Age-structured model for Alaska herring stocks

Steve Martell

December 16, 2016

Executive Summary

This document describes the proposed changes that have been made to the Age-structured assessment model for Alaska herring stocks.

The objective of this project was to review and modify the existing AD Model Builder Code for the Age-structured model for Alaska herring stocks (version 0.1 Jan 2015). The overarching objective of the modifications are: to improve numerical stability, ease of use, general flexibility for alternative structural assumptions, and estimation of observation and process error variance to better quantify uncertainty. The following list of bullets summarizes the proposed changes that have been implemented to date:

- Modifications to the Input Data File. Users can now specify estimates of observation error for each annual observation for: catch, egg surveys, mile milt days, and composition data.
- Modifications to the Control file. Changes to the control file now allow users to estimate or fix parameters, change the phase of estimation, set initial parameter values, apply informative priors of various statistical distributions, all without having to recompile the code. This permits rapid exploration (even automated) of alternative hypotheses and structural assumptions that are repeatable.
- Added controls for the addition of time varying natural mortality rates, blocks of time-varying maturity, a flexible system from implementing a wide variety of selectivity options including time-varying blocks, or continuous non-parametric functions (i.e., cubic splines). The control file is also structured so it can expand with new model features, or custom outputs, that develop in the future.
- Custom command line options were added to the code. Two options were added to permit rapid simulation testing (`-sim` option), and automate the procedures of conducting retrospective analysis without having to make any potentially dangerous modifications to input files (the `-retro` option).
- Many of the previous routines in the current version of the stock assessment model have been broken down into smaller functions. This both reduces the amount of redundant code that currently exists and makes the code easier to read and understand by humans.
- The model has 5 major components:

1. Inputs (includes data and controls that specifies model structure).
2. Population dynamics: a collection of sub-models that relate to the biology (e.g., natural mortality, maturity, stock-recruitment).
3. Observation dynamics: a collection of sub-models that relate how fishing mortality interacts with population model (e.g., fisheries selectivity, fishing mortality, predicted egg abundance index, predicted composition data).
4. Statistical criterion: the objective function that relates estimated model parameters to differences between observed and predicted variables.
5. Outputs: including and not limited to parameter estimates, convergence criterion, derived management quantities and residuals.

- There are a few structural differences being proposed in this model that relate to how selectivity is modeled, the observation error assumed in the composition data, and variance terms that relate to both process error and observation error.
  - To avoid breaking the derivative chain in calculating the objective function and its gradient, use of the max function to re-scale the selectivities should be avoided. Often you can get away with it in very simple models where selectivity is very well informed, but can soon become problematic when your jointly estimating additional parameters that are confounded with selectivity (e.g., time-varying natural mortality). To do so, the proposed change rescales the selectivity vector for ages such that it has a mean of 1.
  - The previous generation used a least-square estimator for the age-composition proportions. The proposed changes implemented in this model assume the age-proportion data are logistic-normal, and these data are weighted by the conditional maximum likelihood estimate of the variance (i.e., objectively weighted). Alternatives likelihood formulations are also easily implemented in future iterations.
  - Lastly, each catch and survey observation in the input data file also has an associated log standard error associated with it (approximately the coefficient of variation). In cases where it is possible to estimate a standard error in the data using bootstrap procedures, the inter-annual variation in observation error can now be specified. In addition, the process error term permits recruitment variation around a stock-recruitment relationship. Currently the Ricker model is implemented, with the option to implement the Beverton-Holt model annotated in the code.
- Additional elements were also introduced in the objective function calculation to improve the overall estimation robustness. These include penalties that are only implemented in the initial phases to set up initial gradients that will get key population parameters in the "ball park". These penalties can then be relaxed (or set = 0) in the terminal phases.
- Of significant difference is the use of informative prior distributions (or sometimes less informative) for population parameters including: natural mortality, initial recruitment, average recruitment, unfished recruitment, steepness of the stock recruitment relationship, and the variance in the recruitment deviations (process

error). The only option for including priors in the previous generation was to fix a parameter value (which implies the variance is 0, or very informative). For example, having the option to estimate natural mortality where the prior mean is set at the original fixed value and assume some arbitrary CV can often reduce model confounding in cases where there are one-way trips in the relative abundance data. Comparing the marginal posterior density and prior density will shed light on how informative the data are about the parameters.

- Model selection criterion can also be evaluated using Deviance Information Criterion (DIC). This criterion is calculated using the posterior sample values generated from one of AD Model Builders built-in sampling routines (e.g., The Metropolis Hastings Algorithm).

Lastly, a few R-scripts have been developed for the purposes of conducting simulation-estimation experiments for self-testing to examine for potential bias in the estimators, and exploring options for correcting any such bias.

