of average discard F to fishery F, for projection discards back before data

  number Dmort_MRFSS;

//--------Bycatch ------------------------------;
  init_bounded_number log_avg_F_shrimp(-15,0,1);
  init_bounded_dev_vector log_F_dev_shrimp(styr_shrimp_B,endyr_shrimp_B,-20,6,2);
  matrix F_shrimp(styr,endyr,1,nages);
  vector F_shrimp_out(styr,endyr);        //used for intermediate calculations in fcn get_mortality

////---Per-recruit stuff-------------------------------------------------------------------------
  vector N_age_spr_F(1,nages);         //numbers at age for SPR calculations: beginning of year, females only
  vector N_age_spr_mdyr_F(1,nages);    //numbers at age for SPR calculations: midyear, females only
  vector C_age_spr_F(1,nages);         //catch at age for SPR calculations, females only
  vector Z_age_spr_F(1,nages);         //total mortality at age for SPR calculations, females only
  vector F_L_age_spr_F(1,nages);      //fishing mortality (landings, not discards) at age for SPR calculations, females
  vector N_age_spr_M(1,nages);         //numbers at age for SPR calculations: beginning of year, males only
  vector N_age_spr_mdyr_M(1,nages);    //numbers at age for SPR calculations: midyear, males only
  vector C_age_spr_M(1,nages);         //catch at age for SPR calculations, males only
  vector Z_age_spr_M(1,nages);         //total mortality at age for SPR calculations, males only
  vector F_L_age_spr_M(1,nages);      //fishing mortality (landings, not discards) at age for SPR calculations, males
  vector spr_static(styr,endyr);     //vector of static SPR values by year
  vector F_spr(1,n_iter_spr);         //values of full F to be used in per-recruit and equilibrium calculations
  vector spr_spr(1,n_iter_spr);       //reporductive capacity-per-recruit values corresponding to F values in F_spr
  vector L_spr(1,n_iter_spr);         //landings(mt)-per-recruit values corresponding to F values in F_spr
  vector E_spr(1,n_iter_spr);         //exploitation rate values corresponding to F values in F_spr

  vector N_spr_F0(1,nages);           //Used to compute spr at F=0: midpt of year
  vector N_bpr_F0(1,nages);           //Used to compute bpr at F=0: start of year
  number spr_F0;                      //Spawning biomass per recruit at F=0
  vector N_spr_F_init(1,nages);          //Used to compute spr at F=0: start of year
  vector N_spr_F_init_mdyr(1,nages);           //Used to compute spr at F=0: midpt of year
  number spr_F_init;                 //Spawning biomass per recruit at F=F_F_hist

  number bpr_F0;                      //Biomass per recruit at F=0

//-------Objective function components---------------------------------------------------------------
  number w_L;
  number w_D;
  number w_ac_HL;
  number w_ac_PN;
  number w_ac_GN;
  number w_ac_CN;
```

```
  number w_ac_MRFSS;
  number w_I_FL_HL;
  number w_I_MRFSS;
 // number w_I_SMAP_1YR;
  number w_I_SMAP_YOY;
  number w_R;
  number w_R_init;
  number w_R_end;
  number w_F;
  number w_B1dB0;
  number w_fullF;
  number w_cvlen_dev;
  number w_cvlen_diff;

  number f_FL_HL_cpue;
  number f_MRFSS_cpue;
  number f_SMAP_YOY_cpue;
  //number f_SMAP_1YR_cpue;

  number f_HL_L;
  number f_PN_L;
  number f_GN_L;
  number f_CN_L;
  number f_MRFSS_L;

  number f_MRFSS_D;

  number f_shrimp_B;

  number f_HL_agec;
  number f_PN_agec;
  number f_GN_agec;
  number f_CN_agec;
  number f_MRFSS_agec;

  number f_N_dev;               //weight on recruitment deviations to fit S-R curve
  number f_N_dev_early;         //extra weight against deviations before styr
  number f_N_dev_end;          //extra constraint on last 3 years of recruitment variability
  number f_Fend_constraint;     //penalty for F deviation in last 5 years
  number f_B1dB0_constraint;    //penalty to fix B(styr)/K
  number f_fullF_constraint;    //penalty for fullF>5
  number f_cvlen_dev_constraint; //deviation penalty on cv's of length at age
  number f_cvlen_diff_constraint;//first diff penalty on cv's of length at age
number f_priors; //prior information on parameters

  objective_function_value fval;
  number fval_unwgt;

//--Dummy arrays for output convenience  --------------------------
  vector xdum(styr,endyr);
  vector xdum2(styr,endyr+1);
//--Other dummy variables ----
  number sel_diff_dum;
  number zero_dum;
  number dzero_dum;
  number huge_number;
 // init_number x_dum; //used only during model development. can be removed.

//##--
INITIALIZATION_SECTION


//##--
//##--
GLOBALS_SECTION
  #include "admodel.h"          // Include AD class definitions   KWS
  #include "admb2r.cpp"    // Include S-compatible output functions (needs preceding)  KWS

//##--
RUNTIME_SECTION
 maximum_function_evaluations 1000, 3000, 10000, 20000;
 //maximum_function_evaluations 1, 1, 1;
 convergence_criteria 1e-4, 1e-4, 1e-4;

//##--
//##--
PRELIMINARY_CALCS_SECTION
// Set values of fixed parameters or set initial guess of estimated parameters
  Dmort_MRFSS=set_Dmort_MRFSS;

  obs_MRFSS_D=Dmort_MRFSS*obs_MRFSS_released;
  obs_MRFSS_L(styr_MRFSS_L,1979)=L_rec_multiply*obs_MRFSS_L(styr_MRFSS_L,1979);
  obs_MRFSS_D(styr_MRFSS_L,1979)=L_rec_multiply*obs_MRFSS_D(styr_MRFSS_L,1979);

  E_age_st=set_E_age_st;   //E computed over ages E_age_st +   [E_age_st-1+ if model starts with age 0]

  //Linf_f=set_Linf_f;
  //Linf_m=set_Linf_m;
  //K_f=set_K_f;
  //K_m=set_K_m;
  //t0_f=set_t0_f;
  //t0_m=set_t0_m;

  nlenbins2=nlenbins+20;  //lenbins,nlenbins for plus group; lenbins2,nlenbins2 assume truncation at 90cm
  lenbins2(1)=lenbins(1);
  for(ilen=2;ilen<=nlenbins2;ilen++){
    lenbins2(ilen)=lenbins2(ilen-1)+1;
  }
  prop_f_a0=set_prop_f_a0;
  //selpar_L50_D_2=t0-log(1.0-304.8/Linf)/K; //age at size limit: 304.8 = limit in mm

  M=set_M;
  M_constant=set_M_constant;
  steep=set_steep;
```

```
  //log_dev_N_rec=0.0;
  R_autocorr=set_R_autocorr;
  rec_sigma=set_rec_sigma;

  log_q_FL_HL=set_logq_FL_HL;
  log_q_MRFSS=set_logq_MRFSS;
  log_q_SMAP_YOY=set_logq_SMAP_YOY;
  //log_q_SMAP_1YR=set_logq_SMAP_1YR;
//log_q_shrimp_B=set_logq_shrimp_B;
  //q_rate=set_q_rate;

  //L_early_bias=set_L_early_bias;
  //L_commHAL_bias=1.0;

  w_L=set_w_L;
  w_D=set_w_D;
  //w_shrimp=set_w_shrimp;
  //w_lc_=set_w_lc;
 //w_lc_HL = set_w_lc_HL;
 //w_lc_PN = set_w_lc_PN;
 //w_lc_GN = set_w_lc_GN;
 //w_lc_CN = set_w_lc_CN;
 //w_lc_MRFSS = set_w_lc_MRFSS;
  //w_ac=set_w_ac;
w_ac_HL = set_w_ac_HL;
  w_ac_PN = set_w_ac_PN;
  w_ac_GN = set_w_ac_GN;
  w_ac_CN = set_w_ac_CN;
  w_ac_MRFSS = set_w_ac_MRFSS;
  w_I_FL_HL=set_w_I_FL_HL;
  w_I_MRFSS=set_w_I_MRFSS;
  w_I_SMAP_YOY=set_w_I_SMAP_YOY;
  //w_I_SMAP_1YR=set_w_I_SMAP_1YR;

  w_R=set_w_R;
  w_R_init=set_w_R_init;
  w_R_end=set_w_R_end;
  w_F=set_w_F;
  w_B1dB0=set_w_B1dB0;
  w_full1F=set_w_full1F;
  w_cvlen_dev=set_w_cvlen_dev;
  w_cvlen_diff=set_w_cvlen_diff;

  F_hist=set_F_hist;
  log_avg_F_HL=set_log_avg_F_HL;
  log_F_dev_HL=set_log_F_dev_HL;
  log_avg_F_PN=set_log_avg_F_PN;
  log_F_dev_PN=set_log_F_dev_PN;
  log_avg_F_GN=set_log_avg_F_GN;
  log_F_dev_GN=set_log_F_dev_GN;
  log_avg_F_CN=set_log_avg_F_CN;
  log_F_dev_CN=set_log_F_dev_CN;
  log_avg_F_MRFSS=set_log_avg_F_MRFSS;
  log_F_dev_MRFSS=set_log_F_dev_MRFSS;

  log_avg_F_MRFSS_D=set_log_avg_F_MRFSS_D;
  log_F_dev_MRFSS_D=set_log_F_dev_MRFSS_D;

  log_avg_F_shrimp=set_log_avg_F_shrimp;
  log_F_dev_shrimp=set_log_F_dev_shrimp;

  log_len_cv=log(set_len_cv);
  log_R0=set_log_R0;

  L50_diff=set_L50_diff; //shift in selectivity b/w males and females (same for all gears)
  sel_historical_F=set_sel_historical_F;

  selpar_slope_HL=set_selpar_slope_HL;
  selpar_L50_HL_keep=set_selpar_L50_HL_keep;

  selpar_slope1_PN=set_selpar_slope1_PN;
  selpar_L501_PN=set_selpar_L501_PN;
  selpar_slope2_PN=set_selpar_slope2_PN;
  selpar_L502_PN=set_selpar_L501_PN;

  selpar_slope_GN=set_selpar_slope_GN;
  selpar_L50_GN_keep=set_selpar_L50_GN_keep;

  selpar_slope1_CN=set_selpar_slope1_CN;
  selpar_L501_CN=set_selpar_L501_CN;
  selpar_slope2_CN=set_selpar_slope2_CN;
  selpar_L502_CN=set_selpar_L502_CN;

  selpar_slope1_MRFSS_keep=set_selpar_slope1_MRFSS_keep;
  selpar_L501_MRFSS_keep=set_selpar_L501_MRFSS_keep;
  selpar_slope2_MRFSS_keep=set_selpar_slope2_MRFSS_keep;
  selpar_L502_MRFSS_keep=set_selpar_L502_MRFSS_keep;
  //selpar_sigma_MRFSS=set_selpar_sigma_MRFSS;
  //selpar_afull_MRFSS=set_selpar_afull_MRFSS

  sel_MRFSS_D_M=set_sel_MRFSS_D_M;  //selectivity for discards - males
  sel_MRFSS_D_F=set_sel_MRFSS_D_F;  //selectivity for discards - females
  sel_shrimp=set_sel_shrimp_B;  //selectivity for bycatch (same for both sexes)

 sqrt2pi=sqrt(2.*3.14159265);
 g2mt=0.000001;         //conversion of grams to metric tons
 g2kg=0.001;         //conversion of grams to kg
 mt2klb=2.20462;   //converstion of metric tons to 1000 lb
 //df=0.001; //difference for msy derivative approximations
 zero_dum=0.0;

 //additive constant to prevent division by zero
 dzero_dum=0.00001;
```

```
 SSB_msy_out=0.0;

 maturity_f=maturity_f_obs;

//Fill in sample sizes of comps sampled in nonconsec yrs.
//Used only for output in R object
      //nsamp_HL_lenc_allyr=missing;
      //nsamp_CN_lenc_allyr=missing;

      nsamp_HL_agec_allyr=missing;
      nsamp_PN_agec_allyr=missing;
      nsamp_CN_agec_allyr=missing;
      nsamp_GN_agec_allyr=missing;
      nsamp_MRFSS_agec_allyr=missing;

      for (iyear=1; iyear<=nyr_HL_agec; iyear++)
         {
           nsamp_HL_agec_allyr(yrs_HL_agec(iyear))=nsamp_HL_agec(iyear);
         }
      for (iyear=1; iyear<=nyr_PN_agec; iyear++)
         {
           nsamp_PN_agec_allyr(yrs_PN_agec(iyear))=nsamp_PN_agec(iyear);
         }
      for (iyear=1; iyear<=nyr_CN_agec; iyear++)
         {
           nsamp_CN_agec_allyr(yrs_CN_agec(iyear))=nsamp_CN_agec(iyear);
         }
         //cout<<"nsamp vector CN"<<nsamp_CN_agec_allyr<<endl;
for (iyear=1; iyear<=nyr_GN_agec; iyear++)
         {
           nsamp_GN_agec_allyr(yrs_GN_agec(iyear))=nsamp_GN_agec(iyear);
         }
        // cout<<"nsamp vector GN"<<nsamp_GN_agec_allyr<<endl;
      for (iyear=1; iyear<=nyr_MRFSS_agec; iyear++)
         {
           nsamp_MRFSS_agec_allyr(yrs_MRFSS_agec(iyear))=nsamp_MRFSS_agec(iyear);
         }
     //  cout<<"nsamp vector MRFSS"<<nsamp_MRFSS_agec_allyr<<endl;
  age_error_matrix=set_age_error_matrix;

//fill in reproductive caculations with zero's
  reprod.initialize();
  prop_f.initialize();
  prop_f_F0.initialize();

//fill in F's, Catch matrices, and log rec dev with zero's
  F_HL_M.initialize();
  F_HL_F.initialize();
  C_HL_M.initialize();
  C_HL_F.initialize();
  F_PN_M.initialize();
  F_PN_F.initialize();
  C_PN_F.initialize();
  C_PN_M.initialize();
  F_GN_M.initialize();
  F_GN_F.initialize();
  C_GN_M.initialize();
  C_GN_F.initialize();
  F_CN_M.initialize();
  F_CN_F.initialize();
  C_CN_M.initialize();
  C_CN_F.initialize();
  F_MRFSS_M.initialize();
  F_MRFSS_F.initialize();
  C_MRFSS_M.initialize();
  C_MRFSS_F.initialize();
  F_shrimp.initialize();
  C_shrimp_M.initialize();
  C_shrimp_F.initialize();

  C_MRFSS_D_M.initialize();
  C_MRFSS_D_F.initialize();

  L_MRFSS_D.initialize();        //discards in 1000lb whole weight
  L_shrimp.initialize();        //discards in 1000lb whole weight

  F_MRFSS_D_M.initialize();
  F_MRFSS_D_F.initialize();

  log_dev_R.initialize();

//##--
TOP_OF_MAIN_SECTION
  arrmblsize=20000000;
  gradient_structure::set_MAX_NVAR_OFFSET(1600);
  gradient_structure::set_GRADSTACK_BUFFER_SIZE(2000000);
  gradient_structure::set_CMPDIF_BUFFER_SIZE(2000000);
  gradient_structure::set_NUM_DEPENDENT_VARIABLES(500);


//
//##--
PROCEDURE_SECTION
R0=mfexp(log_R0);
  // cout<<"start"<<endl;
get_length_and_weight_at_age();
  // cout << "got length and weight transitions" <<endl;
get_selectivity();
 //  cout << "got selectivity" << endl;
get_length_at_age_dist();
  // cout<< "got predicted length at age distribution"<<endl;
get_spr_F0();
  // cout << "got F0 spr" << endl;
get_mortality();
```

```
  // cout << "got mortalities" << endl;
get_bias_corr();
  // cout<< "got recruitment bias correction" << endl;
get_numbers_at_age();
  // cout << "got numbers at age" << endl;
get_landings_numbers();
  // cout << "got catch at age" << endl;
get_landings_wgt();
  // cout << "got landings" << endl;
get_dead_discards();
  // cout << "got discards" << endl;
//get_length_comps();
  // cout<< "got length comps"<< endl;
get_age_comps();
  // cout<< "got age comps"<< endl;
get_sel_weighted_current();
  // cout<<"got sel weighted"<<endl;
get_indices();
  // cout << "got indices" << endl;
  evaluate_objective_function();
  // cout << "objective function calculations complete" << endl;


FUNCTION get_length_and_weight_at_age
  //compute mean length (mm) and weight (whole and gutted) at age
    //meanlen_m=Linf_m*(1.0-mfexp(-K_m*(agebins-t0_m)));   //length in mm - males
    //cout<<"mean length at age--males"<<meanlen_m<<endl;

    //meanlen_f=Linf_f*(1.0-mfexp(-K_f*(agebins-t0_f)));   //length in mm - females
    //cout<<"mean length at age--females"<<meanlen_f<<endl;

    wgt_kg_land=wgtpar_a*pow(meanlen_fishery,wgtpar_b); //wgt in kg in the fishery
    //wgt_kg_land=g2kg*wgt_g_land; //wgt in kilos in the fishery
    wgt_land=0.001*wgt_kg_land; //mt of whole wgt:
    wgt_klb_land=mt2klb*wgt_land; //1000lb of whole wgt

    wgt_kg_m=wgtpar_am*pow(meanlen_m,wgtpar_bm);        //wgt in kg
    //wgt_kg_m=g2kg*wgt_g_m;                           //wgt in kilos
    wgt_m=0.001*wgt_kg_m;      //mt of whole wgt: g2mt converts g to mt
    wgt_klb_m=mt2klb*wgt_m;                          //1000 lb of whole wgt

    wgt_kg_f=wgtpar_af*pow(meanlen_f,wgtpar_bf);        //wgt in grams
    //wgt_kg_f=g2kg*wgt_g_f;                           //wgt in kilos
    wgt_f=0.001*wgt_kg_f;      //mt of whole wgt: g2mt converts g to mt
    wgt_klb_f=mt2klb*wgt_f;    //1000s lb

FUNCTION get_selectivity
// ------- compute landings and discard selectivities
// for (iyear=styr; iyear<=endyr; iyear++)
//{
// sel_HL_keep_M(iyear)=logistic(agebins, selpar_L50_HL, selpar_slope_HL);
//sel_HL_keep_M(iyear)=logistic(agebins, selpar_L50_HL, selpar_slope_HL);
//sel_HL_keep_M(iyear)=logistic(agebins, selpar_L50_HL, selpar_slope_HL);


  for (iage=1; iage<=nages; iage++)
  {
    sel_HL_keep_M(iage)=1./(1.+mfexp(-1.*selpar_slope_HL*(double(agebins(iage))-(selpar_L50_HL_keep-L50_diff))));
     sel_HL_keep_F(iage)=1./(1.+mfexp(-1.*selpar_slope_HL*(double(agebins(iage))-selpar_L50_HL_keep)));
     //sel_PN_M(iage)=1./(1.+mfexp(-1.*selpar_slope_PN*(double(agebins(iage))-selpar_L50_PN-L50_diff)));
     //sel_PN_F(iage)=1./(1.+mfexp(-1.*selpar_slope_PN*(double(agebins(iage))-selpar_L50_PN)));
     sel_PN_M(iage)=(1./(1.+mfexp(-1.*selpar_slope1_PN*(double(agebins(iage))-
                     (selpar_L501_PN-L50_diff)))))*(1-(1./(1.+mfexp(-1.*selpar_slope2_PN*
                     (double(agebins(iage))-((selpar_L501_PN-L50_diff)+selpar_L502_PN))))); //double logistic
     sel_PN_F(iage)=(1./(1.+mfexp(-1.*selpar_slope1_PN*(double(agebins(iage))-
                     selpar_L501_PN))))*(1-(1./(1.+mfexp(-1.*selpar_slope2_PN*
                     (double(agebins(iage))-(selpar_L501_PN+selpar_L502_PN))))); //double logistic
     sel_GN_keep_M(iage)=1./(1.+mfexp(-1.*selpar_slope_GN*(double(agebins(iage))-(selpar_L50_GN_keep-L50_diff))));
     sel_GN_keep_F(iage)=1./(1.+mfexp(-1.*selpar_slope_GN*(double(agebins(iage))-selpar_L50_GN_keep)));
     //sel_GN_keep_M2(iage)=(1./(1.+mfexp(-1.*selpar_slope_GN2*(double(agebins(iage))-
     //                  selpar_L50_GN2))))*(1-(1./(1.+mfexp(-1.*selpar_slope2_GN2*
     //                  (double(agebins(iage))-(selpar_L50_GN2+selpar_L502_GN2)))))); //double logistic