An example assessment using the data for the 2015 Sitka herring stock is provided in this document. This example is not meant to be used as a comparison with other assessments for this stock. The intent of the example is to be illustrative. Finally, the scope of this project focused on the aforementioned points above, and primarily focuses on data weighting and estimation of uncertainty. There are many other graphical methods that could be explored to further communicate levels of uncertainty to fisheries managers, and I would refer you to the work of Dr. Ian Stewart at the Intl. Pacific Halibut Commission on communicating uncertainty to decision makers.

1 Acknowledgments

I greatly appreciate the feedback from the State of Alaska scientists who participated in the training workshop in Juneau Alaska, July 27-29, 2016. A special thank you to Dr. Sherri Dressel for organizing this workshop and inviting me to bid on this contract.

Contents

2 Introduction

This document describes the structural differences between the Age-structured model for Alaska herring (programmed by Pete Hulson, pete.hulson@noaa.gov). The original model was written in Excel, and translated to the AD Model Builder template language. The objective function was a simple weighted least-squares estimator that made numerous simplifying assumptions about the error structure in the data. The objective for the next generation of stock assessment models for Alaska herring stocks is to use a more modern statistical approach to fitting models to data collected from fisheries dependent and independent sources.

In this document we first decompose the original source code to understand how the various observations are related to structural assumptions in the model, and how these data are weighted when estimating model parameters. Next we describe a technical description of the proposed model changes, and provide both the equations and the AD Model Builder template code to document how the equations are actually implemented in the code. A simple example of a simulation-estimation experiment exploring estimation performance (i.e.,

precision and bias) is explained from both an observation and mixed observation-process error model. Following, a simple example of fitting the model to the Sitka herring stock using the example Sitka data and control files that are available in the code repository.

Having both the equations and the code serve a dual purpose, first it provides model documentation and second allows for new programmers to learn more about the language itself and implement changes to accommodate alternative structural assumptions. The model code is intended to be a living document, and I would encourage developers to use use the code repository, create branches and add or modify the existing code to tailor the model for specific applications or research projects.

3 Model deconstruction

This section is intended to serve three purposes: 1) to document the model structure given the code in model.tpl, 2) to detail proposed changes to the model code to improve overall numerical stability, and 3) provide a statistical approach that is amenable to maximum likelihood and Bayesian methods.

3.1 Model Structure

Table 1 begins with the objective function that is being minimized in the original Alaska herring model programmed by Pete Hulson. There are four components defined in (T1.1), where three of the four components are scaled by user defined coefficients λ. The first component is the commercial age-composition data (QC), the second is the spawning biomass age-composition (QS), the third is egg deposition data (WQE), and finally the fourth component is a penalty on the recruitment deviations from the underlying Ricker stock-recruitment model (QR).

For the commercial age-composition data, observation errors in the age-proportions are assumed to be normal (T1.2), where the predicted proportion-at-age (T1.3) is a function of the numbers-at-age (T1.4) and selectivity (T1.5). Note that in (T1.4) that the function is not continuous. In this case the selectivity curve is rescaled to have a maximum value of 1.0. The `max` operation in the denominator of this function breaks the derivative chain in AD Model Builder and can result in numerical problems during parameter optimization associated with corrupt derivative information.

The same normal error distribution is also assumed for the age-proportions in the spawner catch composition samples (T1.6). In this case, the age proportions are based on the mature numbers-at-age at the time of spawning, where the fishery removals are first subtracted from the mature numbers-at-age (T1.7). Note that this further assumes that all removals (i.e., fisheries selectivity) only harvests sexually mature fish. This assumption is inconsistent with (T1.4), where a different logistic curve is assumed for fisheries selectivity.

The catch-at-age data is internally derived in the model (T1.9) conditional on the numbers-at-age and the estimated selectivity. The model further assumes the total catch (in short tons) is measured without error. This is also referred to as conditioning the model on catch.

The residual sum of squares for the egg deposition survey is given by (T1.10). In this case the observation errors are assumed to be log-normal, and each year's observation is weighted by the inverse variance of sampling observation errors (φ_i).

Table 1: Decomposition of the objective function based on the source code provided in `model.tpl`. The objective function f is what AD Model Builder is trying to minimize. Note that $\dot{}$ represents mature state variables (e.g., mature weight-at-age $\dot{w}_j$)