     //sel_CN_M(iage)=1./(1.+mfexp(-1.*selpar_slope_CN*(double(agebins(iage))-selpar_L50_CN-L50_diff)));
     //sel_CN_F(iage)=1./(1.+mfexp(-1.*selpar_slope_CN*(double(agebins(iage))-selpar_L50_CN)));
   }
   //sel_CN_M=logistic_exponential(agebins, selpar_L50_CN, selpar_slope_CN, selpar_sigma_CN, selpar_afull_CN);
   //sel_CN_F=logistic_exponential(agebins, selpar_L50_CN, selpar_slope_CN, selpar_sigma_CN, selpar_afull_CN);
    //--Implement Butterworth's Logistic exponential selex to avoid having to take a max(non-differentiable)


    sel_CN_M=logistic_double(agebins, (selpar_L501_CN-L50_diff), selpar_slope1_CN, selpar_L502_CN, selpar_slope2_CN);
    sel_CN_F=logistic_double(agebins, selpar_L501_CN, selpar_slope1_CN, selpar_L502_CN, selpar_slope2_CN);
    sel_MRFSS_keep_M=logistic_double(agebins, (selpar_L501_MRFSS_keep-L50_diff), selpar_slope1_MRFSS_keep, selpar_L502_MRFSS_keep, selpar_slope2_MRFSS_keep);
    sel_MRFSS_keep_F=logistic_double(agebins, selpar_L501_MRFSS_keep, selpar_slope1_MRFSS_keep, selpar_L502_MRFSS_keep, selpar_slope2_MRFSS_keep);
    //sel_MRFSS_keep_M(1)=0.1;

     //sel_MRFSS_keep_M(iage)=1./(1.+mfexp(-1.*selpar_slope_MRFSS*(double(agebins(iage))-selpar_L50_MRFSS_keep-L50_diff)));
     //sel_MRFSS_keep_F(iage)=1./(1.+mfexp(-1.*selpar_slope_MRFSS*(double(agebins(iage))-selpar_L50_MRFSS_keep)));
  //sel_MRFSS_keep_M=logistic_exponential(agebins, selpar_L501_MRFSS_keep, selpar_slope1_MRFSS_keep, selpar_sigma_MRFSS, selpar_afull_MRFSS);
  //sel_MRFSS_keep_F=logistic_exponential(agebins, selpar_L501_MRFSS_keep, selpar_slope1_MRFSS_keep, selpar_sigma_MRFSS, selpar_afull_MRFSS);
  //sel_MRFSS_keep_M=sel_MRFSS_keep_M/max(sel_MRFSS_keep_M);
 //sel_GN_keep_M2=sel_GN_keep_M2/max(sel_GN_keep_M2); //renormalize
  sel_PN_M=sel_PN_M/max(sel_PN_M);
  sel_PN_F=sel_PN_F/max(sel_PN_F);
  //sel_GN_keep_F2=sel_GN_keep_M2;
 // cout <<"HL male selex"<<sel_HL_keep_M<< endl;
 // cout <<"PN male selex"<<sel_PN_M<< endl;
 // cout <<"GN male selex"<<sel_GN_keep_M<< endl;
 // cout <<"CN male selex"<<sel_CN_M<< endl;
 // cout <<"MRFSS male selex"<<sel_MRFSS_keep_M<< endl;

FUNCTION get_length_at_age_dist
```

```
  //compute matrix of length at age, based on the normal distribution
  for (iage=1;iage<=nages;iage++)
  {
    //len_cv(iage)=mfexp(log_len_cv+log_len_cv_dev(iage));
    len_cv(iage)=mfexp(log_len_cv);
    for (ilen=1;ilen<=nlenbins2;ilen++)
    {
      //convert to centimeters for matching
      lenprob_m2(iage,ilen)=(mfexp(-(square(lenbins2(ilen)-0.1*meanlen_m(iage))/
      (2.*square(len_cv(iage)*0.1*meanlen_m(iage)))))/(sqrt2pi*0.1*len_cv(iage)*meanlen_m(iage)));

      lenprob_f2(iage,ilen)=(mfexp(-(square(lenbins2(ilen)-0.1*meanlen_f(iage))/
      (2.*square(len_cv(iage)*0.1*meanlen_f(iage)))))/(sqrt2pi*0.1*len_cv(iage)*meanlen_f(iage)));
    }
    lenprob_m2(iage)/=sum(lenprob_m2(iage)); //standardize to account for truncated normal (i.e., no sizes<0)
    lenprob_f2(iage)/=sum(lenprob_f2(iage)); //standardize to account for truncated normal (i.e., no sizes<0)
    for(ilen=1;ilen<=nlenbins;ilen++){
      lenprob_m(iage,ilen)=lenprob_m2(iage,ilen);
      lenprob_f(iage,ilen)=lenprob_f2(iage,ilen);
    }
    for(ilen=nlenbins+1;ilen<=nlenbins2;ilen++){
      lenprob_m(iage,nlenbins)+=lenprob_m2(iage,ilen);
      lenprob_f(iage,nlenbins)+=lenprob_f2(iage,ilen);
    }
  }

FUNCTION get_spr_F0
  reprod=elem_prod(maturity_f,wgt_f);
  //at mdyr, apply half this yr's mortality, half next yr's
  N_spr_F0(1)=prop_f_a0*mfexp(-1.0*M(1)/2.0);  //at midpt of year
  N_bpr_F0(1)=1.;
  for (iage=2; iage<=nages; iage++)
  {
    //N_spr_F0(iage)=N_spr_F0(iage-1)*mfexp(-1.0*(M(iage-1));
    N_spr_F0(iage)=N_spr_F0(iage-1)*mfexp(-1.0*(M(iage-1)/2.0 + M(iage)/2.0));
    N_bpr_F0(iage)=N_bpr_F0(iage-1)*mfexp(-1.0*(M(iage-1)));
  }
  N_spr_F0(nages)=N_spr_F0(nages)/(1.0-mfexp(-1.0*M(nages))); //plus group (sum of geometric series)
  N_bpr_F0(nages)=N_bpr_F0(nages)/(1.0-mfexp(-1.0*M(nages)));

  spr_F0=sum(elem_prod(N_spr_F0,reprod));
  bpr_F0=prop_f_a0*sum(elem_prod(N_bpr_F0,wgt_f))+(1.-prop_f_a0)*sum(elem_prod(N_bpr_F0,wgt_m));
  B0=R0*bpr_F0;

FUNCTION get_mortality
  fullF=0.0;

  for (iyear=styr; iyear<=endyr; iyear++) {
    if(iyear<styr_HL_L){
      F_HL_out(iyear)=0.0;
    }
    else{
      F_HL_out(iyear)=mfexp(log_avg_F_HL+log_F_dev_HL(iyear));
    }
    F_HL_F(iyear)=sel_HL_keep_F*F_HL_out(iyear);
    F_HL_M(iyear)=sel_HL_keep_M*F_HL_out(iyear);
    fullF(iyear)+=F_HL_out(iyear);

    if(iyear<styr_PN_L){
      F_PN_out(iyear)=0.0;
    }
    else{
      F_PN_out(iyear)=mfexp(log_avg_F_PN+log_F_dev_PN(iyear));
    }
    F_PN_F(iyear)=sel_PN_F*F_PN_out(iyear);
    F_PN_M(iyear)=sel_PN_M*F_PN_out(iyear);
    fullF(iyear)+=F_PN_out(iyear);

    if(iyear<styr_GN_L){
      F_GN_out(iyear)=0.0;
    }
    else {
      F_GN_out(iyear)=mfexp(log_avg_F_GN+log_F_dev_GN(iyear));
    }

      F_GN_F(iyear)=sel_GN_keep_F*F_GN_out(iyear);
      F_GN_M(iyear)=sel_GN_keep_M*F_GN_out(iyear);
      fullF(iyear)+=F_GN_out(iyear);

    if(iyear<styr_CN_L){
      F_CN_out(iyear)=0.0;
    }
    else{
      F_CN_out(iyear)=mfexp(log_avg_F_CN+log_F_dev_CN(iyear));
    }
    F_CN_F(iyear)=sel_CN_F*F_CN_out(iyear);
    F_CN_M(iyear)=sel_CN_M*F_CN_out(iyear);
    fullF(iyear)+=F_CN_out(iyear);

    if(iyear<styr_MRFSS_L){
      F_MRFSS_out(iyear)=0.0;
    }
    else{
      F_MRFSS_out(iyear)=mfexp(log_avg_F_MRFSS+log_F_dev_MRFSS(iyear));
    }
    F_MRFSS_F(iyear)=sel_MRFSS_keep_F*F_MRFSS_out(iyear);
    F_MRFSS_M(iyear)=sel_MRFSS_keep_M*F_MRFSS_out(iyear);
    fullF(iyear)+=F_MRFSS_out(iyear);
    //cout<<"Rec F"<<F_MRFSS_M<<endl;

    //discards

    if(iyear<styr_MRFSS_D){
      F_MRFSS_D_out(iyear)=0.0;
```

```
    }
    else{
      F_MRFSS_D_out(iyear)=mfexp(log_avg_F_MRFSS_D+log_F_dev_MRFSS_D(iyear));
    }
    fullF(iyear)+=F_MRFSS_D_out(iyear);
    F_MRFSS_D_M(iyear)=sel_MRFSS_D_M*F_MRFSS_D_out(iyear);
    F_MRFSS_D_F(iyear)=sel_MRFSS_D_F*F_MRFSS_D_out(iyear);

    if(iyear<styr_shrimp_B){
      F_shrimp_out(iyear)=0.0;
    }
    else{
      F_shrimp_out(iyear)=mfexp(log_avg_F_shrimp+log_F_dev_shrimp(iyear));
    //F_shrimp_out(iyear)=mfexp(log_q_shrimp_B+log_obs_shrimp_B_effort(iyear));
    }
    fullF(iyear)+=F_shrimp_out(iyear);
    F_shrimp(iyear)=sel_shrimp*F_shrimp_out(iyear);
  }


  for (iyear=styr; iyear<=endyr; iyear++) {
    F_F(iyear)=F_HL_F(iyear);
    F_F(iyear)+=F_PN_F(iyear);
    F_F(iyear)+=F_GN_F(iyear);
    F_F(iyear)+=F_CN_F(iyear);
    F_F(iyear)+=F_MRFSS_F(iyear);
    F_F(iyear)+=F_MRFSS_D_F(iyear);
    F_F(iyear)+=F_shrimp(iyear);
    Z_F(iyear)=M+F_F(iyear);

    F_M(iyear)=F_HL_M(iyear);
    F_M(iyear)+=F_PN_M(iyear);
    F_M(iyear)+=F_GN_M(iyear);
    F_M(iyear)+=F_CN_M(iyear);
    F_M(iyear)+=F_MRFSS_M(iyear);
    F_M(iyear)+=F_MRFSS_D_M(iyear);
    F_M(iyear)+=F_shrimp(iyear);
    Z_M(iyear)=M+F_M(iyear);
  }
  F_F_hist=sel_historical_F*F_hist;
  Z_hist=F_F_hist+M;

FUNCTION get_bias_corr
  //var_rec_dev=norm2(log_dev_N_rec(styr_rec_dev,(endyr-2))-sum(log_dev_N_rec(styr_rec_dev,(endyr-2)))
  //           /(nyrs_rec-2.0))/(nyrs_rec-3.0); //sample variance from yrs styr_rec_dev-2005
  //if (set_BiasCor <= 0.0) {BiasCor=mfexp(var_rec_dev/2.0);}   //bias correction
  //else {BiasCor=set_BiasCor;}

    //may exclude last BiasCor_exclude_yrs yrs bc constrained or lack info to estimate
  //var_rec_dev=norm2(log_rec_dev(styr_rec_dev,(endyr-BiasCor_exclude_yrs))-
  //           sum(log_rec_dev(styr_rec_dev,(endyr-BiasCor_exclude_yrs)))
  //           /(nyrs_rec-BiasCor_exclude_yrs))/(nyrs_rec-BiasCor_exclude_yrs-1.0);
  var_rec_dev=norm2(log_rec_dev(styr_rec_dev,(endyr-2))-sum(log_rec_dev(styr_rec_dev,(endyr-2)))
             /(nyrs_rec-2.0))/(nyrs_rec-3.0);  //sample variance from yrs styr_rec_dev - 2009

  //if (set_BiasCor <= 0.0) {BiasCor=mfexp(var_rec_dev/2.0);}   //bias correction
  rec_sigma_sq=square(rec_sigma);
  if (set_BiasCor <= 0.0) {BiasCor=mfexp(rec_sigma_sq/2.0);}   //bias correction
  else {BiasCor=set_BiasCor;}

FUNCTION get_numbers_at_age
//Initial age
  S0=spr_F0*R0;

  //Assume equilibrium age structure for first year dependent on historical F
  N_spr_F_init(1)=prop_f_a0;
  N_spr_F_init_mdyr(1)=N_spr_F_init(1)*mfexp((-1.*(Z_hist(1)))/2.0);
  for (iage=2; iage<=nages; iage++)
  {
    N_spr_F_init(iage)=N_spr_F_init(iage-1)*mfexp(-1.*(Z_hist(iage-1)));
    N_spr_F_init_mdyr(iage)=N_spr_F_init(iage)*mfexp((-1.*(Z_hist(iage)))/2.0);
  }
  N_spr_F_init(nages)=N_spr_F_init(nages)/(1.0-mfexp(-1.*(Z_hist(nages))));
  N_spr_F_init_mdyr(nages)=N_spr_F_init(nages)*mfexp((-1.*(Z_hist(nages)))/2.0);
  spr_F_init=sum(elem_prod(N_spr_F_init_mdyr,reprod));

  R1=(R0/((5.0*steep-1.0)*spr_F_init))*(BiasCor*4.0*steep*spr_F_init-spr_F0*(1.0-steep));
  //R1=(R0/((5.0*steep-1.0)*spr_F_init))*(4.0*steep*spr_F_init-spr_F0*(1.0-steep));  //take out bias correction in first year because recruitment deterministic at the beginning
  if(R1<0.0)
    {
     R1=1.0;
     }
  // cout<<"R1"<<R1<<endl;
  N_M(styr,1)=(1.-prop_f_a0)*R1;
  N_F(styr,1)=prop_f_a0*R1;
  N_M_mdyr(styr,1)=N_M(styr,1)*mfexp(-1.*Z_M(styr,1)/2.0);
  N_F_mdyr(styr,1)=N_F(styr,1)*mfexp(-1.*Z_F(styr,1)/2.0);
  for (iage=2; iage<=nages; iage++)
  {
    N_M(styr,iage)=N_M(styr,iage-1)*mfexp(-1.*Z_hist(iage-1));
    N_M_mdyr(styr,iage)=N_M(styr,iage)*mfexp(-1.*Z_M(styr,iage)/2.0);
    N_F(styr,iage)=N_F(styr,iage-1)*mfexp(-1.*Z_hist(iage-1));
    N_F_mdyr(styr,iage)=N_F(styr,iage)*mfexp(-1.*Z_F(styr,iage)/2.0);
  }
  //plus group calculation
  N_M(styr,nages)=N_M(styr,nages)/(1.-mfexp(-1.*Z_hist(nages)));
  N_M_mdyr(styr,nages)=N_M(styr,nages)*mfexp(-1.*Z_M(styr,nages)/2.0);
  N_F(styr,nages)=N_F(styr,nages)/(1.-mfexp(-1.*Z_hist(nages)));
  N_F_mdyr(styr,nages)=N_F(styr,nages)*mfexp(-1.*Z_F(styr,nages)/2.0);

  SSB(styr)=sum(elem_prod(N_F_mdyr(styr),reprod));
  B(styr)=elem_prod(N_M(styr),wgt_m)+elem_prod(N_F(styr),wgt_f);
  totB(styr)=sum(B(styr));
```

```
//Rest of years
  for (iyear=styr; iyear<endyr; iyear++)
  {

    N_F(iyear+1)(2,nages)=++elem_prod(N_F(iyear)(1,nages-1),(mfexp(-1.*Z_F(iyear)(1,nages-1))));
    N_F(iyear+1,nages)+=N_F(iyear,nages)*mfexp(-1.*Z_F(iyear,nages));//plus group
    N_F_mdyr(iyear+1)(2,nages)=elem_prod(N_F(iyear+1)(2,nages),(mfexp(-1.*(Z_F(iyear+1)(2,nages))/2.0))); //mdyr
    N_F_mdyr(iyear+1,1)=0.0;
    N_M(iyear+1)(2,nages)=++elem_prod(N_M(iyear)(1,nages-1),(mfexp(-1.*Z_M(iyear)(1,nages-1))));
    N_M(iyear+1,nages)+=N_M(iyear,nages)*mfexp(-1.*Z_M(iyear,nages));//plus group
    N_M_mdyr(iyear+1)(2,nages)=elem_prod(N_M(iyear+1)(2,nages),(mfexp(-1.*(Z_M(iyear+1)(2,nages))/2.0))); //mdyr
    N_M_mdyr(iyear+1,1)=0.0;
    SSB(iyear+1)=sum(elem_prod(N_F_mdyr(iyear+1),reprod));

    for(iage=1;iage<=nages;iage++){
      prop_f(iyear,iage)=N_F(iyear,iage)/(N_F(iyear,iage)+N_M(iyear,iage)+dzero_dum);
    }

    if(iyear<(styr_rec_dev-1)) //recruitment follows S-R curve exactly
    {
        //add 0.00001 to avoid log(zero)
        N_F(iyear+1,1)=prop_f_a0*BiasCor*mfexp(log(((0.8*R0*steep*SSB(iyear+1))/(0.2*R0*spr_F0*
            (1.0-steep)+(steep-0.2)*SSB(iyear+1)))+dzero_dum));
    }
    else{
        N_F(iyear+1,1)=prop_f_a0*mfexp(log(((0.8*R0*steep*SSB(iyear+1))/(0.2*R0*spr_F0*
            (1.0-steep)+(steep-0.2)*SSB(iyear+1)))+dzero_dum)+log_rec_dev(iyear+1));
      // cout << "reading out Fem abundance" <<N_F(endyr+1,1) << endl;
    }


    N_M(iyear+1,1)=N_F(iyear+1,1)*(1./prop_f_a0-1.);  //this is how it works out algebraically
    N_M_mdyr(iyear+1,1)=N_M(iyear+1,1)*mfexp(-1.*(Z_M(iyear+1,1)/2));
    N_F_mdyr(iyear+1,1)=N_F(iyear+1,1)*mfexp(-1.*(Z_F(iyear+1,1)/2));
    B(iyear+1)=elem_prod(N_F(iyear+1),wgt_f)+elem_prod(N_M(iyear+1),wgt_m);
    totB(iyear+1)=sum(B(iyear+1));
  }

    //last year (projection) has no recruitment variability
    N_M(endyr+1)(2,nages)=++elem_prod(N_M(endyr)(1,nages-1),(mfexp(-1.*Z_M(endyr)(1,nages-1))));
    N_M(endyr+1,nages)+=N_M(endyr,nages)*mfexp(-1.*Z_M(endyr,nages));//plus group
    N_M_mdyr(endyr+1)(1,nages)=elem_prod(N_M(endyr+1)(1,nages),(mfexp(-1.*(Z_M(endyr)(1,nages))/2.0))); //mdyr
    N_F(endyr+1)(2,nages)=++elem_prod(N_F(endyr)(1,nages-1),(mfexp(-1.*Z_F(endyr)(1,nages-1))));
    N_F(endyr+1,nages)+=N_F(endyr,nages)*mfexp(-1.*Z_F(endyr,nages));//plus group
    N_F_mdyr(endyr+1)(1,nages)=elem_prod(N_F(endyr+1)(1,nages),(mfexp(-1.*(Z_F(endyr)(1,nages))/2.0))); //mdyr
    SSB_extra=sum(elem_prod(N_F_mdyr(endyr+1),reprod));
    N_F(endyr+1,1)=prop_f_a0*mfexp(log(((0.8*R0*steep*SSB_extra)/(0.2*R0*spr_F0*
                (1.0-steep)+(steep-0.2)*SSB_extra))+dzero_dum));
    N_M(endyr+1,1)=(1.-prop_f_a0)*mfexp(log(((0.8*R0*steep*SSB_extra)/(0.2*R0*spr_F0*
                (1.0-steep)+(steep-0.2)*SSB_extra))+dzero_dum));
    B(endyr+1)=elem_prod(N_M(endyr+1),wgt_m)+elem_prod(N_F(endyr+1),wgt_f);
    totB(endyr+1)=sum(B(endyr+1));

//Recruitment time series
  rec=column(N_F,1)+column(N_M,1);

//Benchmark parameters
  S1S0=SSB(styr)/S0;
  popstatus=SSB(endyr)/S0;

FUNCTION get_landings_numbers //Baranov catch eqn
  for (iyear=styr; iyear<=endyr; iyear++)
  {
    for (iage=1; iage<=nages; iage++)
    {
      C_HL_M(iyear,iage)=N_M(iyear,iage)*F_HL_M(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
     //  cout << "Handline catches, Male" <<C_HL_M << endl;
      C_HL_F(iyear,iage)=N_F(iyear,iage)*F_HL_F(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
      C_PN_M(iyear,iage)=N_M(iyear,iage)*F_PN_M(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
      C_PN_F(iyear,iage)=N_F(iyear,iage)*F_PN_F(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
      C_GN_M(iyear,iage)=N_M(iyear,iage)*F_GN_M(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
      C_GN_F(iyear,iage)=N_F(iyear,iage)*F_GN_F(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
      //  cout << "C_GN_F" <<C_GN_F << endl;
      C_CN_M(iyear,iage)=N_M(iyear,iage)*F_CN_M(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
      //  cout << "C_GN_F" <<C_GN_F << endl;
      C_CN_F(iyear,iage)=N_F(iyear,iage)*F_CN_F(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
      C_MRFSS_M(iyear,iage)=N_M(iyear,iage)*F_MRFSS_M(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
      //  cout << "MRFSS catches, Male" <<C_MRFSS_M << endl;
      C_MRFSS_F(iyear,iage)=N_F(iyear,iage)*F_MRFSS_F(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
      //    cout << "MRFSS catches, Female" <<C_MRFSS_F << endl;
      C_shrimp_M(iyear,iage)=N_M(iyear,iage)*F_shrimp(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
      //    cout << "Shrimp bycatch, Male" <<C_shrimp_M << endl;
      C_shrimp_F(iyear,iage)=N_F(iyear,iage)*F_shrimp(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
    }
  }

FUNCTION get_landings_wgt
//---Predicted landings-------------------------
  for (iyear=styr; iyear<=endyr; iyear++)
  {
    L_HL(iyear)=elem_prod(C_HL_M(iyear),wgt_land)+elem_prod(C_HL_F(iyear),wgt_land);           //in mt
    L_HL_klb(iyear)=elem_prod(C_HL_F(iyear),wgt_klb_land)+elem_prod(C_HL_M(iyear),wgt_klb_land); //in 1000 lb
```

```
    L_PN(iyear)=elem_prod(C_PN_M(iyear),wgt_land)+elem_prod(C_PN_F(iyear),wgt_land);   //in mt
    L_PN_klb(iyear)=elem_prod(C_PN_F(iyear),wgt_klb_land)+elem_prod(C_PN_M(iyear),wgt_klb_land); //in 1000 lb

    L_GN(iyear)=elem_prod(C_GN_M(iyear),wgt_land)+elem_prod(C_GN_F(iyear),wgt_land); //in mt
    L_GN_klb(iyear)=elem_prod(C_GN_F(iyear),wgt_klb_land)+elem_prod(C_GN_M(iyear),wgt_klb_land); //in 1000 lb

    L_CN(iyear)=elem_prod(C_CN_M(iyear),wgt_land)+elem_prod(C_CN_F(iyear),wgt_land); //in mt
    L_CN_klb(iyear)=elem_prod(C_CN_F(iyear),wgt_klb_land)+elem_prod(C_CN_M(iyear),wgt_klb_land); //in 1000 lb

    L_MRFSS(iyear)=elem_prod(C_MRFSS_M(iyear),wgt_land)+elem_prod(C_MRFSS_F(iyear),wgt_land); //in mt
    L_MRFSS_klb(iyear)=elem_prod(C_MRFSS_F(iyear),wgt_klb_land)+elem_prod(C_MRFSS_M(iyear),wgt_klb_land); //in 1000 lb

    L_shrimp(iyear)=elem_prod(C_shrimp_F(iyear),wgt_land)+elem_prod(C_shrimp_M(iyear),wgt_land); //in mt
  }
 // for (iyear=styr_MRFSS_LL; iyear<+endyr_MRFSS_LL; iyear++)
 // {   L_MRFSS(iyear)=elem_prod(C_MRFSS_M(iyear),wgt_m)+elem_prod(C_MRFSS_F(iyear),wgt_f); //in mt
 //     L_MRFSS_klb(iyear)=elem_prod(C_MRFSS_F(iyear),wgt_klb_f)+elem_prod(C_MRFSS_M(iyear),wgt_klb_m); //in 1000 lb
 // }
 // for (iyear=styr_shrimp_B; iyear<+endyr_shrimp_B; iyear++)
 //     L_shrimp(iyear)=elem_prod(C_shrimp_F(iyear),wgt_f)+elem_prod(C_shrimp_M(iyear),wgt_m); //in mt
 // }
  for (iyear=styr_HL_L; iyear<=endyr_HL_L; iyear++){
     pred_HL_L(iyear)=sum(L_HL_klb(iyear));
  }
  for (iyear=styr_PN_L; iyear<=endyr_PN_L; iyear++){
     pred_PN_L(iyear)=sum(L_PN_klb(iyear));
  }
  for (iyear=styr_GN_L; iyear<=endyr_GN_L; iyear++){
     pred_GN_L(iyear)=sum(L_GN_klb(iyear));
  }
  for (iyear=styr_CN_L; iyear<=endyr_CN_L; iyear++){
     pred_CN_L(iyear)=sum(L_CN_klb(iyear));
  }
  for(iyear=styr_MRFSS_L;iyear<=endyr_MRFSS_L;iyear++){
     pred_MRFSS_L(iyear)=sum(C_MRFSS_F(iyear)+C_MRFSS_M(iyear))/1000.0; //in 1000's (numbers)
  }
  for(iyear=styr_shrimp_B;iyear<=endyr_shrimp_B;iyear++){
     pred_shrimp_B(iyear)=sum(C_shrimp_F(iyear)+C_shrimp_M(iyear))/1000.0; //in 1000's (numbers)
  }
  //log_pred_shrimp_B=log(pred_shrimp_B(styr_shrimp_B_fit,endyr_shrimp_B_fit));
  //pred_shrimp_geomean=mfexp(sum(log_pred_shrimp_B)/(endyr_shrimp_B_fit-styr_shrimp_B_fit+1.0));

FUNCTION get_dead_discards //Baranov catch eqn
  //dead discards at age (number fish)

  for (iyear=styr_MRFSS_D; iyear<=endyr_MRFSS_D; iyear++){
    for (iage=1; iage<=nages; iage++){
      C_MRFSS_D_F(iyear,iage)=N_F(iyear,iage)*F_MRFSS_D_F(iyear,iage)*
        (1.-mfexp(-1.*Z_F(iyear,iage)))/Z_F(iyear,iage);
      C_MRFSS_D_M(iyear,iage)=N_M(iyear,iage)*F_MRFSS_D_M(iyear,iage)*
        (1.-mfexp(-1.*Z_M(iyear,iage)))/Z_M(iyear,iage);
    }
    L_MRFSS_D(iyear)=elem_prod(C_MRFSS_D_F(iyear),wgt_land)+elem_prod(C_MRFSS_D_M(iyear),wgt_land);       //discards in 1000lb whole weight
    pred_MRFSS_D(iyear)=(sum(C_MRFSS_D_M(iyear))+sum(C_MRFSS_D_F(iyear)))/1000.0; //pred annual dead discards in 1000s
  }

FUNCTION get_indices
 //FL hand lines cpue
  for (iyear=styr_FL_HL_cpue; iyear<=endyr_FL_HL_cpue; iyear++)
  {   //index in whole wgt (lb) units, wgt_klb in 1000 lb, but the multiplier (1000) is absorbed by q
      N_FL_HL(iyear)=elem_prod(elem_prod(N_F_mdyr(iyear),sel_wgted_tot_F),wgt_klb_f)+
       elem_prod(elem_prod(N_M_mdyr(iyear),sel_wgted_tot_M),wgt_klb_m);
      pred_FL_HL_cpue(iyear)=mfexp(log_q_FL_HL)*sum(N_FL_HL(iyear));
  }
  //MRFSS cpue  //includes discards
  for (iyear=styr_MRFSS_cpue; iyear<=endyr_MRFSS_cpue; iyear++)
  {   //index in 1000's (numbers)
      N_MRFSS(iyear)=elem_prod(N_F_mdyr(iyear),sel_MRFSS_keep_F)+elem_prod(N_F_mdyr(iyear),sel_MRFSS_D_F)+
       elem_prod(N_M_mdyr(iyear),sel_MRFSS_keep_M)+elem_prod(N_M_mdyr(iyear),sel_MRFSS_D_M);
      pred_MRFSS_cpue(iyear)=mfexp(log_q_MRFSS)*sum(N_MRFSS(iyear));
  }

   //SEAMAP YOY cpue
  for (iyear=styr_SMAP_YOY_cpue; iyear<=endyr_SMAP_YOY_cpue; iyear++)
  {   //index in 1000's (numbers)
      pred_SMAP_YOY_cpue(iyear)=mfexp(log_q_SMAP_YOY)*(N_F(iyear,1)+N_M(iyear,1));
  }
  //SEAMAP 1YR cpue
  //for (iyear=styr_SMAP_1YR_cpue; iyear<=endyr_SMAP_1YR_cpue; iyear++)
  //{   //index in 1000's (numbers)
  //    pred_SMAP_1YR_cpue(iyear)=mfexp(log_q_SMAP_1YR)*(N_F(iyear,2)+N_M(iyear,2));
  //}


FUNCTION get_age_comps
   //Hand lines
  for (iyear=1;iyear<=nyr_HL_agec;iyear++)
  {
    pred_HL_agec(iyear)=(C_HL_F(yrs_HL_agec(iyear))+C_HL_M(yrs_HL_agec(iyear)))/
                        (sum(C_HL_F(yrs_HL_agec(iyear))+C_HL_M(yrs_HL_agec(iyear)))+dzero_dum);
  }
  pred_HL_agec=pred_HL_agec*age_error_matrix;
 //  cout << "HL age comps" <<pred_HL_agec << endl;

  //Pound nets
  for (iyear=1;iyear<=nyr_PN_agec;iyear++)
  {
    pred_PN_agec(iyear)=(C_PN_F(yrs_PN_agec(iyear))+C_PN_M(yrs_PN_agec(iyear)))/
                          sum(C_PN_F(yrs_PN_agec(iyear))+C_PN_M(yrs_PN_agec(iyear)));
  }
  pred_PN_agec=pred_PN_agec*age_error_matrix;
 // cout << "PN age comps" <<pred_PN_agec << endl;
 //Gill nets
```

```
  for (iyear=1;iyear<=nyr_GN_agec;iyear++)
  {
    pred_GN_agec(iyear)=(C_GN_F(yrs_GN_agec(iyear))+C_GN_M(yrs_GN_agec(iyear)))/
                        sum(C_GN_F(yrs_GN_agec(iyear))+C_GN_M(yrs_GN_agec(iyear)));
  }
//  cout << "GN age comps" <<pred_GN_agec << endl;
  //Cast nets
  for (iyear=1;iyear<=nyr_CN_agec;iyear++)
  {
    pred_CN_agec(iyear)=(C_CN_F(yrs_CN_agec(iyear))+C_CN_M(yrs_CN_agec(iyear)))/
                        sum(C_CN_F(yrs_CN_agec(iyear))+C_CN_M(yrs_CN_agec(iyear)));
  }
//  cout << "CN age comps" <<pred_CN_agec << endl;
  //MRFSS
  for (iyear=1;iyear<=nyr_MRFSS_agec;iyear++)
  {
    pred_MRFSS_agec(iyear)=(C_MRFSS_F(yrs_MRFSS_agec(iyear))+C_MRFSS_M(yrs_MRFSS_agec(iyear)))/
                        sum(C_MRFSS_F(yrs_MRFSS_agec(iyear))+C_MRFSS_M(yrs_MRFSS_agec(iyear)));
  }
  //cout << "MRFSS age comps" <<pred_MRFSS_agec << endl;
//--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
FUNCTION get_sel_weighted_current //get average of most recent 3 years selectivity for MSY calcs
  F_temp_sum=0.0;
  F_temp_sum+=mfexp((3.0*log_avg_F_HL+sum(log_F_dev_HL(endyr-2,endyr)))/3.0);
  F_temp_sum+=mfexp((3.0*log_avg_F_PN+sum(log_F_dev_PN(endyr-2,endyr)))/3.0);
  F_temp_sum+=mfexp((3.0*log_avg_F_GN+sum(log_F_dev_GN(endyr-2,endyr)))/3.0);
  F_temp_sum+=mfexp((3.0*log_avg_F_CN+sum(log_F_dev_CN(endyr-2,endyr)))/3.0);
  F_temp_sum+=mfexp((3.0*log_avg_F_MRFSS+sum(log_F_dev_MRFSS(endyr-2,endyr)))/3.0);

  F_temp_sum+=mfexp((3.0*log_avg_F_MRFSS_D+sum(log_F_dev_MRFSS_D(endyr-2,endyr)))/3.0);

  F_temp_sum+=mfexp((3.0*log_avg_F_shrimp+sum(log_F_dev_shrimp(endyr-2,endyr)))/3.0);

  F_HL_prop=mfexp((3.0*log_avg_F_HL+sum(log_F_dev_HL(endyr-2,endyr)))/3.0)/F_temp_sum;
  F_PN_prop=mfexp((3.0*log_avg_F_PN+sum(log_F_dev_PN(endyr-2,endyr)))/3.0)/F_temp_sum;
  F_GN_prop=mfexp((3.0*log_avg_F_GN+sum(log_F_dev_GN(endyr-2,endyr)))/3.0)/F_temp_sum;
  F_CN_prop=mfexp((3.0*log_avg_F_CN+sum(log_F_dev_CN(endyr-2,endyr)))/3.0)/F_temp_sum;
  F_MRFSS_prop=mfexp((3.0*log_avg_F_MRFSS+sum(log_F_dev_MRFSS(endyr-2,endyr)))/3.0)/F_temp_sum;

  F_MRFSS_D_prop=mfexp((3.0*log_avg_F_MRFSS_D+sum(log_F_dev_MRFSS_D(endyr-2,endyr)))/3.0)/F_temp_sum;

  F_shrimp_prop=mfexp((3.0*log_avg_F_shrimp+sum(log_F_dev_shrimp(endyr-2,endyr)))/3.0)/F_temp_sum;

  sel_wgted_L_F=F_HL_prop*sel_HL_keep_F+F_PN_prop*sel_PN_F+F_GN_prop*sel_GN_keep_F+F_CN_prop*sel_CN_F+F_MRFSS_prop*sel_MRFSS_keep_F;
  sel_wgted_D_F=F_MRFSS_D_prop*sel_MRFSS_D_F+F_shrimp_prop*sel_shrimp;
  sel_wgted_tot_F=sel_wgted_L_F+sel_wgted_D_F;

  sel_wgted_L_M=F_HL_prop*sel_HL_keep_M+F_PN_prop*sel_PN_M+F_GN_prop*sel_GN_keep_M+F_CN_prop*sel_CN_M+F_MRFSS_prop*sel_MRFSS_keep_M;
  sel_wgted_D_M=F_MRFSS_D_prop*sel_MRFSS_D_M+F_shrimp_prop*sel_shrimp;
  sel_wgted_tot_M=sel_wgted_L_M+sel_wgted_D_M;


FUNCTION get_msy

  //fill in Fs for per-recruit stuff
  F_msy.fill_seqadd(0,.0001);  //step size should be of inverse dimension to n_iter_msy

  //compute values as functions of F
  for(ff=1; ff<=n_iter_msy; ff++){ //commented out 'int' before ff=1
    //int ff=1001;
    //uses fishery-weighted F's
    Z_age_msy_F=0.0;
    F_L_age_msy_F=0.0;
    F_D_age_msy_F=0.0;
    Z_age_msy_M=0.0;
    F_L_age_msy_M=0.0;
    F_D_age_msy_M=0.0;

    F_L_age_msy_F=F_msy(ff)*sel_wgted_L_F;
    F_D_age_msy_F=F_msy(ff)*sel_wgted_D_F;
    Z_age_msy_F=M+F_L_age_msy_F+F_D_age_msy_F;
    F_L_age_msy_M=F_msy(ff)*sel_wgted_L_M;
    F_D_age_msy_M=F_msy(ff)*sel_wgted_D_M;
    Z_age_msy_M=M+F_L_age_msy_M+F_D_age_msy_M;

    N_age_msy_F(1)=prop_f_a0;
    for (iage=2; iage<=nages; iage++)
    {
      N_age_msy_F(iage)=N_age_msy_F(iage-1)*mfexp(-1.*Z_age_msy_F(iage-1));
    }
    N_age_msy_F(nages)=N_age_msy_F(nages)/(1.0-mfexp(-1.*Z_age_msy_F(nages)));
    N_age_msy_mdyr_F(1,(nages-1))=elem_prod(N_age_msy_F(1,(nages-1)),
                                mfexp((-1.*Z_age_msy_F(1,(nages-1)))/2.0));
    N_age_msy_mdyr_F(nages)=(N_age_msy_mdyr_F(nages-1)*
                           (mfexp(-1.*(Z_age_msy_F(nages-1)/2 + Z_age_msy_F(nages)/2) )))
                           /(1.0-mfexp(-1.*Z_age_msy_F(nages)));

    N_age_msy_M(1)=1.-prop_f_a0;
    for (iage=2; iage<=nages; iage++)
    {
      N_age_msy_M(iage)=N_age_msy_M(iage-1)*mfexp(-1.*Z_age_msy_M(iage-1));
    }
    N_age_msy_M(nages)=N_age_msy_M(nages)/(1.0-mfexp(-1.*Z_age_msy_M(nages)));
    N_age_msy_mdyr_M(1,(nages-1))=elem_prod(N_age_msy_M(1,(nages-1)),
                                mfexp((-1.*Z_age_msy_M(1,(nages-1)))/2.0));
    N_age_msy_mdyr_M(nages)=(N_age_msy_mdyr_M(nages-1)*
                           (mfexp(-1.*(Z_age_msy_M(nages-1)/2 + Z_age_msy_M(nages)/2) )))
                           /(1.0-mfexp(-1.*Z_age_msy_M(nages)));

    for(iage=1;iage<=nages;iage++){
      prop_f_F0(iage)=N_age_msy_F(iage)/(N_age_msy_F(iage)+N_age_msy_M(iage));  //not really at F=0 here; just using vector
    }

    spr_msy(ff)=sum(elem_prod(N_age_msy_mdyr_F,reprod));
```

```
    //Compute equilibrium values of R (including bias correction), SSB and Yield at each F
    R_eq(ff)=(R0/((5.0*steep-1.0)*spr_msy(ff)))*
                (BiasCor*4.0*steep*spr_msy(ff)-spr_F0*(1.0-steep));
    if (R_eq(ff)<dzero_dum) {R_eq(ff)=dzero_dum;}
    N_age_msy_F*=R_eq(ff); //proportion female/male already accounted for
    N_age_msy_mdyr_F*=R_eq(ff);
    N_age_msy_M*=R_eq(ff);
    N_age_msy_mdyr_M*=R_eq(ff);

    for (iage=1; iage<=nages; iage++){
      C_age_msy_F(iage)=N_age_msy_F(iage)*(F_L_age_msy_F(iage)/Z_age_msy_F(iage))*
                        (1.-mfexp(-1.0*Z_age_msy_F(iage)));
      C_age_msy_M(iage)=N_age_msy_M(iage)*(F_L_age_msy_M(iage)/Z_age_msy_M(iage))*
                        (1.-mfexp(-1.0*Z_age_msy_M(iage)));
      D_age_msy_F(iage)=N_age_msy_F(iage)*(F_D_age_msy_F(iage)/Z_age_msy_F(iage))*
                        (1.-mfexp(-1.0*Z_age_msy_F(iage)));
      D_age_msy_M(iage)=N_age_msy_M(iage)*(F_D_age_msy_M(iage)/Z_age_msy_M(iage))*
                        (1.-mfexp(-1.0*Z_age_msy_M(iage)));
    }