Objective function

$$f = \lambda_C QC + \lambda_S QS + WQE + \lambda_R QR \tag{T1.1}$$

Commercial catch proportion-at-age

$$QC = \sum_i \sum_j (\hat{Q}_{i,j} - Q_{i,j})^2 \qquad \hat{Q}_{i,j} \text{ observed proportions-at-age} \tag{T1.2}$$

$$Q_{i,j} = \frac{V_{i,j}}{\sum_j V_{i,j}} \qquad Q_{i,j} \text{ predicted proportion-at-age} \tag{T1.3}$$

$$V_{i,j} = N_{i,j} \frac{S_{i,j}}{\max(S_{i,j})} \qquad V_{i,j} \text{ vulnerable numbers-at-age} \tag{T1.4}$$

$$S_{i,j} = \frac{1}{1 + \exp(-g_i(j - a_i))} \qquad S_{i,j} \text{ Selectivity in year } i \text{ for age } j \tag{T1.5}$$

Spawn proportion-at-age

$$QS = \sum_i \sum_j (\hat{O}_{i,j} - O_{i,j})^2 \qquad \hat{O}_{i,j} \text{ observed proportion-at-age} \tag{T1.6}$$

$$O_{i,j} = \frac{\dot{N}_{i,j}}{\sum_j \dot{N}_{i,j}} \qquad O_{i,j} \text{ predicted proportion mature-at-age} \tag{T1.7}$$

$$\dot{N}_{i,j} = N_{i,j} M_{i,j} - C_{i,j} \qquad \dot{N}_{ij} \text{ Number-at-age at spawning} \tag{T1.8}$$

$$C_{i,j} = \frac{\hat{c}_i Q_{i,j}}{\sum_j Q_{i,j} w_j} \qquad \hat{c}_i \text{ observed catch, } w_j \text{ weight-at-age} \tag{T1.9}$$

Egg deposition survey

$$WQE = \sum_i \varphi_i \left[\ln(\hat{y}_i) - \ln(y_i)\right]^2 \qquad \hat{y}_i \text{ observed egg deposition, } \varphi_i \text{ weight} \tag{T1.10}$$

$$y_i = 0.5 \sum_j O_{i,j}(\rho_i \dot{w}_{i,j} - \alpha_i) \qquad \rho_i \text{ and } \alpha_i \text{ fecundity-weight regression} \tag{T1.11}$$

Penalized recruitment deviations

$$QR = \sum_i^{(I-k)} \left[\ln\left(N_{i+k,k}\right) - \ln\left(f(\dot{N}_{i,j})\right)\right]^2 \qquad N_{i+k,k} \text{ number of age } k \text{ recruits} \tag{T1.12}$$

$$f(\dot{N}_{i,j}) = a\dot{B}_i \exp(-b\dot{B}_i) \qquad \text{Ricker stock-recruitment} \tag{T1.13}$$

$$\dot{B}_i = \sum_j \dot{N}_{i,j} \dot{w}_j \qquad \dot{B}_i \text{ mature biomass at spawning} \tag{T1.14}$$

4 Technical description of the proposed model changes

4.1 Input Data

The best resource for looking at the input data is the html file that describes the input data. There are 7 major sections to the data file.

1. Model dimensions.
2. Fecundity regression coefficients.
3. Total Annual Catch.
4. Empirical Weight-at-age (spawn and commercial).
5. Age-composition (spawn and commercial).
6. Egg deposition data.
7. Mile milt days.

These are the same data inputs that were used in the ASA model; however, there have a been a number of significant changes to the input data file. The most significant change is the addition of the log standard error for each catch, egg deposition, and spawn survey observation.

4.2 Control file

There are also significant changes to the control file. In fact, it's a completely different control file than what was used in the ASA model. Again, the best resource for looking at the specific contents of the control file is the control file itself.

To highlight some of the major changes, the control file now consists of a design matrix for controlling the leading model parameters; specifically, the bounds and phases in which these parameters are estimated. There is a block for time-varying maturity, a block for time-varying natural mortality rate deviations, a block for selectivity, where the user can choose among alternative parametric and non-parametric selectivity curves. Lastly, there is a vector of other miscellaneous model controls for, *inter Alia*, re-scaling catch, conditioning the model on catch or effort.

4.3 Age-schedule information

Empirical weight-at-age data are part of the input data file. Maturity-at-age is assumed to follow a logistic function with age, where the parameters of the logistic function are estimated within the model.

4.3.1 Maturity-at-age

For the maturity-at-age, the HAM assumes that age-specific maturity follows a logistic relationship, where the estimated parameters define the age-at-50% maturity and the standard deviation, for each unique block period (the block periods are user defined).

The source code for the TPL file is as follows:

```
FUNCTION void initializeMaturitySchedules()
  int iyr = mod_syr;
  int jyr = 0;
  mat.initialize();

  for(int h = 1; h <= nMatBlocks; h++) {
    dvariable mat_a50 = mat_params(h,1);
    dvariable mat_a95 = mat_params(h,2);

    jyr = (h != nMatBlocks) ? nMatBlockYear(h) : nMatBlockYear(h)-retro_yrs;

    // fill maturity array using logistic function
    do{
      mat(iyr++) = plogis95(age,mat_a50,mat_a95);
    } while(iyr <= jyr);
  }
```

where `plogis` is a built in ADMB function for the logistic:

$$f(x) = (1 + \exp(-(x - \mu)/\sigma))^{-1}$$

where μ and σ are the location and scale parameters that are estimated.