    SSB_eq(ff)=sum(elem_prod(N_age_msy_mdyr_F,reprod));
    B_eq(ff)=sum(elem_prod(N_age_msy_F,wgt_f))+sum(elem_prod(N_age_msy_M,wgt_m));
    L_eq(ff)=sum(elem_prod(C_age_msy_F,wgt_land))+sum(elem_prod(C_age_msy_M,wgt_land));
    E_eq(ff)=(sum(C_age_msy_F(E_age_st,nages))+sum(C_age_msy_M(E_age_st,nages)));
    E_eq(ff)/=(sum(N_age_msy_F(E_age_st,nages))+sum(N_age_msy_M(E_age_st,nages)));
    D_eq(ff)=(sum(D_age_msy_F)+sum(D_age_msy_M));//1000.0;

  }

  msy_out=max(L_eq);

  for(ff=1; ff<=n_iter_msy; ff++)
  {
   if(L_eq(ff) == msy_out)
      {
        SSB_msy_out=SSB_eq(ff);
        B_msy_out=B_eq(ff);
        R_msy_out=R_eq(ff);
        D_msy_out=D_eq(ff);
        E_msy_out=E_eq(ff);
        F_msy_out=F_msy(ff);
        spr_msy_out=spr_msy(ff);
      }
  }

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
FUNCTION get_miscellaneous_stuff

  sigma_rec_dev=sqrt(var_rec_dev+0.0001); //sample SD of predicted residuals (may not equal rec_sigma)

  //compute total catch-at-age and landings
  C_total=C_HL_F+C_HL_M+C_PN_F+C_PN_M+C_GN_M+C_GN_F+C_CN_M+C_CN_F+C_MRFSS_M+C_MRFSS_F; //catch in number fish
  L_total=L_HL+L_PN+L_GN+L_CN+L_MRFSS;   //landings in mt whole weight
  D_total=L_MRFSS_D; //total discards in mt whole weight
  B_total=L_shrimp;

  //compute exploitation rate of age E_age_st +
  for(iyear=styr; iyear<=endyr; iyear++)
  {
    E(iyear)=(sum(C_total(iyear)(E_age_st,nages)))/(sum(N_F(iyear)(E_age_st,nages))+sum(N_M(iyear)(E_age_st,nages)));
    L_total_yr(iyear)=sum(L_total(iyear));
    B_total_yr(iyear)=sum(B_total(iyear));
    D_total_yr(iyear)=sum(D_total(iyear));
  }

  steep_sd=steep;
  fullF_sd=fullF;
  E_sd=E;

  if(E_msy_out>0)
    {
      EdE_msy=E/E_msy_out;
      EdE_msy_end=EdE_msy(endyr);
    }
  if(F_msy_out>0)
    {
      FdF_msy=fullF/F_msy_out;
      FdF_msy_end=FdF_msy(endyr);
    }
  if(SSB_msy_out>0)
    {
      SdSSB_msy=SSB/SSB_msy_out;
      SdSSB_msy_end=SdSSB_msy(endyr);
    }

   //fill in log recruitment deviations for yrs they are nonzero
   for(iyear=styr_rec_dev; iyear<=endyr; iyear++)
   {
    log_dev_R(iyear)=log_rec_dev(iyear);
   }

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
FUNCTION get_per_recruit_stuff

  //static per-recruit stuff

  for(iyear=styr; iyear<=endyr; iyear++)
  {
    N_age_spr_F(1)=prop_f_a0;
    for(iage=2; iage<=nages; iage++)
    {
      N_age_spr_F(iage)=N_age_spr_F(iage-1)*mfexp(-1.*Z_F(iyear,iage-1));
    }
```

```
    N_age_spr_F(nages)=N_age_spr_F(nages)/(1.0-mfexp(-1.*Z_F(iyear,nages)));
    N_age_spr_mdyr_F(1,(nages-1))=elem_prod(N_age_spr_F(1,(nages-1)),
                                mfexp(-1.*Z_F(iyear)(1,(nages-1))/2.0));
    N_age_spr_mdyr_F(nages)=(N_age_spr_mdyr_F(nages-1)*
                          (mfexp(-1.*(Z_F(iyear)(nages-1)/2.0 + Z_F(iyear)(nages)/2.0) )))
                          /(1.0-mfexp(-1.*Z_F(iyear)(nages)));
    spr_static(iyear)=sum(elem_prod(N_age_spr_mdyr_F,reprod))/spr_F0;
  }

  //fill in Fs for per-recruit stuff
  F_spr.fill_seqadd(0,.001);

  //compute SSB/R and YPR as functions of F
  for(int ff=1; ff<=n_iter_spr; ff++)
  {
    //uses fishery-weighted F's, same as in MSY calculations
    Z_age_spr_F=0.0;
    F_L_age_spr_F=0.0;
    Z_age_spr_M=0.0;
    F_L_age_spr_M=0.0;

    F_L_age_spr_F=F_spr(ff)*sel_wgted_L_F;
    F_L_age_spr_M=F_spr(ff)*sel_wgted_L_M;

    Z_age_spr_F=M+F_L_age_spr_F+F_spr(ff)*sel_wgted_D_F;
    Z_age_spr_M=M+F_L_age_spr_M+F_spr(ff)*sel_wgted_D_M;

    N_age_spr_F(1)=prop_f_a0;
    N_age_spr_M(1)=1.-prop_f_a0;
    for (iage=2; iage<=nages; iage++){
      N_age_spr_F(iage)=N_age_spr_F(iage-1)*mfexp(-1.*Z_age_spr_F(iage-1));
      N_age_spr_M(iage)=N_age_spr_M(iage-1)*mfexp(-1.*Z_age_spr_M(iage-1));
    }
    N_age_spr_F(nages)=N_age_spr_F(nages)/(1-mfexp(-1.*Z_age_spr_F(nages)));
    N_age_spr_M(nages)=N_age_spr_M(nages)/(1-mfexp(-1.*Z_age_spr_M(nages)));

    N_age_spr_mdyr_F(1,(nages-1))=elem_prod(N_age_spr_F(1,(nages-1)),
                                  mfexp((-1.*Z_age_spr_F(1,(nages-1)))/2.0));
    N_age_spr_mdyr_F(nages)=(N_age_spr_mdyr_F(nages-1)*
                          (mfexp(-1.*(Z_age_spr_F(nages-1)/2 + Z_age_spr_F(nages)/2) )))
                          /(1.0-mfexp(-1.*Z_age_spr_F(nages)));

    for(iage=1;iage<=nages;iage++){
      prop_f_F0(iage)=N_age_spr_F(iage)/(N_age_spr_F(iage)+N_age_spr_M(iage));  //not really at F=0 here; just using vector
    }
    spr_spr(ff)=sum(elem_prod(N_age_spr_mdyr_F,reprod));
    L_spr(ff)=0.0;
    for (iage=1; iage<=nages; iage++)
    {
      C_age_spr_F(iage)=N_age_spr_F(iage)*(F_L_age_spr_F(iage)/Z_age_spr_F(iage))*
                        (1.-mfexp(-1.*Z_age_spr_F(iage)));
      C_age_spr_M(iage)=N_age_spr_M(iage)*(F_L_age_spr_M(iage)/Z_age_spr_M(iage))*
                        (1.-mfexp(-1.*Z_age_spr_M(iage)));
      L_spr(ff)+=(C_age_spr_M(iage)*wgt_m(iage)+C_age_spr_F(iage)*wgt_land(iage));
    }
    E_spr(ff)=(sum(C_age_spr_M(E_age_st,nages))+sum(C_age_spr_F(E_age_st,nages)))/
              (sum(N_age_spr_M(E_age_st,nages))+sum(N_age_spr_F(E_age_st,nages)));

  }

//-----------------------------------------------------------------------------------------------------------------------------------------------------------

FUNCTION evaluate_objective_function
  fval=0.0;
  fval_unwgt=0.0;

//---likelihoods---------------------------
// fval=square(x_dum-3.0);

//---Indices-------------------------------
  f_FL_HL_cpue=0.0;
  for (iyear=styr_FL_HL_cpue; iyear<=endyr_FL_HL_cpue; iyear++)
  {
    f_FL_HL_cpue+=square(log((pred_FL_HL_cpue(iyear)+dzero_dum)/
        (obs_FL_HL_cpue(iyear)+dzero_dum)))/(2.0*square(FL_HL_cpue_cv(iyear)));
  }
  fval+=w_I_FL_HL*f_FL_HL_cpue;
  fval_unwgt+=f_FL_HL_cpue;

  f_MRFSS_cpue=0.0;
  for (iyear=styr_MRFSS_cpue; iyear<=endyr_MRFSS_cpue; iyear++)
  {
    f_MRFSS_cpue+=square(log((pred_MRFSS_cpue(iyear)+dzero_dum)/
        (obs_MRFSS_cpue(iyear)+dzero_dum)))/(2.0*square(MRFSS_cpue_cv(iyear)));
  }
  //cout<<w_I_MRFSS<<endl<<f_MRFSS_cpue<<endl<<endl;
  fval+=w_I_MRFSS*f_MRFSS_cpue;
  fval_unwgt+=f_MRFSS_cpue;

  f_SMAP_YOY_cpue=0.0;
  for (iyear=styr_SMAP_YOY_cpue; iyear<=endyr_SMAP_YOY_cpue; iyear++)
  {
    f_SMAP_YOY_cpue+=square(log((pred_SMAP_YOY_cpue(iyear)+dzero_dum)/
        (obs_SMAP_YOY_cpue(iyear)+dzero_dum)))/(2.0*square(SMAP_YOY_cpue_cv(iyear)));
  }
  fval+=w_I_SMAP_YOY*f_SMAP_YOY_cpue;
  fval_unwgt+=f_SMAP_YOY_cpue;

  //f_SMAP_1YR_cpue=0.0;
  //for (iyear=styr_SMAP_1YR_cpue; iyear<=endyr_SMAP_1YR_cpue; iyear++)
  //{
  //  f_SMAP_1YR_cpue+=square(log((pred_SMAP_1YR_cpue(iyear)+dzero_dum)/
  //      (obs_SMAP_1YR_cpue(iyear)+dzero_dum)))/(2.0*square(SMAP_1YR_cpue_cv(iyear)));
  //}
```

```
  //fval+=w_I_SMAP_1YR*f_SMAP_1YR_cpue;
  //fval_unwgt+=f_SMAP_1YR_cpue;

//---Landings-------------------------------

  f_HL_L=0.0; //in 1000s total pounds
  for (iyear=styr_HL_L; iyear<=endyr_HL_L; iyear++)
  {
      f_HL_L+=square(log((pred_HL_L(iyear)+dzero_dum)/
        (obs_HL_L(iyear)+dzero_dum)))/(2.0*square(HL_L_cv(iyear)));
  }
  fval+=w_L*f_HL_L;
  fval_unwgt+=f_HL_L;

  f_PN_L=0.0; //in 1000s total pounds
  for (iyear=styr_PN_L; iyear<=endyr_PN_L; iyear++)
  {
       f_PN_L+=square(log((pred_PN_L(iyear)+dzero_dum)/
        (obs_PN_L(iyear)+dzero_dum)))/(2.0*square(PN_L_cv(iyear)));
  }
  fval+=w_L*f_PN_L;
  fval_unwgt+=f_PN_L;

  f_GN_L=0.0; //in 1000s total pounds
  for (iyear=styr_GN_L; iyear<=endyr_GN_L; iyear++)
  {
      f_GN_L+=square(log((pred_GN_L(iyear)+dzero_dum)/
        (obs_GN_L(iyear)+dzero_dum)))/(2.0*square(GN_L_cv(iyear)));
  }
  fval+=w_L*f_GN_L;
  fval_unwgt+=f_GN_L;

  f_CN_L=0.0; //in 1000s total pounds
  for (iyear=styr_CN_L; iyear<=endyr_CN_L; iyear++)
  {
    f_CN_L+=square(log((pred_CN_L(iyear)+dzero_dum)/
        (obs_CN_L(iyear)+dzero_dum)))/(2.0*square(CN_L_cv(iyear)));
  }
  fval+=w_L*f_CN_L;
  fval_unwgt+=f_CN_L;

  f_MRFSS_L=0.0; //in 1000s - numbers
  for (iyear=styr_MRFSS_L; iyear<=endyr_MRFSS_L; iyear++)
  {
      f_MRFSS_L+=square(log((pred_MRFSS_L(iyear)+dzero_dum)/
        (obs_MRFSS_L(iyear)+dzero_dum)))/(2.0*square(MRFSS_L_cv(iyear)));
  }
  fval+=w_L*f_MRFSS_L;
  fval_unwgt+=f_MRFSS_L;

//---Discards & Bycatch--------------------------------

  f_MRFSS_D=0.0; //in 1000s (numbers)
  for (iyear=styr_MRFSS_D; iyear<=endyr_MRFSS_D; iyear++)
  {
      f_MRFSS_D+=square(log((pred_MRFSS_D(iyear)+dzero_dum)/
        (obs_MRFSS_D(iyear)+dzero_dum)))/(2.0*square(MRFSS_D_cv(iyear)));
  }
  fval+=w_D*f_MRFSS_D;
  fval_unwgt+=f_MRFSS_D;

  f_shrimp_B=0.0;
  for (iyear=styr_shrimp_B; iyear<=endyr_shrimp_B; iyear++)
  {
      f_shrimp_B+=square(log((pred_shrimp_B(iyear)+dzero_dum)/
        (obs_shrimp_B(iyear)+dzero_dum)))/(2.0*square(shrimp_B_cv(iyear)));
   }

  fval+=w_D*f_shrimp_B;
  //fval+=w_shrimp*f_shrimp_B;
  fval_unwgt+=f_shrimp_B;


///---Age comps--------------------------------
  f_HL_agec=0.0;
  f_HL_agec=lk_robust_multinomial(nsamp_HL_agec, pred_HL_agec, obs_HL_agec, nyr_HL_agec, double(nages), minSS_HL_agec, w_ac_HL);
  fval+=f_HL_agec;
  fval_unwgt+=f_HL_agec;

  f_PN_agec=0.0;
 // f_PN_agec=lk_robust_multinomial(nsamp_PN_agec, pred_PN_agec, obs_PN_agec, nyr_PN_agec, double(nages), minSS_PN_agec, w_ac_PN);
 // fval+=f_PN_agec;
 // fval_unwgt+=f_PN_agec;

  for (iyear=1; iyear<=nyr_PN_agec; iyear++)
  {
    f_PN_agec-=nsamp_PN_agec(iyear)*
        sum( elem_prod((obs_PN_agec(iyear)+dzero_dum),
            log(elem_div((pred_PN_agec(iyear)+dzero_dum),
               (obs_PN_agec(iyear)+dzero_dum)))));
  }
  fval+=w_ac_PN*f_PN_agec;
  fval_unwgt+=f_PN_agec;

  f_GN_agec=0.0;
  f_GN_agec=lk_robust_multinomial(nsamp_GN_agec, pred_GN_agec, obs_GN_agec, nyr_GN_agec, double(nages), minSS_GN_agec, w_ac_GN);
  fval+=f_GN_agec;
  fval_unwgt+=f_GN_agec;


  f_CN_agec=0.0;
  f_CN_agec=lk_robust_multinomial(nsamp_CN_agec, pred_CN_agec, obs_CN_agec, nyr_CN_agec, double(nages), minSS_CN_agec, w_ac_CN);
  fval+=f_CN_agec;
  fval_unwgt+=f_CN_agec;
```

```
  f_MRFSS_agec=0.0;
  f_MRFSS_agec=lk_robust_multinomial(nsamp_MRFSS_agec, pred_MRFSS_agec, obs_MRFSS_agec, nyr_MRFSS_agec, double(nages), minSS_MRFSS_agec, w_ac_MRFSS);
  fval+=f_MRFSS_agec;
  fval_unwgt+=f_MRFSS_agec;


//-----------Constraints and penalties--------------------------------
  f_N_dev=0.0;
    f_N_dev=pow(log_rec_dev(styr_rec_dev),2);
  for(iyear=(styr_rec_dev+1); iyear<=endyr; iyear++)
  {
    f_N_dev+=pow(log_rec_dev(iyear)-R_autocorr*log_rec_dev(iyear-1),2);
  }
  fval+=w_R*f_N_dev;


  f_N_dev_end=0.0; //last 3 yrs
    f_N_dev_end=norm2(log_rec_dev(endyr-2,endyr));
  fval+=w_R_end*f_N_dev_end;


  f_Fend_constraint=0.0; //last 3 yrs
  f_Fend_constraint=norm2(first_difference(fullF(endyr-2,endyr)));
  fval+=w_F*f_Fend_constraint;

  f_fullF_constraint=0.0;
  for (iyear=styr; iyear<=endyr; iyear++)
    {
     if (fullF(iyear)>3.0)
     {
     f_fullF_constraint+=square(fullF(iyear)-3.0);
     }
    }
  fval+=w_fullF*f_fullF_constraint;

//---Priors----------------------------------------------------
//neg_log_prior arguments: estimate, prior, variance, pdf type
//Variance input as a negative value is considered to be CV in arithmetic space (CV=-1 implies loose prior)
//pdf type 1=none, 2=lognormal, 3=normal, 4=beta
  f_priors=0.0;
  //f_priors+=neg_log_prior(len_cv_val, set_len_cv, square(set_len_cv_se), 2);
  //f_priors+=neg_log_prior(steep, set_steep, square(set_steep_se), 3);
  //f_priors+=neg_log_prior(rec_sigma,set_rec_sigma,square(set_rec_sigma_se),3);
  //f_priors+=neg_log_prior(R_autocorr,set_R_autocorr, 1.0, 1);
  //f_priors+=neg_log_prior(q_rate, set_q_rate, 0.00001+square(set_q_rate), 2);
  //f_priors+=neg_log_prior(F_init, set_F_init, -1.0 , 2);
  //f_priors+=neg_log_prior(M_constant, set_M_constant, square(set_M_constant_se), 2);

  //f_priors+=neg_log_prior(selpar_L50_HL, set_selpar_L50_HL, 1.0, 3);
  f_priors+=neg_log_prior(selpar_slope_HL,set_selpar_slope_HL, -0.25, 3);
  f_priors+=neg_log_prior(selpar_L501_PN, set_selpar_L501_PN, -0.15, 3);
  f_priors+=neg_log_prior(selpar_slope1_PN,set_selpar_slope1_PN, -0.15, 3);
  f_priors+=neg_log_prior(selpar_L502_PN, set_selpar_L502_PN, -0.25, 3);
  f_priors+=neg_log_prior(selpar_slope2_PN, set_selpar_slope2_PN, -0.25, 3);
//f_priors+=neg_log_prior(selpar_L50_GN, set_selpar_L50_GN, 1.0, 3);
  f_priors+=neg_log_prior(selpar_slope_GN,set_selpar_slope_GN, -0.15, 3);
  f_priors+=neg_log_prior(selpar_slope1_CN,set_selpar_slope1_CN, -0.25, 3);
  f_priors+=neg_log_prior(selpar_slope2_CN,set_selpar_slope2_CN, -0.25, 3);
  f_priors+=neg_log_prior(selpar_L502_CN, set_selpar_L502_CN, 1.0, 3);
//f_priors+=neg_log_prior(selpar_L501_MRFSS_keep, set_selpar_L501_MRFSS_keep, -0.25, 2);
  f_priors+=neg_log_prior(selpar_slope1_MRFSS_keep,set_selpar_slope1_MRFSS_keep, -0.25, 3);
  f_priors+=neg_log_prior(selpar_L502_MRFSS_keep, set_selpar_L502_MRFSS_keep, -0.25, 3);
  f_priors+=neg_log_prior(selpar_slope2_MRFSS_keep,set_selpar_slope2_MRFSS_keep, -0.25, 3);

  fval+=f_priors;

  if(!last_phase())
  {
    for (iyear=styr; iyear<=endyr; iyear++)
    {
      if(fullF(iyear)>1.0)
      {
        fval+=10*(mfexp(fullF(iyear)-1.0)-1.0);
      }
    }
  }
  //cout << "fval = " << fval << "  fval_data = " << fval_data << endl;
  //cout << endl;
  //cout << "f_cL_U = " << f_cL_cpue << " f_mm_U = " << f_mm_cpue << " f_cL_L = " << f_cL_L << " f_cH_L = " << f_cH_L
  //<< " f_rA_L = " << f_rA_L << endl;
  //cout << "f_cL_lenc = " << f_cL_lenc << " f_cH_lenc = " << f_cH_lenc << " f_rA_lenc = " << f_rA_lenc << " f_mm_lenc = " << f_mm_lenc << endl;
  //cout << "f_cL_agec = " << f_cL_agec << " f_cH_agec = " << f_cH_agec << " f_mm_agec = " << f_mm_agec << endl;
  //cout << "f_rec_dev = " << f_rec_dev << " f_rec_dev_early = " << f_rec_dev_early << " f_rec_dev_end = " << f_rec_dev_end <<  " f_rec_hist_dev = " << f_rec_historic_dev <<  " f_cL_RW = " << f_cL_RW_cpue << end
  //cout << endl;
//-------------------------------------------------------------------------------------
//-------------------------------------------------------------------------------------
FUNCTION dvar_vector logistic_exponential(const dvar_vector& ages, const dvariable& L50, const dvariable& slope, const dvariable& sigma, const dvariable& joint)
  //ages=vector of ages, L50=age at 50% sel (ascending limb), slope=rate of increase, sigma=controls rate of descent (descending)
  //joint=age to join curves
  RETURN_ARRAYS_INCREMENT();
  dvar_vector Sel_Tmp(ages.indexmin(),ages.indexmax());
  Sel_Tmp=1.0;
  for (iage=1; iage<=nages; iage++)
  {
   if (ages(iage)<joint) {Sel_Tmp(iage)=1./(1.+mfexp(-1.*slope*(ages(iage)-L50)));}
   if (ages(iage)>joint){Sel_Tmp(iage)=mfexp(-1.*square((ages(iage)-joint)/sigma));}
  }
  Sel_Tmp=Sel_Tmp/max(Sel_Tmp);
  RETURN_ARRAYS_DECREMENT();
  return Sel_Tmp;

//-------------------------------------------------------------------------------------
```

```
//Logistic function: 2 parameters
//FUNCTION dvar_vector logistic(const dvar_vector& ages, const dvariable& L50, const dvariable& slope)
//  //ages=vector of ages, L50=age at 50% selectivity, slope=rate of increase
//  RETURN_ARRAYS_INCREMENT();
//  dvar_vector Sel_Tmp(ages.indexmin(),ages.indexmax());
//  Sel_Tmp=1./(1.+mfexp(-1.*slope*(ages-L50))); //logistic;
//  RETURN_ARRAYS_DECREMENT();
//  return Sel_Tmp;

//--------------------------------------------------------------------------------
//Logistic function: 4 parameters
FUNCTION dvar_vector logistic_double(const dvar_vector& ages, const dvariable& L501, const dvariable& slope1, const dvariable& L502, const dvariable& slope2)
  //ages=vector of ages, L50=age at 50% selectivity, slope=rate of increase, L502=age at 50% decrease additive to L501, slope2=slope of decrease
  RETURN_ARRAYS_INCREMENT();



  dvar_vector Sel_Tmp(ages.indexmin(),ages.indexmax());
  Sel_Tmp=elem_prod( (1./(1.+mfexp(-1.*slope1*(ages-L501)))),(1.-(1./(1.+mfexp(-1.*slope2*(ages-(L501+L502)))))) );
  Sel_Tmp=Sel_Tmp/max(Sel_Tmp);
  RETURN_ARRAYS_DECREMENT();
  return Sel_Tmp;

//--------------------------------------------------------------------------------
//Jointed logistic function: 6 parameters (increasing and decreasing logistics joined at peak selectivity)
//FUNCTION dvar_vector logistic_joint(const dvar_vector& ages, const dvariable& L501, const dvariable& slope1, const dvariable& L502, const dvariable& slope2, const dvariable& satval, const dvariable& joint)
//  //ages=vector of ages, L501=age at 50% sel (ascending limb), slope1=rate of increase,L502=age at 50% sel (descending), slope1=rate of increase (ascending),
//  //satval=saturation value of descending limb, joint=location in age vector to join curves (may equal age or age + 1 if age-0 is included)
//  RETURN_ARRAYS_INCREMENT();
//  dvar_vector Sel_Tmp(ages.indexmin(),ages.indexmax());
//  Sel_Tmp=1.0;
//  for (iage=1; iage<=nages; iage++)
//  {
//   if (double(iage)<joint) {Sel_Tmp(iage)=1./(1.+mfexp(-1.*slope1*(ages(iage)-L501)));}
//   if (double(iage)>joint){Sel_Tmp(iage)=1.0-(1.0-satval)/(1.+mfexp(-1.*slope2*(ages(iage)-L502)));}
//  }
//  Sel_Tmp=Sel_Tmp/max(Sel_Tmp);
//  RETURN_ARRAYS_DECREMENT();
//  return Sel_Tmp;

//--------------------------------------------------------------------------------
//Double Gaussian function: 6 parameters (as in SS3)
FUNCTION dvar_vector gaussian_double(const dvar_vector& ages, const dvariable& peak, const dvariable& top, const dvariable& ascwid, const dvariable& deswid, const dvariable& init, const dvariable& final)
  //ages=vector of ages, peak=ascending inflection location (as logistic), top=width of plateau, ascwid=ascent width (as log(width))
  //deswid=descent width (as log(width))
  RETURN_ARRAYS_INCREMENT();
  dvar_vector Sel_Tmp(ages.indexmin(),ages.indexmax());
  dvar_vector sel_step1(ages.indexmin(),ages.indexmax());
  dvar_vector sel_step2(ages.indexmin(),ages.indexmax());
  dvar_vector sel_step3(ages.indexmin(),ages.indexmax());
  dvar_vector sel_step4(ages.indexmin(),ages.indexmax());
  dvar_vector sel_step5(ages.indexmin(),ages.indexmax());
  dvar_vector sel_step6(ages.indexmin(),ages.indexmax());
  dvar_vector pars_tmp(1,6); dvar_vector sel_tmp_iq(1,2);

  pars_tmp(1)=peak;
  pars_tmp(2)=peak+1.0+(0.99*ages(nages)-peak-1.0)/(1.0+mfexp(-top));
  pars_tmp(3)=mfexp(ascwid);
  pars_tmp(4)=mfexp(deswid);
  pars_tmp(5)=1.0/(1.0+mfexp(-init));
  pars_tmp(6)=1.0/(1.0+mfexp(-final));

  sel_tmp_iq(1)=mfexp(-(square(ages(1)-pars_tmp(1))/pars_tmp(3)));
  sel_tmp_iq(2)=mfexp(-(square(ages(nages)-pars_tmp(2))/pars_tmp(4)));

  sel_step1=mfexp(-(square(ages-pars_tmp(1))/pars_tmp(3)));
  sel_step2=pars_tmp(5)+(1.0-pars_tmp(5))*(sel_step1-sel_tmp_iq(1))/(1.0-sel_tmp_iq(1));
  sel_step3=mfexp(-(square(ages-pars_tmp(2))/pars_tmp(4)));
  sel_step4=1.0+(pars_tmp(6)-1.0)*(sel_step3-1.0)/(sel_tmp_iq(2)-1.0);
  sel_step5=1.0/ (1.0+mfexp(-(20.0* elem_div((ages-pars_tmp(1)), (1.0+sfabs(ages-pars_tmp(1))))  )));
  sel_step6=1.0/(1.0+mfexp(-(20.0*elem_div((ages-pars_tmp(2)),(1.0+sfabs(ages-pars_tmp(2))))  )));

  Sel_Tmp=elem_prod(sel_step2,(1.0-sel_step5))+
          elem_prod(sel_step5,((1.0-sel_step6)+ elem_prod(sel_step4,sel_step6)) );

  Sel_Tmp=Sel_Tmp/max(Sel_Tmp);
  RETURN_ARRAYS_DECREMENT();
  return Sel_Tmp;


//--------------------------------------------------------------------------------
//Likelihood contribution: multinomial
FUNCTION dvariable lk_robust_multinomial(const dvar_vector& nsamp, const dvar_matrix& pred_comp, const dvar_matrix& obs_comp, const double& ncomp, const dvariable& mbin, const double& minSS, const dvariable& wgt
  //nsamp=vector of N's, pred_comp=matrix of predicted comps, obs_comp=matrix of observed comps, ncomp = number of yrs in matrix, mbin=number of bins, minSS=min N threshold, wgt_dat=scaling of N's
  RETURN_ARRAYS_INCREMENT();
  dvariable LkvalTmp;
  LkvalTmp=0.0;
  dvar_matrix Eprime=elem_prod((1.0-obs_comp), obs_comp)+0.1/mbin; //E' of Francis 2011, p.1131
  dvar_vector nsamp_wgt=nsamp*wgt_dat;
  //cout<<nsamp_wgt<<endl;
  for (int ii=1; ii<=ncomp; ii++)
  {if (nsamp(ii)>=minSS)
    {LkvalTmp+= sum(0.5*log(Eprime(ii))-log(0.00001+mfexp(elem_div((-square(obs_comp(ii)-pred_comp(ii))) , (Eprime(ii)*2.0/nsamp_wgt(ii)) ))) );
    }
  }
  RETURN_ARRAYS_DECREMENT();
  return LkvalTmp;

//--------------------------------------------------------------------------------
//Likelihood contribution: priors
FUNCTION  dvariable neg_log_prior(dvariable pred, const double& prior, dvariable var, int pdf)
  //prior=prior point estimate, var=variance (if negative, treated as CV in arithmetic space), pred=predicted value, pdf=prior type (1=none, 2=lognormal, 3=normal, 4=beta)
    dvariable LkvalTmp;
    dvariable alpha, beta, ab_iq;
```

```
    LkvalTmp=0.0;
    // compute generic pdf's
    switch(pdf) {
        case 1: //option to turn off prior
          LkvalTmp=0.0;
          break;
        case 2: // lognormal
          if(prior<=0.0) cout << "YIKES: Don't use a lognormal distn for a negative prior" << endl;
          else if(pred<=0) LkvalTmp=huge_number;
          else {
            if(var<0.0) var=log(1.0+var*var) ;      // convert cv to variance on log scale
            LkvalTmp= 0.5*( square(log(pred/prior))/var + log(var) );
          }
        break;
        case 3: // normal
          if(var<0.0 && prior!=0.0) var=square(var*prior);       // convert cv to variance on observation scale
          else if(var<0.0 && prior==0.0) var=-var;               // cv not really appropriate if prior value equals zero
          LkvalTmp= 0.5*( square(pred-prior)/var + log(var) );
          break;
        case 4: // beta
          if(var<0.0) var=square(var*prior);          // convert cv to variance on observation scale
          if(prior<=0.0 || prior>=1.0) cout << "YIKES: Don't use a beta distn for a prior outside (0,1)" << endl;
          ab_iq=prior*(1.0-prior)/var - 1.0; alpha=prior*ab_iq; beta=(1.0-prior)*ab_iq;
          if(pred>=0 && pred<=1) LkvalTmp= (1.0-alpha)*log(pred)+(1.0-beta)*log(1.0-pred)-gammln(alpha+beta)+gammln(alpha)+gammln(beta);
          else LkvalTmp=huge_number;
          break;
        default: // no such prior pdf currently available
          cout << "The prior must be either 1(lognormal), 2(normal), or 3(beta)." << endl;
          cout << "Presently it is " << pdf << endl;
          exit(0);
    }
    return LkvalTmp;


REPORT_SECTION
//  cout<<"start report"<<endl;
  get_sel_weighted_current();
//  cout<<"got sel weighted"<<endl;
  get_msy();
//  cout<<"got msy"<<endl;
  get_miscellaneous_stuff();
//  cout<<"got misc stuff"<<endl;
  get_per_recruit_stuff();
//  cout<<"got per recruit"<<endl;
  cout << "BC Fmsy=" << F_msy_out<< "   BC SSBmsy=" << SSB_msy_out <<endl;
  cout<<"Pop status="<<SSB(endyr)/SSB_msy_out<<endl;
  cout<<"SSB last year "<<SSB(endyr);
  cout<<"SSB msy "<<SSB_msy_out<<endl;
  cout << "var_rec_resid="<<var_rec_dev<<endl;
//// cout << "x_dum="<<x_dum<<endl;

  report << "TotalLikelihood " << fval << endl;
  report<<" "<<endl;

  report << "Bias-corrected (BC) MSY stuff" << endl;
  report << "BC Fmsy " << F_msy_out << endl;
  report << "BC Emsy " << E_msy_out << endl;
  report << "BC SSBmsy " << SSB_msy_out << endl;
  report << "BC Rmsy " << R_msy_out << endl;
  report << "BC Bmsy " << B_msy_out << endl;
  report << "BC MSY " << msy_out << endl;
  report << "BC F/Fmsy " << fullF/F_msy_out << endl;
  report << "BC E/Emsy " << E/E_msy_out << endl;
  report << "BC SSB/SSBmsy " << SSB/SSB_msy_out << endl;
  report << "BC B/Bmsy " << totB/B_msy_out << endl;
  report << "BC Yield/MSY " << L_total_yr/msy_out <<endl;
  report << "BC F(2010)/Fmsy " << fullF(endyr)/F_msy_out << endl;
  report << "BC E(2010)/Emsy " << E(endyr)/E_msy_out << endl;
  report << "BC SSB(2010)/SSBmsy " << SSB(endyr)/SSB_msy_out << endl;
  report << "BC Predicted Landings(2010)/MSY " << L_total_yr(endyr)/msy_out <<endl;
  report  << " "<<endl;

  report << "Mortality and growth" << endl;
  report << "M "<<M<<endl;
  //report << "Linf,males="<<Linf_m << "   K, males=" <<K_m<<"   t0_males="<< t0_m<<endl;
  report << "mean length, females " << meanlen_f << endl;
  //report << "Linf,males="<<Linf_f << "   K, males=" <<K_f<<"   t0_fales="<< t0_f<<endl;
  report << "mean length, males " << meanlen_m << endl;
  report << "cv length " << len_cv << endl;
  report << "wgt, males" << wgt_m << endl;
  report << "wgt, females" << wgt_f << endl;
  report<<" "<<endl;

  report << "Stock-Recruit " << endl;
  report << "R0= " << R0 << endl;
  report << "Steepness= " << steep << endl;
  report << "spr_F0= " << spr_F0 << endl;
  report << "Recruits(R) " << rec << endl;
  report << "VirginSSB " << S0 << endl;
  report << "SSB(styr)/VirginSSB " << S1S0 << endl;
  report << "SSB(2010)/VirginSSB " << popstatus << endl;
  report << "SSB " << SSB << endl;
  report << "Biomass " << totB << endl;
  report << "log recruit deviations (styr_rec_dev-2011)   " << log_rec_dev(styr_rec_dev,2011) <<endl;
  report << "variance of log rec dev (select yrs)  "<<var_rec_dev<<endl;
  report << "autocorrelation " << R_autocorr <<endl;
  report<<" "<<endl;

  report << "Exploitation rate (1958-2007)" << endl;
 report << E << endl;
 report << "Fully-selected F (1958-2007)" << endl;
 report << fullF << endl;
 report << "Comm hand lines F" << endl;
```

```
 report << F_HL_out << endl;
 report << "Poundnet F" << endl;
 report << F_PN_out << endl;
 report << "Gillnet F" << endl;
 report << F_GN_out << endl;
 report << "Castnet F" << endl;
 report << F_CN_out << endl;
 report << "MRFSS F" << endl;
 report << F_MRFSS_out << endl;
 report << "Shrimp Bycatch F"<<endl;
 report << F_shrimp_out << endl;

 report << "selpar_L50_HL_keep" <<endl;
 report << selpar_L50_HL_keep <<endl;
 report << "selpar_slope_HL" <<endl;
 report << selpar_slope_HL <<endl;
 report << "selpar_L501_PN" <<endl;
 report << selpar_L501_PN <<endl;
 report << "selpar_slope1_PN" <<endl;
 report << selpar_slope1_PN <<endl;
 report << "selpar_L50_GN_keep" <<endl;
 report << selpar_L50_GN_keep <<endl;
 report << "selpar_slope_GN" <<endl;
 report << selpar_slope_GN <<endl;
 report << "selpar_L501_CN" <<endl;
 report << selpar_L501_CN <<endl;
 report << "selpar_slope1_CN" <<endl;
 report << selpar_slope1_CN <<endl;
 report << "selpar_L502_CN" <<endl;
 report << selpar_L502_CN <<endl;
 report << "selpar_slope2_CN" <<endl;
 report << selpar_slope2_CN <<endl;
 report << "selpar_L50_MRFSS_keep" <<endl;
 report << selpar_L501_MRFSS_keep <<endl;
 report << "selpar_slope_MRFSS_keep" <<endl;
 report << selpar_slope1_MRFSS_keep <<endl;
 report << "Hand lines selectivity - females" << endl;
 report << sel_HL_keep_F << endl;
 report << "Hand lines selectivity - males" << endl;
 report << sel_HL_keep_M << endl;
 report << "Poundnet selectivity - females" << endl;
 report << sel_PN_F << endl;
 report << "Poundnet selectivity - males" << endl;
 report << sel_PN_M << endl;
 report << "Gillnet selectivity - pre-1995- females" << endl;
 report << sel_GN_keep_F << endl;
 report << "Gillnet selectivity - pre-1995- males" << endl;
 report << sel_GN_keep_M << endl;
 report << "Gillnet selectivity - post-1995- females" << endl;
// report << sel_GN_keep_F2 << endl;
 report << "Gillnet selectivity - post-1995- males" << endl;
// report << sel_GN_keep_M2 << endl;
 report << "Castnet selectivity - females" << endl;
 report << sel_CN_F << endl;
 report << "Castnet selectivity - males" << endl;
 report << sel_CN_M << endl;
 report << "MRFSS selectivity - females" << endl;
 report << sel_MRFSS_keep_F << endl;
 report << "MRFSS selectivity - males" << endl;
 report << sel_MRFSS_keep_M << endl;

 report << "mean log F - HL" << endl;
 report << log_avg_F_HL << endl;
 report << "log F deviations - HL" << endl;
 report << log_F_dev_HL << endl;
 report << "mean log F - PN" << endl;
 report << log_avg_F_PN << endl;
 report << "log F deviations - PN" << endl;
 report << log_F_dev_PN << endl;
 report << "mean log F - GN" << endl;
 report << log_avg_F_GN << endl;
 report << "log F deviations - GN" << endl;
 report << log_F_dev_GN << endl;
 report << "mean log F - CN" << endl;
 report << log_avg_F_CN << endl;
 report << "log F deviations - CN" << endl;
 report << log_F_dev_CN << endl;
 report << "mean log F - MRFSS" << endl;
 report << log_avg_F_MRFSS << endl;
 report << "log F deviations - MRFSS" << endl;
 report << log_F_dev_MRFSS << endl;
 report << "mean log F - shrimp" << endl;
 report << log_avg_F_shrimp << endl;
 report << "log F deviations - shrimp" << endl;
 report << log_F_dev_shrimp << endl;

 report << "mean log F - MRFSS - Discards" << endl;
 report << log_avg_F_MRFSS_D << endl;
 report << "log F deviations - MRFSS - Discards" << endl;
 report << log_F_dev_MRFSS_D << endl;

 report << "Obs  FL_HL U "<<obs_FL_HL_cpue << endl;
 report << "pred FL_HL U "<<pred_FL_HL_cpue << endl;
 report << "Obs  MRFSS U "<<obs_MRFSS_cpue << endl;
 report << "pred MRFSS U "<<pred_MRFSS_cpue << endl;
 report << "Obs  SMAP_YOY U "<<obs_SMAP_YOY_cpue << endl;
 report << "pred SMAP_YOY U "<<pred_SMAP_YOY_cpue << endl;
 //report << "Obs  SMAP_1YR U "<<obs_SMAP_1YR_cpue << endl;
 //report << "pred SMAP_1YR U "<<pred_SMAP_1YR_cpue << endl;

 report << "Obs  HL landings (1000 lb) "<<obs_HL_L << endl;
 report << "pred HL landings (1000 lb) "<<pred_HL_L << endl;
 report << "Obs  PN landings (1000 lb) "<<obs_PN_L << endl;
 report << "pred PN landings (1000 lb) "<<pred_PN_L << endl;
```