4.3.2 Natural mortality

Natural mortality is both age- and time-specific. At the time of writing, there is no code that allows for age-dependent natural mortality, but this option is easily added as a feature to the HAM.

The source code for the TPL file is as follows:

```
FUNCTION void calcNaturalMortality()

  int iyr = mod_syr;
  int jyr;
  Mij.initialize();
  switch(mort_type) {
    case 1: // constant M within block.
      for(int h = 1; h <= nMortBlocks; h++){
        dvariable mi = mfexp(log_natural_mortality + log_m_devs(h));

        jyr = h != nMortBlocks?nMortBlockYear(h):nMortBlockYear(h)-retro_yrs;
        // fill mortality array by block
        do{
          Mij(iyr++) = mi;
```

```
      } while(iyr <= jyr);
    }
    break;
    case 2: // cubic spline
      dvector iiyr = dvector((nMortBlockYear - mod_syr)/(mod_nyr-mod_syr));
      dvector jjyr(mod_syr,mod_nyr);
      jjyr.fill_seqadd(0,double(1.0/(mod_nyr-mod_syr)));
      dvar_vector mi = log_natural_mortality + log_m_devs;
      vcubic_spline_function cubic_spline_m(iiyr,mi);
      dvar_vector mtmp = cubic_spline_m(jjyr);
      for(int i=mod_syr; i <= mod_nyr; i++)
      {
        Mij(i) = mfexp(mtmp(i));
      }
    break;
  }
```

where the Matrix $M_{i,j}$ is the instantaneous natural mortality rate for year i and age j. At this point the code just fills each row of the matrix with the same annual natural mortality rate (i.e., age-independent). This is where you would want to modify the code to allow for age-dependent natural mortality rates.

4.3.3 Selectivity

Currently only the logistic selectivity option is implemented. But the source code is structured such that alternative parametric and non-parametric functions can be easily added to the source code using a switch statement.

The source code for the TPL file is as follows:

```
FUNCTION void calcSelectivity()
  /**
    - Loop over each of the selectivity block/pattern
    - Determine which selectivity type is being used.
    - get parameters from log_slx_pars
    - calculate the age-dependent selectivity pattern
    - fill selectivty array for that block.
    - selectivity is scaled to have a mean = 1 across all ages.
  */
  dvariable p1,p2;
  dvar_vector slx(sage,nage);
  log_slx.initialize();



  for(int h = 1; h <= nSlxBlks; h++){

    switch(nSelType(h)){
      case 1: //logistic
        p1  = mfexp(log_slx_pars(h,1,1));
        p2  = mfexp(log_slx_pars(h,1,2));
        slx = plogis(age,p1,p2) + TINY;
```

```
      break;
    }

    int jyr = h != nSlxBlks ? nslx_nyr(h):nslx_nyr(h)-retro_yrs;
    for(int i = nslx_syr(h); i <= jyr; i++){
      log_slx(i) = log(slx) - log(mean(slx));
    }
  }
  Sij.sub(mod_syr,mod_nyr) = mfexp(log_slx);
```

The matrix $S_{i,j}$ is the relative selectivity for age j in year i. Additional functions for computing the vector `slx` can be new cases (e.g. case 2: // coefficients).

In this model, selectivity is parameterized to have a mean value of 1.0. The reason for this particular parameterization is to ensure the objective function remains continuous and differentiable. In each year, the vector of age-specific selectivity coefficients is scaled to have a mean value of 1.0, rather than have an asymptote of 1.0. This is accomplished, in log-space, by subtracting the mean from the vector of age-specific selectivities. This avoids having to use the max function; the use of the max function can lead to a discontinuity in the objective function which result in non-convergence. The trade-off for this numerical stability is that the interpretation of fishing mortality rates changes. The estimator is calculating the average-age-specific fishing mortality rate. The fully-selected fishing mortality rate is more commonly used metric, and this is easily accommodated post-estimation by re-scaling the vector of fishing mortality rates by the maximum age-specific selectivity in each year.

Note that the trends in the mean fishing mortality rates already integrates the changes in selectivity over time. These estimates of F probably better reflect the measure of fishing effort (i.e., mile-hours fished). Whereas, asymptotic estimates reflect the trends in fishing mortality rates for "fully recruited" cohorts and are less likely to reflect fishing intensity.