```
report << "Obs  GN landings (1000 lb) "<<obs_GN_L << endl;
report << "pred GN landings (1000 lb) "<<pred_GN_L << endl;
report << "Obs  CN landings (1000 lb) "<<obs_CN_L << endl;
report << "pred CN landings (1000 lb) "<<pred_CN_L << endl;
report << "Obs  MRFSS landings (1000's) "<<obs_MRFSS_L << endl;
report << "pred MRFSS landings (1000's) "<<pred_MRFSS_L << endl;
report << "Obs shrimp bycatch (1000's) "<<obs_shrimp_B<<endl;
report << "pred shrimp bycatch (1000's) "<<pred_shrimp_B<<endl;

report << "Obs MRFSS discards (1000's) "<<obs_MRFSS_D<<endl;
report << "pred MRFSS discards (1000's) "<<pred_MRFSS_D<<endl;
#include "sm_make_Robject14.cxx"   // write the S-compatible report
```

Appendix B AD Model Builder code to implement the Beaufort Assessment Model

```
##-------------------------------------------------------------------
##  Data Input File
##  SEDAR28 Assessment: Spanish Mackerel
##
##-------------------------------------------------------------------
#starting and ending year of model
1950
2011
#Starting year to estimate recruitment deviation from S-R curve
1982
##Number of ages (11 classes is 0,1,...9,10+)
11
##vector of agebins, last is a plus group
0 1 2 3 4 5 6 7 8 9 10
#number length bins 11cm to 70+
60
#Vector of length bins (cm)(midpoint of bin) - last value = plus group
11.5 12.5 13.5 14.5 15.5 16.5 17.5 18.5 19.5 20.5 21.5 22.5 23.5 24.5 25.5 26.5
27.5 28.5 29.5 30.5 31.5 32.5 33.5 34.5 35.5 36.5 37.5 38.5 39.5 40.5 41.5 42.5
43.5 44.5 45.5 46.5 47.5 48.5 49.5 50.5 51.5 52.5 53.5 54.5 55.5 56.5 57.5 58.5
59.5 60.5 61.5 62.5 63.5 64.5 65.5 66.5 67.5 68.5 69.5 70.5
#discard mortality constant
0.20 #MRFSS
#initial proportion female (age 0)
0.5
#number of iterations in spr calculations (max F examined is (value-1)/1000
2001
#number of iterations in msy calculations (max F examined is (value-1)/10000
20001
#starting age for exploitation rate
1
#multiplicative bias correction (may set to 1.0 for none or negative to compute from rec variance)
-1.0
#starting value of constant cv of length at age ******was 0.1 for the RS assessment
0.1
#length-weight (whole wgt) coefficients a and b, W=aL^b, (W=Kg, L=mm)
2.2492E-08
2.8452
1.6486E-08
2.8934
7.48558E-09
3.0244
#Length at age vectors
276.87 360.88 421.88 466.18 498.35 521.71 538.67 550.98 559.93 566.42 571.14
129.093 300.398 398.249 454.142 486.069 504.306 514.723 520.673 524.072 526.014 527.123
120.809 298.113 414.61 491.154 541.446 574.491 596.203 610.469 619.842 626.001 630.047
#weight-weight conversion (gutted wgt=a*whole weight) NOT CURRENTLY USED
#time-variant vector of % maturity-at-age for females (ages 0-10)
0.0 0.939 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#time-variant vector of proportion female (ages 0-10+) NOT CURRENTLY USED
0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
################################Commercial Hand lines fishery##################################################
#starting/ending years for FL Trip ticket index
1986
2011
#CPUE and CV
0.60 0.73 0.89 0.80 0.87 0.65 0.80 0.93 0.66 0.93 0.82 0.80 0.77 0.95 0.98 0.98 0.94 1.01 1.31 1.29 1.37 1.20 1.23 1.64 1.41 1.42
0.04 0.04 0.05 0.05 0.04 0.03 0.04 0.04 0.04 0.04 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.03 0.02 0.02 0.02 0.02
#Starting and ending years of landings time series, respectively
1950
2011
#Observed landings (1000 lb whole weight) and CV's (LEW UPDATED THESE TO CONTAIN THE LANDINGS AND DISCARDS, HAS 2011 DATA)
371.468 531.019 63.340 184.932 123.365 367.263 662.788 294.198 434.657 170.918 157.720 123.235 128.832 79.227 66.813 148.942 180.867 135.111 155.181 106.455 113.555 140.288 108.692 158.439 172.637
379.202 933.068 349.264 82.924 75.330 93.868 89.164 128.545 58.603 56.484 30.988 79.915 109.930 66.577 41.181 115.250 150.550 51.449 102.047 58.984 211.970 142.066 129.433 151.327 190.787 316.299
356.040 440.526 392.125 594.248 846.684 710.637 780.521 872.909 982.224 1234.796 898.517
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05   0.05 0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05   0.05 0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05   0.05 0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05         0.05 0.05 0.05 0.05 0.05
0.05 0.05
#Number and vector of years of age compositions (comm hand line)
15
1989 1990 1992 1995 1996 1997 1998 1999 2000 2001 2002 2007 2008 2009 2011
#sample sizes of age comp data by year
22 79 81 25 35 19 31 120 147 242 61 177 185 104 72
#commercial handline age comp samples (year,age) (MADE CHANGE TO AGE ZERO COMPS FROM ROB)
0.0000 0.3182 0.3182 0.1364 0.1364 0.0909 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.1392 0.2658 0.3038 0.2658 0.0253 0.0000 0.0000 0.0000 0.0000
0.0123 0.3457 0.4198 0.1235 0.0494 0.0247 0.0247 0.0000 0.0000 0.0000 0.0000
0.5600 0.3600 0.0800 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0571 0.6857 0.0857 0.0571 0.0000 0.0571 0.0571 0.0000 0.0000 0.0000 0.0000
0.1053 0.1579 0.4211 0.2632 0.0526 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.1613 0.0000 0.1935 0.3226 0.2258 0.0645 0.0000 0.0323 0.0000 0.0000 0.0000
0.0000 0.0917 0.1333 0.1417 0.3333 0.1750 0.0750 0.0417 0.0000 0.0083 0.0000
0.0000 0.0884 0.4354 0.1837 0.1361 0.0816 0.0272 0.0408 0.0000 0.0000 0.0068
0.0620 0.3760 0.2603 0.1694 0.0455 0.0248 0.0372 0.0207 0.0041 0.0000 0.0000
0.0984 0.1475 0.0492 0.2459 0.2623 0.0984 0.0492 0.0000 0.0492 0.0000 0.0000
0.0067 0.1797 0.2977 0.1471 0.1886 0.0725 0.0698 0.0275 0.0019 0.0065 0.0021
0.0000 0.0070 0.1330 0.3108 0.2609 0.1756 0.0667 0.0449 0.0011 0.0000 0.0000
0.0000 0.0036 0.2498 0.1629 0.2122 0.1539 0.1316 0.0747 0.0113 0.0000 0.0000
0.0000 0.0000 0.0171 0.2269 0.3661 0.2413 0.1307 0.0179 0.0000 0.0000 0.0000
###################################Pound nets#################################
#Starting and ending years for landings time series
```

```
1950
2011
#Poundnet landings vector (1000 lb whole weight) and CV's (Added 2011 Landings)
26.244 38.731 55.422 51.274 195.579 60.083 89.846 57.149 15.316 25.938 24.543 136.871 20.789 83.100 33.149 91.791 114.436 24.068 74.328 89.048 108.094 26.491 23.313 51.892 25.682 62.257 77.479
28.955 2.405 0.727 5.859 5.580 24.065 16.436 23.470 47.699 205.482 485.574 412.781 528.488 524.866 489.504 406.283 338.182 333.778 201.242 299.923 211.194 116.667 274.158 163.927 199.847 121.629
90.843 71.453 47.298 43.079 50.315 192.927 364.502 144.896 88.109
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05
#number of years, year vector for poundnet age comps
15
1992 1995 1998 1999 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011
#sample sizes for pound net age comps
28 20 50 23 60 773 328 400 341 288 226 111 99 186    210
#poundnet age comps
0.6747 0.3224 0.0017 0.0012 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.9079 0.0921 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.2200 0.3400 0.2800 0.0800 0.0800 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.1304 0.8696 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0659 0.6593 0.1978 0.0549 0.0110 0.0000 0.0110 0.0000 0.0000 0.0000 0.0000
0.0039 0.6715 0.1143 0.1925 0.0158 0.0013 0.0000 0.0000 0.0000 0.0007 0.0000
0.0000 0.9834 0.0087 0.0000 0.0069 0.0011 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.5150 0.3150 0.0925 0.0100 0.0550 0.0100 0.0000 0.0000 0.0025 0.0000
0.0783 0.9031 0.0163 0.0022 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.6720 0.3028 0.0201 0.0000 0.0051 0.0000 0.0000 0.0000 0.0000 0.0000
0.0288 0.8777 0.0576 0.0216 0.0000 0.0144 0.0000 0.0000 0.0000 0.0000 0.0000
0.1042 0.6042 0.1667 0.0625 0.0625 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.4286 0.5000 0.0000 0.0714 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.6290 0.2688 0.0860 0.0000 0.0108 0.0000 0.0000 0.0054 0.0000 0.0000
0.0000 0.4667 0.2048 0.1762 0.0857 0.0429 0.0048 0.0143 0.0000 0.0048 0.0000
###################Commercial Gillnet CPUE and Landings#########################
#Commercial Gillnet Landings
#Starting and ending years of post-WW2 landings time series, respectively
1950
2011
#Observed landings (in 1000 lb whole weight - rounded) and assumed CVs  (gear="Other" added in as well)#(LEW UPDATED THESE TO CONTAIN THE LANDINGS AND DISCARDS)
3339.888 1619.950 3493.038 3541.995 2115.156 2981.054 4188.166 4141.253 7081.728 2330.744 2244.536 3158.995 2540.079 2184.873 2016.237 2865.868 2109.497 1749.621 4314.991 2379.997 3619.851 2566.621
3366.395 3116.169 2249.381 4835.640 9816.053 10984.081 5647.641 4850.916 9861.882 4258.124 3936.530 5995.916 2456.814 4321.597 4137.058 3733.177 3367.706 3302.445 2778.497 3971.404 2753.939 4547.820
3752.899 3272.712 2729.457 2726.108 2723.804 1912.282 1892.290 1740.271 1327.352 1096.767 720.336 1264.334 1656.917 1727.173 1085.148 1446.879 1354.809 1094.010
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05
#Number and vector of years of age compositions (comm gillnet)
23
1988 1989 1990 1991 1992 1993 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011
#sample sizes of age comps by year (minimum sample size of 45)
52 87 232 203 190 150 167 417 246 363 447 588 315 365 365 551 255 358 234 350 348 287 325
#age composition samples (year,age)
0.0769 0.2885 0.3654 0.1731 0.0769 0.0192 0.0000 0.0000 0.0000 0.0000 0.0000
0.2874 0.3218 0.1609 0.0805 0.0690 0.0690 0.0115 0.0000 0.0000 0.0000 0.0000
0.1595 0.2500 0.2284 0.1638 0.1379 0.0560 0.0043 0.0000 0.0000 0.0000 0.0000
0.0148 0.4532 0.2709 0.1232 0.0837 0.0443 0.0049 0.0049 0.0000 0.0000 0.0000
0.0035 0.4045 0.4604 0.1094 0.0132 0.0090 0.0000 0.0000 0.0000 0.0000 0.0000
0.0133 0.2467 0.2533 0.1533 0.1200 0.0467 0.0733 0.0600 0.0333 0.0000 0.0000
0.1497 0.2575 0.2814 0.1497 0.1078 0.0180 0.0180 0.0120 0.0060 0.0000 0.0000
0.0528 0.2398 0.3381 0.2782 0.0600 0.0216 0.0024 0.0048 0.0024 0.0000 0.0000
0.0347 0.2801 0.4729 0.1751 0.0241 0.0003 0.0044 0.0042 0.0000 0.0041 0.0000
0.2011 0.1460 0.3581 0.1736 0.0744 0.0303 0.0138 0.0028 0.0000 0.0000 0.0000
0.1432 0.3065 0.1723 0.1812 0.1275 0.0492 0.0179 0.0022 0.0000 0.0000 0.0000
0.0174 0.3802 0.3700 0.1060 0.0895 0.0215 0.0122 0.0025 0.0006 0.0000 0.0000
0.1720 0.4121 0.3093 0.0819 0.0227 0.0007 0.0007 0.0003 0.0003 0.0000 0.0000
0.1073 0.1449 0.2678 0.3124 0.1148 0.0439 0.0067 0.0000 0.0000 0.0022 0.0000
0.1362 0.5275 0.1146 0.0771 0.0854 0.0414 0.0121 0.0032 0.0000 0.0019 0.0006
0.0803 0.2789 0.3040 0.1840 0.0850 0.0450 0.0147 0.0077 0.0003 0.0000 0.0001
0.1097 0.6960 0.1330 0.0512 0.0074 0.0002 0.0023 0.0004 0.0000 0.0000 0.0000
0.0072 0.2459 0.3757 0.2366 0.1036 0.0218 0.0091 0.0000 0.0000 0.0000 0.0000
0.1796 0.4874 0.2276 0.0516 0.0263 0.0100 0.0040 0.0104 0.0028 0.0002 0.0000
0.1252 0.2944 0.3007 0.1338 0.1119 0.0246 0.0094 0.0000 0.0000 0.0000 0.0000
0.0315 0.3072 0.3222 0.1777 0.0923 0.0488 0.0204 0.0000 0.0000 0.0000 0.0000
0.2985 0.2110 0.1811 0.1281 0.1258 0.0534 0.0014 0.0008 0.0000 0.0000 0.0000
0.0124 0.2553 0.2017 0.2015 0.1904 0.1206 0.0158 0.0023 0.0000 0.0000 0.0000
#####################################Cast net#######################################
#Starting and ending years for landings time series
1995
2011
#Castnet landings vector (1000 lb whole weight) and CV's (THIS HAS 2011 LANDINGS)
15.590 67.163 214.259 69.025 67.099 366.080 909.541 971.702 1901.255 2253.709 1583.229 1529.961 1275.134 704.889 970.449 1807.528 1248.089
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05
#Number and vector of years of age compositions (castnet)
9
1997 2001 2005 2006 2007 2008 2009 2010   2011
#sample sizes of age comp data by year
34 110 147 211 50 199 331 138 94
#CN age comp samples (year,age)
0.0000 0.0588 0.5588 0.2059 0.1176 0.0588 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.0273 0.2727 0.3818 0.1727 0.1091 0.0273 0.0000 0.0091 0.0000 0.0000
0.0000 0.0105 0.2844 0.2827 0.2568 0.0944 0.0296 0.0088 0.0303 0.0000 0.0025
0.0000 0.0247 0.4764 0.2697 0.1487 0.0505 0.0176 0.0024 0.0049 0.0052 0.0000
0.0000 0.0000 0.2226 0.1768 0.2363 0.2628 0.0767 0.0249 0.0000 0.0000 0.0000
0.0000 0.0818 0.2423 0.2521 0.2147 0.1484 0.0382 0.0040 0.0174 0.0010 0.0000
0.0000 0.0078 0.2452 0.2479 0.2339 0.1820 0.0471 0.0257 0.0079 0.0000 0.0023
0.0000 0.0031 0.2439 0.3345 0.2249 0.1553 0.0327 0.0057 0.0000 0.0000 0.0000
0.0000 0.0000 0.2042 0.2395 0.2598 0.1952 0.0706 0.0306 0.0000 0.0000 0.0000
################################### Recreational  ##################################
#Recreational MRFSS CPUE Index
```

```
#Starting and ending years of CPUE index
1982
2011
#Observed CPUE and assumed CVs
0.84 0.50 0.58 0.67 1.42 0.84 0.89 0.97 0.92 0.88 0.83 0.62 0.99 0.76 1.03 1.30 1.03 1.25 1.14 1.17 1.25 1.22 1.04 1.13 1.03 1.03 1.46 1.06 1.05 1.07
0.20 0.15 0.15 0.12 0.06 0.07 0.07 0.06 0.06 0.06 0.07 0.06 0.07 0.07 0.07 0.06 0.06 0.06 0.06 0.06 0.06 0.06 0.06 0.07 0.06 0.06 0.06 0.05 0.06 0.16
#Recreational Charter+Private boat landings
#Starting and ending years for landings time series
1955
2011
#MRFSS + headboat landings vector (1000's)
252.837 266.763 280.69 294.616 308.543 322.469 343.288 364.106 384.925 405.744 426.562 444.157 461.752 479.348 496.943 514.538 559.473 604.408 649.344 694.279 739.214 731.721 724.228 716.735 709.243
701.75 867.492 965.918 130.237 938.061 495.354 937.429 1198.109 1884.597 1232.315 1391.631 1638.608 1346.942 980.356 1252.47 753.008 969.077 1155.037 690.496 1116.645 1437.33 1307.163 1439.449
1243.097 800.943 962.09 663.235 1087.412 1415.57 1170.894 1103.948 879.23
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05
#Starting and ending years of discards time series, respectively
1981
2011
#Observed discards (1000s) and assumed CVs
61.990 6.613 5.443 25.458 53.886 322.181 58.444 67.425 237.817 163.088 384.225 339.112 249.243 769.157 341.450 403.594 414.422 266.657 500.496 814.396 509.191 790.900 872.303 462.853 620.683 283.597
616.008 931.700 572.612 639.027 425.852
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05
#Starting and ending year of mrfss age composition data
1988
2011
24
1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011
#sample sizes of mrfss age comp data by year
115 34 271 192 198 104 171 70 78 316 219 89 130 49 204 235 239 204 255 183 182 63 296 280
#mrfss age comps (year,age)
0.0138 0.1209 0.4754 0.3695 0.0107 0.0056 0.0020 0.0011 0.0011 0.0000 0.0000
0.0000 0.1869 0.0406 0.1971 0.3268 0.1443 0.0952 0.0091 0.0000 0.0000 0.0000
0.0305 0.5137 0.2631 0.1279 0.0329 0.0175 0.0139 0.0005 0.0000 0.0000 0.0000
0.0315 0.5431 0.2373 0.1140 0.0167 0.0290 0.0209 0.0073 0.0000 0.0000 0.0000
0.0029 0.5329 0.2766 0.0653 0.0907 0.0109 0.0085 0.0056 0.0066 0.0000 0.0000
0.0016 0.5374 0.2331 0.0444 0.1221 0.0380 0.0078 0.0133 0.0019 0.0000 0.0004
0.0330 0.6050 0.2269 0.1116 0.0234 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.1070 0.7513 0.1194 0.0000 0.0222 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.5935 0.3122 0.0060 0.0494 0.0389 0.0000 0.0000 0.0000 0.0000 0.0000
0.0325 0.6640 0.2478 0.0401 0.0037 0.0086 0.0033 0.0000 0.0000 0.0000 0.0000
0.0870 0.4704 0.3113 0.0710 0.0361 0.0038 0.0023 0.0153 0.0027 0.0000 0.0001
0.0000 0.8823 0.0345 0.0478 0.0194 0.0097 0.0052 0.0011 0.0000 0.0000 0.0000
0.0000 0.4520 0.2595 0.0415 0.1001 0.0609 0.0401 0.0306 0.0066 0.0002 0.0085
0.0474 0.2464 0.2154 0.0859 0.1930 0.1237 0.0743 0.0002 0.0137 0.0000 0.0000
0.0734 0.4353 0.2212 0.1494 0.0808 0.0168 0.0012 0.0085 0.0071 0.0044 0.0018
0.0313 0.8660 0.0855 0.0122 0.0031 0.0017 0.0000 0.0002 0.0000 0.0000 0.0000
0.0342 0.7176 0.1158 0.0925 0.0305 0.0044 0.0050 0.0000 0.0000 0.0000 0.0000
0.0165 0.9258 0.0184 0.0175 0.0081 0.0086 0.0024 0.0000 0.0027 0.0000 0.0000
0.1262 0.7967 0.0363 0.0202 0.0157 0.0013 0.0033 0.0002 0.0000 0.0000 0.0000
0.1734 0.7573 0.0494 0.0080 0.0045 0.0055 0.0000 0.0019 0.0000 0.0000 0.0000
0.0135 0.8306 0.0905 0.0549 0.0106 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0000 0.7776 0.1046 0.1004 0.0174 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
0.0877 0.4732 0.2954 0.1027 0.0333 0.0034 0.0042 0.0000 0.0000 0.0000 0.0000
0.2165 0.6110 0.0936 0.0592 0.0197 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
################## Shrimp Bycatch ##########################################################################
#starting, ending years for shrimp bycatch (LEW ADDED 2011 SHRIMP BYCATCH)
1950
2011
#bycatch estimates (in thousands)
11.239929 22.47985799 33.71978699 44.95971598 56.19964498 67.43957398 78.67950297 89.91943197 101.159361 112.39929 123.639219 134.879148 146.119077 157.3590059 168.5989349 179.8388639 191.0787929
202.3187219 213.5586509 224.7985799 236.0385089 247.2784379 258.5183669 269.7582959 280.9982249 292.2381539 303.4780829 314.7180119 325.9579409 255.1738355 385.4373569 296.1014008 445.1744086
433.1467925 272.9051115 265.0871433 290.9917255 245.2705746 292.5448055 345.6559124 268.6551857 333.3993057 252.8378971 266.4794525 297.7101317 301.2597395 245.0850855 280.2564363 252.1524977
293.6549916 265.6777038 210.9083238 236.3407972 177.9929502 178.256264 132.9337981 127.8276852 121.8886564 112.4796616 104.3848914 122.9739946 113.8752721
#70440.4 140880.7 211321.1 281761.5 352201.8 422642.2 493082.6 563522.9 633963.3 704403.7 774844.0 845284.4 915724.8 986165.1 1056605.5 1127045.9 1197486.2 1267926.6 1338367.0 1408807.3 1479247.7
1549688.0 1620128.4 1690568.8 1761009.1 1831449.5 1901889.9 1972330.2 1949137.9 1647029.2 2655308.2 2023911.1 3072163.8 2986210.1 1958459.5 1857566.8 1991657.0 1654446.7 2012567.9 2397712.5 1778494.2
2228659.1 1496039.3 1664717.4 1919211.3 1970538.4 1563803.7 1801486.3 1548191.9 1851822.5 1668779.7 1324712.7 1564404.4 1122626.6 1147100.8 776134.1 787964.9 794135.5 758118.5 671553.4 758516.7
685442.4
#CV's for bycatch
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05 0.05 0.05 0.05  0.05 0.05 0.05 0.05 0.05
0.05 0.05
#starting, ending years for computing geometric mean of shrimp bycatch (for fitting)
#1978
#2011
################################### SEAMAP  ################################
#SEAMAP YOY Index
#Starting and ending years of CPUE index
1989
2011
#Observed CPUE and assumed CVs
0.73 1.63 1.72 1.19 0.93 0.82 1.07 1.20 0.26 1.46 0.81 1.16 0.88 1.00 0.40 0.74 0.77 1.33 1.25 1.52 1.10 0.62 0.41
0.56 0.46 0.48 0.52 0.49 0.47 0.5 0.47 0.59 0.52 0.51 0.5 0.47 0.48 0.55 0.53 0.53 0.53 0.51 0.48 0.49 0.54 0.56
#1 Yr Old index
#Starting and ending years of CPUE index
#1990
#2011
#Observed CPUE and assumed CVs
#0.44 0.89 3.19 0.59 1.6 0.88 1.6 0.57 0.28 1.79 2.21 0.73 0.5 0.68 0.71 0.57 0.99 0.99 0.73 1.01 0.41 0.64
#0.28 0.21 0.15 0.35 0.19 0.31 0.24 0.31 0.31 0.29 0.17 0.19 0.22 0.21 0.24 0.38 0.28 0.17 0.22 0.18 0.18 0.16
################Likelihood Component Weighting#############################################################################
#Weights in objective fcn
1.0  #landings
```

```
1.0  #discards
0.05449 #age comps
0.03223 #
0.06323 #
0.10371 #
0.01064 #
0.33455 #FL hand lines cpue index
0.40447 #MRFSS index
1.21863 #SEAMAP YOY Index
#1.0    #SEAMAP 1-Yr-Old Index
1.0     #S-R residuals
0.0 #constrain first several years of recruitment variability
1.0     #additional constraint on variability of recruitment in last three yrs
0.0 #constrain variability of F in last three years
0.0 #constraint B1/B0
1.0   #penalty if F exceeds 5.0
0.0 #penalty on deviation in CV at age
0.0     #penalty on first difference in CV at age
################Parameter values and initial guesses##########################################################################
#steepness (fixed or initial guess)
0.75
#SE of steepness
0.15
#natural mortality at age
0.792965273 0.516472882 0.425924009 0.38393558 0.361489288 0.348588231 0.340865577 0.336130336 0.333184104 0.331334307 0.330166307
#age-independent natural mortality (used only to compute MSST=(1-M)SSBmsy)
0.35
# SD of recruitment in log space
0.6
# SE of SD recruitment
0.15
#log catchabilities (initial guesses)
-11.0
-11.0
-11.0
#historical full F
0.2
#FL hand lines
#log mean F - landings
-4
#F deviations (HL-L)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0   0 0 0 0 0  0 0
#Pound nets
#log mean F - PN
-4
#F deviations (PN)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0
#Gillnets
#log mean F - GN
-2
#F deviations (GN)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0 0 0
#Castnets
#log mean F - CN
-4
#F deviations (CN)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0
#log mean F - MRFSS
-2.0
#F deviations (MRFSS)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0
#log mean F - MRFSS - discards
-4.0
#F deviations (MRFSS discards)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0
#log mean F - shrimp
-1.0
#F deviations (shrimp)
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0  0 0 0 0 0
0 0 0 0 0  0 0 0 0 0  0 0 0 0 0   0 0 0 0 0  0 0
#log_R0 - log virgin recruitment
18
# R autocorrelation
0.0
#Selectivity parameters.
2.0 #selpar_L50_HL_keep b1
2.0 #selpar_slope_HL    b3
0.70 #selpar_L50_PN_keep b1
2.0 #selpar_slope_PN    b3
1.0 #selpar_L502_PN_keep b1
2.0 #selpar_slope2_PN    b3
2.0 #selpar_L50_GN pre netban b1
2.0 #selpar_slope_GN pre netban   b3
2.0 #selpar_L501_CN  b1
2.0 #selpar_slope1_CN    b3
2.0 #selpar_L502_CN
2.0 #selpar_slope2_CN
1.0 #selpar_L501_MRFSS_keep b1
3.0 #selpar_slope1_MRFSS    b3
1.0 #selpar_L501_MRFSS_keep b1
3.0 #selpar_slope1_MRFSS    b3
0.05 0.5  1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 #historical selectivity - females
1.00 0.34 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 #selectivity vector for discards - MRFSS - females  b1
1.00 0.41 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 #selectivity vector for discards - MRFSS - males
```

```
1.00 0.2  0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 #selectivity vector for shrimp bycatch
0.2 #init_number set_L50_diff: difference between males and females
#Aging Error Matrix
1 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 1
# historic recreational landings multiplier
1.0
#min SS for the age comps for the 5 fisheries
10.0
10.0
10.0
10.0
10.0
999 #end of data file flag
```

Appendix C ASPIC Output: Results of production model run for Spanish mackerel.

```
SAFMC Spanish mackerel (2011) Landings and Indices

ASPIC -- A Surplus-Production Model Including Covariates (Ver. 5.31)
                                                                                                  BOT program mode
Author:     Michael H. Prager; NOAA Center for Coastal Fisheries and Habitat Research          LOGISTIC model mode
            101 Pivers Island Road; Beaufort, North Carolina  28516  USA                          YLD conditioning
            Mike.Prager@noaa.gov                                                                 SSE optimization

Reference:  Prager, M. H. 1994. A suite of extensions to a nonequilibrium          ASPIC User's Manual is available
            surplus-production model.  Fishery Bulletin 92: 374-389.                       gratis from the author.

CONTROL PARAMETERS (FROM INPUT FILE)                    Input file: w:\sedar\sedar 28\sm\aw\aspic\sm2011_011boot.inp
-------------------------------------------------------------------------------------------------------------------------
Operation of ASPIC:  Fit logistic (Schaefer) model by direct optimization with bootstrap.
Number of years analyzed:                      62          Number of bootstrap trials:                        1000
Number of data series:                          2          Bounds on MSY (min, max):       1.000E+03     2.000E+06
Objective function:                Least squares          Bounds on K (min, max):         1.000E+04     1.000E+09
Relative conv. criterion (simplex):     1.000E-08          Monte Carlo search mode, trials:         0        100000
Relative conv. criterion (restart):     3.000E-08          Random number seed:                           82184571
Relative conv. criterion (effort):      1.000E-04          Identical convergences required in fitting:          8
Maximum F allowed in fitting:               8.000


PROGRAM STATUS INFORMATION (NON-BOOTSTRAPPED ANALYSIS)                                                 error code   0
-------------------------------------------------------------------------------------------------------------------------
Normal convergence


CORRELATION AMONG INPUT SERIES EXPRESSED AS CPUE (NUMBER OF PAIRWISE OBSERVATIONS BELOW)
-------------------------------------------------------------------------------------------------------------------------
                                    |
 1  MRFSS Index (1982-2011), Total L...|   1.000
                                    |      30
                                    |
 2  Commercial                      |   0.136   1.000
                                    |      26      26
                                    ------------------------------------------------
                                           1       2


GOODNESS-OF-FIT AND WEIGHTING (NON-BOOTSTRAPPED ANALYSIS)
-------------------------------------------------------------------------------------------------------------------------
                                                  Weighted           Weighted      Current    Inv. var.    R-squared
Loss component number and title                        SSE     N          MSE       weight       weight      in CPUE

Loss(-1)  SSE in yield                           0.000E+00
Loss(0)   Penalty for B1 > K                     0.000E+00     1          N/A    1.000E+00          N/A
Loss(1)   MRFSS Index (1982-2011), Total Ldgs 100  1.016E+00    30    3.629E-02    1.000E+00    8.551E-01        0.382
Loss(2)   Commercial                             6.380E-01    26    2.658E-02    1.000E+00    1.167E+00        0.618
.........................................................................................
TOTAL OBJECTIVE FUNCTION, MSE, RMSE:          1.65402306E+00          3.243E-02    1.801E-01
Estimated contrast index (ideal = 1.0):             0.4958          C* = (Bmax-Bmin)/K
Estimated nearness index (ideal = 1.0):             1.0000          N* = 1 - |min(B-Bmsy)|/K
SAFMC Spanish mackerel (2011) Landings and Indices                                                     Page 2


MODEL PARAMETER ESTIMATES (NON-BOOTSTRAPPED)
-------------------------------------------------------------------------------------------------------------------------
Parameter                                          Estimate     User/pgm guess    2nd guess    Estimated   User guess

B1/K      Starting relative biomass (in 1950)      3.787E-01          5.000E-01    6.022E-01           1            1
MSY       Maximum sustainable yield                6.484E+03          6.200E+03    4.546E+03           1            1
K         Maximum population size                  4.478E+04          1.200E+05    2.728E+04           1            1
phi       Shape of production curve (Bmsy/K)       0.5000             0.5000            ----           0            1

--------- Catchability Coefficients by Data Series ---------------
q(1)      MRFSS Index (1982-2011), Total Ldgs 100  4.244E-05          5.000E-06    4.750E-04           1            1
q(2)      Commercial                               4.042E-05          5.000E-06    4.750E-04           1            1