4.4 Fishing mortality

If the model is conditioned on effort, then a routine that calculates the age-specific fishing mortality rate is invoked. In this routine, a vector of fishing mortality rate parameters, in log-space, is estimated, and the age-specific fishing mortality rate is the product of the fishing rate and age-specific selectivity. The source code for this routine is as follows:

```
FUNCTION void calcFishingMortalitiy()
  /**
    - Calculate Fishing mortality, and then Zij = Mij + Fij
    */

  for(int i = mod_syr; i <= mod_nyr; i++) {
    Fij(i) = exp(log_ft_pars(i)) * Sij(i);
  }
```

If the model is conditioned on catch (i.e., the method the ASA model uses), then there is a resulting difference equation which has the potential to result in negative numbers-at-age, which results in negative infinity in log-space. To guard against this, a simple solution

might be to use a max function to ensure that a positive number is returned. However, this is yet another occurrence where the objective function is discontinuous and subject to non-convergence issues. AD Model Builder has a function `posfun` that can be used to ensure the objective function remains continuous and differentiable.

4.5 Population dynamics

Estimated parameters for the population dynamics model include the initial abundance of ages 3-9+ for the initial year, abundance of age-3 recruits each year, and the natural mortality rate. In the original parameterization of the model, these initial recruitments and the vector of initial numbers-at-age result in creating $(N + A - 1)$ scaling parameters. To reduce the potential confounding with other global scaling parameters, updates to the model code include estimation of two recruitment scaling parameters, and two vectors that represent deviations from the mean. This modification reduces the potential for parameter confounding among the many parameters that affect global scaling (i.e., catchability coefficients, natural mortality rates, average recruitment or unfished biomass).

4.5.1 Initial state variables.

In this routine, the objective is to set the initial values for the numbers-at-age matrix $N_{i,j}$. Specifically, the row dimensions of the matrix are from the start year to the terminal year + 1, the column dimensions are the ages. This routine first calculates the survivorship to age j based on natural mortality rates, then initializes the first row of $N_{i,j}$ matrix using the average initial recruitment and deviations around that average recruitment multiplied by the survivorship at age j. The source code also includes the Taylor series for the plus group:

```
FUNCTION void initializeStateVariables()
  /**
    - Set initial values for numbers-at-age matrix in first year
      and sage recruits for all years.
    */
  Nij.initialize();

  // initialize first row of numbers-at-age matrix
  // lx is a vector of survivorship (probability of surviving to age j)
  dvar_vector lx(sage,nage);

  for(int j = sage; j <= nage; j++){

    lx(j) = exp(-Mij(mod_syr,j)*(j-sage));
    if( j==nage ) lx(j) /= (1.0-exp(-Mij(mod_syr,j)));


    if( j > sage ){
      Nij(mod_syr)(j) = mfexp(log_rinit + log_rinit_devs(j)) * lx(j);
    }
  }
```

```
// iniitialize first column of numbers-at-age matrix
for(int i = mod_syr; i <= mod_nyr + 1; i++){
  Nij(i,sage) = mfexp(log_rbar + log_rbar_devs(i));
}
```

The survivorship calculation `lx` corresponds to (T2.2) in Table 2, and the numbers-at-age in the initial year and age-3 recruits corresponds to (T2.3) and (T2.4), respectively.

4.5.2 Update state variables

In this routine, the numbers-at-age are propagated in time, where the-age specific survival rate is partitioned into two periods: a fishing period, and a period of natural mortality. The ASA model currently in use for Sitka Sound herring assumes a pulse fishery. At the start of each time step, the model first calculates the predicted catch-at-age in numbers in year i. This is done by first converting the catch-in-weight to catch-in-numbers using the predicted average weight of the catch. This corresponds to the `wbar` variable in the following code chunk (note the dependency on predicted proportions-at-age):

```
FUNCTION void updateStateVariables()
  /**
  - Update the numbers-at-age conditional on the catch-at-age.
  - Assume a pulse fishery.
  - step 1 calculate a vector of vulnerable-numbers-at-age
  - step 2 calculate vulnerable proportions-at-age.
  - step 3 calc average weight of catch (wbar) conditional on Qij.
  - step 4 calc catch-at-age | catch in biomass Cij = Ct/wbar * Qij.
  - step 5 condition on Ft or else condition on observed catch.
  - step 6 update numbers-at-age (using a very dangerous difference eqn.)

  Nov 30, 2016
  - added options for simulation model:
    - b_simulation_flag is true, then condition model on catch & S-R curve.

  */

  Qij.initialize();
  Cij.initialize();
  Pij.initialize();
  dvariable wbar;   // average weight of the catch.
  dvar_vector vj(sage,nage);
  //dvar_vector pj(sage,nage);
  dvar_vector sj(sage,nage);


  for(int i = mod_syr; i <= mod_nyr; i++){
```

```
    // step 1.
    vj = elem_prod(Nij(i),Sij(i));

    // step 2.
    Qij(i) = vj / sum(vj);

    // ADF&G's approach.
    if( !dMiscCont(2) ) {
      // step 3.
      dvector wa = data_cm_waa(i)(sage,nage);
      wbar = wa * Qij(i);

      // step 4.
      Cij(i) = data_catch(i,2) / wbar * Qij(i);
      // should use posfun here
      Pij(i) = posfun(Nij(i) - Cij(i),0.01,fpen);

      // step 6. update numbers at age
      sj = mfexp(-Mij(i));
      Nij(i+1)(sage+1,nage) =++ elem_prod(Pij(i)(sage,nage-1),
                                          sj(sage,nage-1));
      Nij(i+1)(nage) += Pij(i,nage) * sj(nage);
    }
    // step 5.
    // Condition on Ft
    else {
      // add flexibility here for Popes approx, or different seasons
      Pij(i) = elem_prod( Nij(i), exp(-Fij(i)) );
      Cij(i) = elem_prod( Nij(i), 1.-exp(-Fij(i)));

      // the following assumes fishery is at the start of the year.
      dvar_vector zi = Mij(i) + Fij(i);
      Nij(i+1)(sage+1,nage) = ++ elem_prod(Nij(i)(sage,nage-1),
                                           mfexp(-zi(sage,nage-1)));
      Nij(i+1)(nage) += Nij(i,nage) * mfexp(-zi(nage));
    }
  }
```

Once the catch-at-age data is calculated, the pulse fishery proceeds by subtracting the C_{ij} from the $N_{i,j}$. The last two steps correspond to step 4 in the annotated code chunk. The last steps the survive the remaining numbers-at-age using the natural mortality rate in year i. Then finally, update the numbers-at-age j to $j+1$ in year i to year $i+1$, including the plus group for the terminal age-group.

4.5.3 Stock-recruitment & Spawning stock biomass

The spawning stock biomass (after roe-fishery removal) is the product of the remaining numbers-at-age, the maturity-at-age, and the weight-at-age from spawn samples. The equation is defined in (T2.10) in Table 2. Note that the lag between spawning biomass and age-3 recruits is taken into account.

The stock recruitment relationship assumes that recruitment follows a Ricker type. The form of the Ricker model is as follows

$$R_i = s_o B_i \exp(-\beta B_i + \epsilon_i)$$

where the parameter s_o is the slope at the origin (or maximum number of recruits per spawning unit), and β defines slope of $\ln(R_i/B_i)$ versus the independent variable B_i. These two parameters are derived from the leading parameters the unfished recruitment R_0 and the steepness of the stock recruitment relationship as defined by (Mace and Doonan, 1988).

The source code for the stock-recruitment relationship is well annotated and describes some of the derivations:

```
FUNCTION void calcSpawningStockRecruitment()
  /**
  The functional form of the stock recruitment model follows that of a
  Ricker model, where R = so * SSB * exp(-beta * SSB). The two parameters
  so and beta where previously estimated as free parameters in the old
  herring model. Herein this fucntion I derive so and beta from the
  leading parameters Ro and reck; Ro is the unfished sage recruits, and reck
  is the recruitment compensation parameter, or the relative improvement in
  juvenile survival rates as the spawning stock SSB tends to 0. Simply a
  multiple of the replacement line Ro/Bo.

  At issue here is time varying maturity and time-varying natural mortality.
  When either of these two variables are assumed to change over time, then
  the underlying stock recruitment relationship will also change. This
  results in a non-stationary distribution. For the purposes of this
  assessment model, I use the average mortality and maturity schedules to
  derive the spawning boimass per recruit, which is ultimately used in
  deriving the parameters for the stock recruitment relationship.
    */

  /*
    Spoke to Sherri about this. Agreed to change the equation to prevent
    substracting immature fish from the numbers before calculating SSB.
  */
  for(int i = mod_syr; i <= mod_nyr; i++){
    //Oij(i) = elem_prod(mat(i),Nij(i));
    //ssb(i) = (Oij(i) - Cij(i)) * data_sp_waa(i)(sage,nage);

    Oij(i) = elem_prod(mat(i),Nij(i)-Cij(i));
    ssb(i) = Oij(i) * data_sp_waa(i)(sage,nage);
  }


  // average natural mortality
  dvar_vector mbar(sage,nage);
  int n = Mij.rowmax() - Mij.rowmin() + 1;
  mbar  = colsum(Mij)/n;

  // average maturity
```

```
dvar_vector mat_bar(sage,nage);
mat_bar = colsum(mat)/n;


// unfished spawning biomass per recruit
dvar_vector lx(sage,nage);
lx(sage) = 1.0;
for(int j = sage + 1; j <= nage; j++){
  lx(j) = lx(j-1) * mfexp(-mbar(j-1));
  if(j == nage){
    lx(j) /= 1.0 - mfexp(-mbar(j));
  }
}
dvariable phie = lx * elem_prod(avg_sp_waa,mat_bar);

// Ricker stock-recruitment function
// so = reck/phiE; where reck > 1.0
// beta = log(reck)/(ro * phiE)

// Beverton Holt use:
// beta = (reck - 1.0)/(ro *phiE)
ro    = mfexp(log_ro);
reck = mfexp(log_reck) + 1.0;
so    = reck/phie;
beta = log(reck) / (ro * phie);

spawners = ssb(mod_syr,mod_nyr-sage+1).shift(rec_syr);
recruits = elem_prod( so*spawners , mfexp(-beta*spawners) );
resd_rec = log(column(Nij,sage)(rec_syr,mod_nyr+1)+TINY)
           - log(recruits+TINY);
```