MANAGEMENT and DERIVED PARAMETER ESTIMATES (NON-BOOTSTRAPPED)
-------------------------------------------------------------------------------------------------------------------------
Parameter                                          Estimate               Logistic formula          General formula
```

MSY	Maximum sustainable yield	6.484E+03	----	----
Bmsy	Stock biomass giving MSY	2.239E+04	K/2	K*n**(1/(1-n))
Fmsy	Fishing mortality rate at MSY	2.896E-01	MSY/Bmsy	MSY/Bmsy
n	Exponent in production function	2.0000	----	----
g	Fletcher's gamma	4.000E+00	----	[n**(n/(n-1))]/[n-1]
B./Bmsy	Ratio: B(2012)/Bmsy	1.287E+00	----	----
F./Fmsy	Ratio: F(2011)/Fmsy	6.404E-01	----	----
Fmsy/F.	Ratio: Fmsy/F(2011)	1.561E+00	----	----
Y.(Fmsy)	Approx. yield available at Fmsy in 2012	8.344E+03	MSY*B./Bmsy	MSY*B./Bmsy
	...as proportion of MSY	1.287E+00	----	----
Ye.	Equilibrium yield available in 2012	5.950E+03	4*MSY*(B/K-(B/K)**2)	g*MSY*(B/K-(B/K)**n)
	...as proportion of MSY	9.177E-01	----	----
--------- Fishing effort rate at MSY in units of each CE or CC series ---------				
fmsy(1)	MRFSS Index (1982-2011), Total Ldgs 100	6.823E+03	Fmsy/q(1)	Fmsy/q(1)

ESTIMATED POPULATION TRAJECTORY (NON-BOOTSTRAPPED)

Obs	Year or ID	Estimated total F mort	Estimated starting biomass	Estimated average biomass	Observed total yield	Model total yield	Estimated surplus production	Ratio of F mort to Fmsy	Ratio of biomass to Bmsy
1	1950	0.205	1.696E+04	1.824E+04	3.739E+03	3.739E+03	6.254E+03	7.081E-01	7.574E-01
2	1951	0.101	1.947E+04	2.163E+04	2.193E+03	2.193E+03	6.457E+03	3.501E-01	8.697E-01
3	1952	0.144	2.374E+04	2.517E+04	3.617E+03	3.617E+03	6.376E+03	4.963E-01	1.060E+00
4	1953	0.137	2.650E+04	2.771E+04	3.785E+03	3.785E+03	6.112E+03	4.716E-01	1.183E+00
5	1954	0.080	2.882E+04	3.049E+04	2.442E+03	2.442E+03	5.625E+03	2.766E-01	1.287E+00
6	1955	0.118	3.201E+04	3.268E+04	3.840E+03	3.840E+03	5.113E+03	4.058E-01	1.429E+00
7	1956	0.163	3.328E+04	3.307E+04	5.398E+03	5.398E+03	5.009E+03	5.637E-01	1.486E+00
8	1957	0.151	3.289E+04	3.293E+04	4.974E+03	4.974E+03	5.048E+03	5.217E-01	1.469E+00
9	1958	0.255	3.296E+04	3.153E+04	8.038E+03	8.038E+03	5.396E+03	8.803E-01	1.472E+00
10	1959	0.097	3.032E+04	3.155E+04	3.059E+03	3.059E+03	5.393E+03	3.348E-01	1.354E+00
11	1960	0.089	3.265E+04	3.364E+04	2.983E+03	2.983E+03	4.843E+03	3.062E-01	1.458E+00
12	1961	0.115	3.451E+04	3.478E+04	4.012E+03	4.012E+03	4.500E+03	3.984E-01	1.541E+00
13	1962	0.093	3.500E+04	3.551E+04	3.319E+03	3.319E+03	4.259E+03	3.228E-01	1.563E+00
14	1963	0.083	3.594E+04	3.644E+04	3.013E+03	3.013E+03	3.932E+03	2.856E-01	1.605E+00
15	1964	0.076	3.686E+04	3.729E+04	2.818E+03	2.818E+03	3.613E+03	2.610E-01	1.646E+00
16	1965	0.103	3.766E+04	3.749E+04	3.845E+03	3.845E+03	3.536E+03	3.542E-01	1.682E+00
17	1966	0.085	3.735E+04	3.753E+04	3.174E+03	3.174E+03	3.519E+03	2.921E-01	1.668E+00
18	1967	0.071	3.769E+04	3.802E+04	2.709E+03	2.709E+03	3.324E+03	2.461E-01	1.683E+00
19	1968	0.144	3.831E+04	3.734E+04	5.376E+03	5.376E+03	3.592E+03	4.972E-01	1.711E+00
20	1969	0.094	3.652E+04	3.673E+04	3.438E+03	3.438E+03	3.825E+03	3.232E-01	1.631E+00
21	1970	0.130	3.691E+04	3.647E+04	4.735E+03	4.735E+03	3.921E+03	4.484E-01	1.648E+00
22	1971	0.102	3.609E+04	3.625E+04	3.704E+03	3.704E+03	3.999E+03	3.528E-01	1.612E+00
23	1972	0.126	3.639E+04	3.612E+04	4.545E+03	4.545E+03	4.046E+03	4.346E-01	1.625E+00
24	1973	0.124	3.589E+04	3.574E+04	4.450E+03	4.450E+03	4.178E+03	4.299E-01	1.603E+00
25	1974	0.102	3.562E+04	3.588E+04	3.648E+03	3.648E+03	4.131E+03	3.511E-01	1.591E+00
26	1975	0.188	3.610E+04	3.495E+04	6.554E+03	6.554E+03	4.438E+03	6.475E-01	1.612E+00
27	1976	0.398	3.399E+04	3.042E+04	1.209E+04	1.209E+04	5.606E+03	1.373E+00	1.518E+00
28	1977	0.524	2.750E+04	2.410E+04	1.262E+04	1.262E+04	6.405E+03	1.808E+00	1.228E+00
29	1978	0.332	2.129E+04	2.102E+04	6.977E+03	6.977E+03	6.459E+03	1.146E+00	9.507E-01
30	1979	0.294	2.077E+04	2.093E+04	6.148E+03	6.148E+03	6.456E+03	1.014E+00	9.276E-01
31	1980	0.608	2.108E+04	1.840E+04	1.119E+04	1.119E+04	6.252E+03	2.100E+00	9.413E-01
32	1981	0.373	1.614E+04	1.612E+04	6.012E+03	6.012E+03	5.976E+03	1.288E+00	7.208E-01
33	1982	0.336	1.610E+04	1.637E+04	5.496E+03	5.496E+03	6.015E+03	1.159E+00	7.192E-01
34	1983	0.388	1.662E+04	1.644E+04	6.378E+03	6.378E+03	6.026E+03	1.340E+00	7.424E-01
35	1984	0.229	1.627E+04	1.737E+04	3.982E+03	3.982E+03	6.153E+03	7.916E-01	7.266E-01
36	1985	0.275	1.844E+04	1.901E+04	5.227E+03	5.227E+03	6.335E+03	9.494E-01	8.236E-01
37	1986	0.313	1.955E+04	1.967E+04	6.151E+03	6.151E+03	6.388E+03	1.080E+00	8.731E-01
38	1987	0.319	1.979E+04	1.983E+04	6.325E+03	6.325E+03	6.399E+03	1.102E+00	8.837E-01
39	1988	0.377	1.986E+04	1.937E+04	7.294E+03	7.294E+03	6.365E+03	1.300E+00	8.870E-01
40	1989	0.302	1.893E+04	1.922E+04	5.804E+03	5.804E+03	6.353E+03	1.043E+00	8.455E-01
41	1990	0.280	1.948E+04	1.991E+04	5.568E+03	5.568E+03	6.404E+03	9.656E-01	8.700E-01
42	1991	0.396	2.032E+04	1.960E+04	7.750E+03	7.750E+03	6.381E+03	1.366E+00	9.074E-01
43	1992	0.295	1.895E+04	1.929E+04	5.693E+03	5.693E+03	6.359E+03	1.019E+00	8.462E-01
44	1993	0.353	1.961E+04	1.937E+04	6.829E+03	6.829E+03	6.366E+03	1.217E+00	8.760E-01
45	1994	0.340	1.915E+04	1.908E+04	6.481E+03	6.481E+03	6.342E+03	1.173E+00	8.553E-01
46	1995	0.254	1.901E+04	1.972E+04	5.010E+03	5.010E+03	6.390E+03	8.773E-01	8.491E-01
47	1996	0.235	2.039E+04	2.116E+04	4.981E+03	4.981E+03	6.462E+03	8.131E-01	9.107E-01
48	1997	0.244	2.187E+04	2.240E+04	5.469E+03	5.469E+03	6.483E+03	8.431E-01	9.768E-01
49	1998	0.187	2.289E+04	2.391E+04	4.477E+03	4.477E+03	6.450E+03	6.466E-01	1.022E+00

50	1999	0.169	2.486E+04	2.588E+04	4.365E+03	4.365E+03	6.322E+03	5.824E-01	1.110E+00
51	2000	0.201	2.682E+04	2.719E+04	5.476E+03	5.476E+03	6.186E+03	6.954E-01	1.198E+00
52	2001	0.204	2.753E+04	2.777E+04	5.656E+03	5.656E+03	6.110E+03	7.034E-01	1.229E+00
53	2002	0.205	2.798E+04	2.814E+04	5.764E+03	5.764E+03	6.057E+03	7.074E-01	1.250E+00
54	2003	0.212	2.827E+04	2.829E+04	5.998E+03	5.998E+03	6.034E+03	7.320E-01	1.263E+00
55	2004	0.193	2.831E+04	2.856E+04	5.512E+03	5.512E+03	5.991E+03	6.664E-01	1.264E+00
56	2005	0.199	2.879E+04	2.889E+04	5.747E+03	5.747E+03	5.938E+03	6.869E-01	1.286E+00
57	2006	0.184	2.898E+04	2.924E+04	5.383E+03	5.383E+03	5.877E+03	6.357E-01	1.294E+00
58	2007	0.212	2.947E+04	2.929E+04	6.207E+03	6.207E+03	5.868E+03	7.318E-01	1.316E+00
59	2008	0.202	2.913E+04	2.913E+04	5.898E+03	5.898E+03	5.896E+03	6.991E-01	1.301E+00
60	2009	0.207	2.913E+04	2.907E+04	6.027E+03	6.027E+03	5.907E+03	7.160E-01	1.301E+00
61	2010	0.243	2.901E+04	2.852E+04	6.918E+03	6.918E+03	5.997E+03	8.376E-01	1.296E+00
62	2011	0.185	2.809E+04	2.847E+04	5.281E+03	5.281E+03	6.005E+03	6.404E-01	1.255E+00
63	2012		2.882E+04						1.287E+00

RESULTS FOR DATA SERIES # 1 (NON-BOOTSTRAPPED) MRFSS Index (1982-2011), Total Ldgs 1000

Data type CC: CPUE-catch series Series weight: 1.000

Obs	Year	Observed CPUE	Estimated CPUE	Estim F	Observed yield	Model yield	Resid in log scale	Statist weight
1	1950	*	7.739E-01	0.2050	3.739E+03	3.739E+03	0.00000	1.000E+00
2	1951	*	9.181E-01	0.1014	2.193E+03	2.193E+03	0.00000	1.000E+00
3	1952	*	1.068E+00	0.1437	3.617E+03	3.617E+03	0.00000	1.000E+00
4	1953	*	1.176E+00	0.1366	3.785E+03	3.785E+03	0.00000	1.000E+00
5	1954	*	1.294E+00	0.0801	2.442E+03	2.442E+03	0.00000	1.000E+00
6	1955	*	1.387E+00	0.1175	3.840E+03	3.840E+03	0.00000	1.000E+00
7	1956	*	1.403E+00	0.1632	5.398E+03	5.398E+03	0.00000	1.000E+00
8	1957	*	1.397E+00	0.1511	4.974E+03	4.974E+03	0.00000	1.000E+00
9	1958	*	1.338E+00	0.2549	8.038E+03	8.038E+03	0.00000	1.000E+00
10	1959	*	1.339E+00	0.0970	3.059E+03	3.059E+03	0.00000	1.000E+00
11	1960	*	1.428E+00	0.0887	2.983E+03	2.983E+03	0.00000	1.000E+00
12	1961	*	1.476E+00	0.1154	4.012E+03	4.012E+03	0.00000	1.000E+00
13	1962	*	1.507E+00	0.0935	3.319E+03	3.319E+03	0.00000	1.000E+00
14	1963	*	1.546E+00	0.0827	3.013E+03	3.013E+03	0.00000	1.000E+00
15	1964	*	1.582E+00	0.0756	2.818E+03	2.818E+03	0.00000	1.000E+00
16	1965	*	1.591E+00	0.1026	3.845E+03	3.845E+03	0.00000	1.000E+00
17	1966	*	1.593E+00	0.0846	3.174E+03	3.174E+03	0.00000	1.000E+00
18	1967	*	1.614E+00	0.0713	2.709E+03	2.709E+03	0.00000	1.000E+00
19	1968	*	1.584E+00	0.1440	5.376E+03	5.376E+03	0.00000	1.000E+00
20	1969	*	1.559E+00	0.0936	3.438E+03	3.438E+03	0.00000	1.000E+00
21	1970	*	1.548E+00	0.1298	4.735E+03	4.735E+03	0.00000	1.000E+00
22	1971	*	1.539E+00	0.1022	3.704E+03	3.704E+03	0.00000	1.000E+00
23	1972	*	1.533E+00	0.1258	4.545E+03	4.545E+03	0.00000	1.000E+00
24	1973	*	1.517E+00	0.1245	4.450E+03	4.450E+03	0.00000	1.000E+00
25	1974	*	1.523E+00	0.1017	3.648E+03	3.648E+03	0.00000	1.000E+00
26	1975	*	1.483E+00	0.1875	6.554E+03	6.554E+03	0.00000	1.000E+00
27	1976	*	1.291E+00	0.3976	1.209E+04	1.209E+04	0.00000	1.000E+00
28	1977	*	1.023E+00	0.5235	1.262E+04	1.262E+04	0.00000	1.000E+00
29	1978	*	8.919E-01	0.3320	6.977E+03	6.977E+03	0.00000	1.000E+00
30	1979	*	8.882E-01	0.2938	6.148E+03	6.148E+03	0.00000	1.000E+00
31	1980	*	7.809E-01	0.6081	1.119E+04	1.119E+04	0.00000	1.000E+00
32	1981	*	6.842E-01	0.3729	6.012E+03	6.012E+03	0.00000	1.000E+00
33	1982	8.410E-01	6.948E-01	0.3357	5.496E+03	5.496E+03	-0.19101	1.000E+00
34	1983	5.006E-01	6.977E-01	0.3880	6.378E+03	6.378E+03	0.33193	1.000E+00
35	1984	5.794E-01	7.373E-01	0.2292	3.982E+03	3.982E+03	0.24100	1.000E+00
36	1985	6.711E-01	8.068E-01	0.2749	5.227E+03	5.227E+03	0.18422	1.000E+00
37	1986	1.422E+00	8.349E-01	0.3127	6.151E+03	6.151E+03	-0.53274	1.000E+00
38	1987	8.428E-01	8.413E-01	0.3191	6.325E+03	6.325E+03	-0.00172	1.000E+00
39	1988	8.884E-01	8.222E-01	0.3765	7.294E+03	7.294E+03	-0.07749	1.000E+00
40	1989	9.728E-01	8.155E-01	0.3020	5.804E+03	5.804E+03	-0.17637	1.000E+00
41	1990	9.220E-01	8.451E-01	0.2796	5.568E+03	5.568E+03	-0.08707	1.000E+00
42	1991	8.846E-01	8.316E-01	0.3955	7.750E+03	7.750E+03	-0.06178	1.000E+00
43	1992	8.258E-01	8.188E-01	0.2951	5.693E+03	5.693E+03	-0.00857	1.000E+00
44	1993	6.234E-01	8.221E-01	0.3525	6.829E+03	6.829E+03	0.27668	1.000E+00
45	1994	9.885E-01	8.097E-01	0.3397	6.481E+03	6.481E+03	-0.19958	1.000E+00
46	1995	7.603E-01	8.370E-01	0.2540	5.010E+03	5.010E+03	0.09611	1.000E+00
47	1996	1.029E+00	8.979E-01	0.2354	4.981E+03	4.981E+03	-0.13594	1.000E+00
48	1997	1.297E+00	9.506E-01	0.2442	5.469E+03	5.469E+03	-0.31071	1.000E+00
49	1998	1.033E+00	1.015E+00	0.1872	4.477E+03	4.477E+03	-0.01750	1.000E+00
50	1999	1.247E+00	1.098E+00	0.1687	4.365E+03	4.365E+03	-0.12721	1.000E+00
51	2000	1.143E+00	1.154E+00	0.2014	5.476E+03	5.476E+03	0.00942	1.000E+00
52	2001	1.174E+00	1.178E+00	0.2037	5.656E+03	5.656E+03	0.00345	1.000E+00
53	2002	1.252E+00	1.194E+00	0.2049	5.764E+03	5.764E+03	-0.04755	1.000E+00
54	2003	1.221E+00	1.201E+00	0.2120	5.998E+03	5.998E+03	-0.01651	1.000E+00
55	2004	1.041E+00	1.212E+00	0.1930	5.512E+03	5.512E+03	0.15224	1.000E+00

56	2005	1.133E+00	1.226E+00	0.1989	5.747E+03	5.747E+03	0.07858	1.000E+00
57	2006	1.030E+00	1.241E+00	0.1841	5.383E+03	5.383E+03	0.18673	1.000E+00
58	2007	1.031E+00	1.243E+00	0.2119	6.207E+03	6.207E+03	0.18673	1.000E+00
59	2008	1.464E+00	1.236E+00	0.2024	5.898E+03	5.898E+03	-0.16885	1.000E+00
60	2009	1.058E+00	1.234E+00	0.2073	6.027E+03	6.027E+03	0.15407	1.000E+00
61	2010	1.054E+00	1.210E+00	0.2425	6.918E+03	6.918E+03	0.13885	1.000E+00
62	2011	1.071E+00	1.208E+00	0.1855	5.281E+03	5.281E+03	0.12072	1.000E+00

* Asterisk indicates missing value(s).

RESULTS FOR DATA SERIES # 2 (NON-BOOTSTRAPPED) Commercial

Data type I1: Abundance index (annual average) Series weight: 1.000

Obs	Year	Observed effort	Estimated effort	Estim F	Observed index	Model index	Resid in log index	Statist weight
1	1950	0.000E+00	0.000E+00	--	*	7.371E-01	0.00000	1.000E+00
2	1951	0.000E+00	0.000E+00	--	*	8.745E-01	0.00000	1.000E+00
3	1952	0.000E+00	0.000E+00	--	*	1.017E+00	0.00000	1.000E+00
4	1953	0.000E+00	0.000E+00	--	*	1.120E+00	0.00000	1.000E+00
5	1954	0.000E+00	0.000E+00	--	*	1.232E+00	0.00000	1.000E+00
6	1955	0.000E+00	0.000E+00	--	*	1.321E+00	0.00000	1.000E+00
7	1956	0.000E+00	0.000E+00	--	*	1.337E+00	0.00000	1.000E+00
8	1957	0.000E+00	0.000E+00	--	*	1.331E+00	0.00000	1.000E+00
9	1958	0.000E+00	0.000E+00	--	*	1.275E+00	0.00000	1.000E+00
10	1959	0.000E+00	0.000E+00	--	*	1.275E+00	0.00000	1.000E+00
11	1960	0.000E+00	0.000E+00	--	*	1.360E+00	0.00000	1.000E+00
12	1961	0.000E+00	0.000E+00	--	*	1.406E+00	0.00000	1.000E+00
13	1962	0.000E+00	0.000E+00	--	*	1.435E+00	0.00000	1.000E+00
14	1963	0.000E+00	0.000E+00	--	*	1.473E+00	0.00000	1.000E+00
15	1964	0.000E+00	0.000E+00	--	*	1.507E+00	0.00000	1.000E+00
16	1965	0.000E+00	0.000E+00	--	*	1.515E+00	0.00000	1.000E+00
17	1966	0.000E+00	0.000E+00	--	*	1.517E+00	0.00000	1.000E+00
18	1967	0.000E+00	0.000E+00	--	*	1.537E+00	0.00000	1.000E+00
19	1968	0.000E+00	0.000E+00	--	*	1.509E+00	0.00000	1.000E+00
20	1969	0.000E+00	0.000E+00	--	*	1.485E+00	0.00000	1.000E+00
21	1970	0.000E+00	0.000E+00	--	*	1.474E+00	0.00000	1.000E+00
22	1971	0.000E+00	0.000E+00	--	*	1.465E+00	0.00000	1.000E+00
23	1972	0.000E+00	0.000E+00	--	*	1.460E+00	0.00000	1.000E+00
24	1973	0.000E+00	0.000E+00	--	*	1.445E+00	0.00000	1.000E+00
25	1974	0.000E+00	0.000E+00	--	*	1.450E+00	0.00000	1.000E+00
26	1975	0.000E+00	0.000E+00	--	*	1.413E+00	0.00000	1.000E+00
27	1976	0.000E+00	0.000E+00	--	*	1.230E+00	0.00000	1.000E+00
28	1977	0.000E+00	0.000E+00	--	*	9.742E-01	0.00000	1.000E+00
29	1978	0.000E+00	0.000E+00	--	*	8.495E-01	0.00000	1.000E+00
30	1979	0.000E+00	0.000E+00	--	*	8.460E-01	0.00000	1.000E+00
31	1980	0.000E+00	0.000E+00	--	*	7.438E-01	0.00000	1.000E+00
32	1981	0.000E+00	0.000E+00	--	*	6.516E-01	0.00000	1.000E+00
33	1982	0.000E+00	0.000E+00	--	*	6.617E-01	0.00000	1.000E+00
34	1983	0.000E+00	0.000E+00	--	*	6.645E-01	0.00000	1.000E+00
35	1984	0.000E+00	0.000E+00	--	*	7.023E-01	0.00000	1.000E+00
36	1985	0.000E+00	0.000E+00	--	*	7.685E-01	0.00000	1.000E+00
37	1986	1.000E+00	1.000E+00	--	6.018E-01	7.952E-01	-0.27860	1.000E+00
38	1987	1.000E+00	1.000E+00	--	7.332E-01	8.014E-01	-0.08892	1.000E+00
39	1988	1.000E+00	1.000E+00	--	8.852E-01	7.831E-01	0.12263	1.000E+00
40	1989	1.000E+00	1.000E+00	--	8.050E-01	7.767E-01	0.03570	1.000E+00
41	1990	1.000E+00	1.000E+00	--	8.727E-01	8.049E-01	0.08081	1.000E+00
42	1991	1.000E+00	1.000E+00	--	6.500E-01	7.921E-01	-0.19769	1.000E+00
43	1992	1.000E+00	1.000E+00	--	7.997E-01	7.798E-01	0.02518	1.000E+00
44	1993	1.000E+00	1.000E+00	--	9.346E-01	7.830E-01	0.17692	1.000E+00
45	1994	1.000E+00	1.000E+00	--	6.608E-01	7.712E-01	-0.15447	1.000E+00
46	1995	1.000E+00	1.000E+00	--	9.305E-01	7.972E-01	0.15458	1.000E+00
47	1996	1.000E+00	1.000E+00	--	8.161E-01	8.552E-01	-0.04681	1.000E+00
48	1997	1.000E+00	1.000E+00	--	7.986E-01	9.054E-01	-0.12559	1.000E+00
49	1998	1.000E+00	1.000E+00	--	7.746E-01	9.665E-01	-0.22126	1.000E+00
50	1999	1.000E+00	1.000E+00	--	9.512E-01	1.046E+00	-0.09509	1.000E+00
51	2000	1.000E+00	1.000E+00	--	9.821E-01	1.099E+00	-0.11248	1.000E+00
52	2001	1.000E+00	1.000E+00	--	9.760E-01	1.122E+00	-0.13970	1.000E+00
53	2002	1.000E+00	1.000E+00	--	9.360E-01	1.137E+00	-0.19476	1.000E+00
54	2003	1.000E+00	1.000E+00	--	1.010E+00	1.144E+00	-0.12468	1.000E+00
55	2004	1.000E+00	1.000E+00	--	1.314E+00	1.155E+00	0.12958	1.000E+00
56	2005	1.000E+00	1.000E+00	--	1.289E+00	1.168E+00	0.09846	1.000E+00
57	2006	1.000E+00	1.000E+00	--	1.373E+00	1.182E+00	0.14998	1.000E+00
58	2007	1.000E+00	1.000E+00	--	1.197E+00	1.184E+00	0.01084	1.000E+00
59	2008	1.000E+00	1.000E+00	--	1.233E+00	1.178E+00	0.04561	1.000E+00

60	2009	1.000E+00	1.000E+00	--	1.642E+00	1.175E+00	0.33486	1.000E+00
61	2010	1.000E+00	1.000E+00	--	1.410E+00	1.153E+00	0.20154	1.000E+00
62	2011	1.000E+00	1.000E+00	--	1.424E+00	1.151E+00	0.21288	1.000E+00

* Asterisk indicates missing value(s).

ESTIMATES FROM BOOTSTRAPPED ANALYSIS

Param name	Point estimate	Estimated bias in pt estimate	Estimated relative bias	Bias-corrected approximate confidence limits 80% lower	80% upper	50% lower	50% upper	Inter-quartile range	Relative IQ range
B1/K	3.787E-01	1.327E-03	0.35%	3.541E-01	4.069E-01	3.727E-01	3.871E-01	1.433E-02	0.038
K	4.478E+04	4.253E+00	0.01%	4.147E+04	4.870E+04	4.310E+04	4.698E+04	3.888E+03	0.087
q(1)	4.244E-05	4.644E-07	1.09%	3.741E-05	4.691E-05	3.972E-05	4.459E-05	4.876E-06	0.115
q(2)	4.042E-05	5.467E-07	1.35%	3.566E-05	4.407E-05	3.769E-05	4.225E-05	4.563E-06	0.113
MSY	6.484E+03	2.015E+01	0.31%	6.362E+03	6.605E+03	6.417E+03	6.545E+03	1.276E+02	0.020
Ye(2012)	5.950E+03	1.271E+01	0.21%	5.909E+03	6.009E+03	5.924E+03	5.975E+03	5.098E+01	0.009
Y.@Fmsy	8.344E+03	2.468E+01	0.30%	7.917E+03	8.754E+03	8.114E+03	8.565E+03	4.509E+02	0.054
Bmsy	2.239E+04	2.127E+00	0.01%	2.073E+04	2.435E+04	2.155E+04	2.349E+04	1.944E+03	0.087
Fmsy	2.896E-01	3.016E-03	1.04%	2.611E-01	3.185E-01	2.733E-01	3.035E-01	3.027E-02	0.105
fmsy(1)	6.823E+03	1.426E+01	0.21%	6.313E+03	7.431E+03	6.556E+03	7.157E+03	6.012E+02	0.088
fmsy(2)	7.164E+03	-2.178E+00	-0.03%	6.636E+03	7.928E+03	6.904E+03	7.578E+03	6.740E+02	0.094
B./Bmsy	1.287E+00	-5.982E-04	-0.05%	1.244E+00	1.325E+00	1.265E+00	1.308E+00	4.296E-02	0.033
F./Fmsy	6.404E-01	-7.541E-04	-0.12%	6.104E-01	6.738E-01	6.239E-01	6.579E-01	3.402E-02	0.053
Ye./MSY	9.177E-01	-6.560E-04	-0.07%	8.947E-01	9.404E-01	9.049E-01	9.296E-01	2.465E-02	0.027
q2/q1	9.524E-01	3.694E-03	0.39%	8.932E-01	1.011E+00	9.215E-01	9.820E-01	6.045E-02	0.063

INFORMATION FOR REPAST (Prager, Porch, Shertzer, & Caddy. 2003. NAJFM 23: 349-361)

Unitless limit reference point in F (Fmsy/F.): 1.561
CV of above (from bootstrap distribution): 0.3860E-01

NOTES ON BOOTSTRAPPED ESTIMATES:

- Bootstrap results were computed from 1000 trials.
- Results are conditional on bounds set on MSY and K in the input file.
- All bootstrapped intervals are approximate. The statistical literature recommends using at least 1000 trials for accurate 95% intervals. The default 80% intervals used by ASPIC should require fewer trials for equivalent accuracy. Using at least 500 trials is recommended.
- Bias estimates are typically of high variance and therefore may be misleading.

Trials replaced for lack of convergence: 0
Trials replaced for MSY out of bounds: 0
Trials replaced for q out-of-bounds: 0
Trials replaced for K out-of-bounds: 0
Residual-adjustment factor: 1.0479

Elapsed time: 0 hours, 1 minutes, 36 seconds.