There is an issue with regards to calculating reference points in cases where there is non-stationarity (i.e., any time varying parameters such as natural mortality, maturity etc.). At what period should be used to define the average weight-at-age for spawning herring? What period should be used for calculating the average maturity? All of these are subjective decisions, and based on my experience will have little impact on the overall fit to the data, but could have major implications for harvest policy changes.

4.6 Observation models

4.6.1 Age composition

The predicted age-composition is based on the vulnerable proportions at age $Q_{i,j}$ in (T2.5). The residual difference is used to compute the negative log likelihood in the objective function. The code for the age composition residual calculation is as follows:

```
FUNCTION void calcAgeCompResiduals()
  /**
  - Commercial catch-age comp residuals
  - Spawning survey catch-age comp residuals.
```

```
  */

  resd_cm_comp.initialize();
  resd_sp_comp.initialize();
  for(int i = mod_syr; i <= mod_nyr; i++){

    // commercial age-comp prediction
    pred_cm_comp(i) = Qij(i);
    if( data_cm_comp(i,sage) >= 0 ){
      resd_cm_comp(i) = data_cm_comp(i)(sage,nage) - pred_cm_comp(i);
    }

    // spawning age-comp prediction
    pred_sp_comp(i) = (Oij(i)+TINY) / sum(Oij(i)+TINY);
    if( data_sp_comp(i,sage) >= 0 ){
      resd_sp_comp(i) = data_sp_comp(i)(sage,nage) - pred_sp_comp(i);
    }
  }
```

Note that both the residual difference between the commercial and spawning samples are calculated in this routine. If there are missing data for a given year (denoted by a -9.0 in the data file), then the residual difference is set to 0 for that year and there is no contribution to the negative log likelihood.

4.6.2 Egg deposition

The observation model for the egg deposition data treats these observations as estimates of absolute abundance. Therefore, there is no associated scaling parameter that is estimated. The observation errors in the egg deposition data are assumed to be log-normal. The predicted age-deposition data is based on the female (assuming a 50:50 sex ratio) mature numbers-at-age multiplied by the fecundity at age. The annotated source code is as follows:

```
FUNCTION void calcEggSurveyResiduals()
  /**
  - Observed egg data is in trillions of eggs
  - Predicted eggs is the mature female numbers-at-age multiplied
    by the fecundity-at-age, which comes from a regession of
    fecundity = slope * obs_sp_waa - intercept
  - Note Eij is the Fecundity-at-age j in year i.
  -
  */
  resd_egg_dep.initialize();
  for(int i = mod_syr; i <= mod_nyr; i++){
    pred_egg_dep(i) = (0.5 * Oij(i)) * Eij(i);
    if(data_egg_dep(i,2) > 0){
      resd_egg_dep(i) = log(data_egg_dep(i,2)) - log(pred_egg_dep(i));
    }
  }
```

4.6.3 Aerial surveys

The aerial survey index, or mile milt days, are treated as relative abundance data. The underlying assumption in this observation model is that the observation errors are log normal, and that the index is proportional to the spawning stock biomass. Note that the code does not estimate the coefficient, rather the conditional maximum likelihood estimate of the scaling coefficent is used (see Walters and Ludwig, 1994, for a full explanation). The annotated code follows:

```
FUNCTION void calcMiledaySurveyResiduals()
  /**
  - Assumed index from aerial survey is a relative abundance
  index. The slope of the regression ln(SSB) = q * ln(MileMilt) + 0
  is computed on the fly in case of missing data.
  - See Walters and Ludwig 1994 for more details.
  */
  resd_mileday.initialize();
  pred_mileday.initialize();
  int n = 1;
  dvar_vector zt(mod_syr,mod_nyr); zt.initialize();
  dvariable zbar = 0;
  for(int i = mod_syr; i <= mod_nyr; i++){
    if(data_mileday(i,2) > 0){
      zt(i) = log(data_mileday(i,2)) - log(ssb(i));
      zbar  = zt(i)/n + zbar *(n-1)/n;
      n++ ;
    }
  }
  pred_mileday = ssb * exp(zbar);
  resd_mileday = zt - zbar;
```

4.6.4 Predicted catch

In the case where the model is conditioned on effort and fitted to the catch time series data, the predicted catch is the sum of products between the catch-at-age in numbers and the observed weight-at-age in the commercial fishery. We further assume observation errors are log-normal. The source code follows:

```
FUNCTION void calcCatchResiduals()
  /**
  - Catch residuals assuming a lognormal error structure.
  */
  pred_catch.initialize();
  resd_catch.initialize();
  for(int i = mod_syr; i <= mod_nyr; i++) {
    if(data_catch(i,2) > 0) {
      pred_catch(i) = Cij(i) * data_cm_waa(i)(sage,nage);
      resd_catch(i) = log(data_catch(i,2)) - log(pred_catch(i));
    }
  }
```

4.7 Objective function

The objective function is organized into two sections, the first contains the negative log-likelihoods for the data given the model parameters. The second are a series of penalties, in the case of maximum likelihood estimation, prior density functions in the case of a Bayesian estimation.

4.7.1 Negative log-likelihoods

The negative log-likelihoods There are five 6 negative log-likelihoods functions in the objective function that correspond to the 5 different data elements and the residual process errors associated with a stock-recruitment relationship. Table 3 summarizes the available options currently implemented.

The source code for the negative loglikelihoods follows:

```
// 1. Negative loglikelihoods
nll.initialize();

// Mulitvariate logistic likelihood for composition data.
double sp_tau2;
double minp = 0.00;
int t1 = mod_syr;
int t2 = mod_nyr;
dmatrix d_sp_comp = trans(trans(data_sp_comp).sub(sage,nage)).sub(t1,t2);
nll(1) = dmvlogistic(d_sp_comp,pred_sp_comp,resd_sp_comp,sp_tau2,minp);

// Mulitvariate logistic likelihood for composition data.
double cm_tau2;
dmatrix d_cm_comp  = trans(trans(data_cm_comp).sub(sage,nage)).sub(t1,t2);
nll(2) = dmvlogistic(d_cm_comp,pred_cm_comp,resd_cm_comp,cm_tau2,minp);

// Negative loglikelihood for egg deposition data
dvector std_egg_dep = TINY + column(data_egg_dep,3)(t1,t2);
nll(3) = dnorm(resd_egg_dep,std_egg_dep);

// Negative loglikelihood for milt mile day
dvector std_mileday = TINY + column(data_mileday,3)(t1,t2);
nll(4) = dnorm(resd_mileday,std_mileday);

// Negative loglikelihood for stock-recruitment data
dvariable std_rec = log_sigma_r;
nll(5) = dnorm(resd_rec,std_rec);

// Negative loglikelihood for catch data
dvector std_catch = column(data_catch,3)(t1,t2);
nll(6) = dnorm(resd_catch,std_catch);
```

For the composition data, the multivariate logistic likelihood is currently implemented, as this is a self-scaling likelihood. A good reference for this particular likelihood and how it's implemented in AD Model Builder can be found in Schnute and Richards (1995). More

recent work on the weighting of composition data is available in Francis (2011). The function `dmvlogistic` requires 5 arguments: the observed and predicted composition matrices, a matrix for returning the residuals, a variable for the conditional MLE of the variance of the observation errors, and a threshold value called `minp`. The multivariate logistic likelihood does not accommodate 0 observations for age proportions. Therefore there are two options for dealing with 0s: 1) add a small constant to all observed and predicted observations and re-normalize, or 2) pool the adjacent cohorts if the observed proportion is less than some minimum observed proportion. The first option is widely used in programs such as stock synthesis, and is akin to manufacturing data. If a particular cohort is relatively weak, and only partially selected, the sample size required to observe just one individual of a certain age in a given year could be infinitely large. Instead of adding a constant, option 2 pools the data such that the likelihood of ages, for example, 3-4 are evaluated jointly, rather than the likelihood of age-3 plus the likelihood of age-4. This pooling of the data and predictions does not require the addition of a constant that could potentially bias the result. A good practice that I've found is to just set `minp=0`, and then conduct sensitivity tests using where proportions less than 1% or 2% etc are pooled into the adjacent cohort.

In the case of the likelihoods for the egg deposition index, aerial surveys, and catch data, the function `dnorm` is used, where the arguments are the vector of residuals, and a vector of standard-deviations for each observation. Note that you can effectively turn individual years of data off by setting the `log.se` value to a large number (e.g., 5.0).

The current version of the source code will estimate an annual fishing mortality rate for each year (when the model is conditioned on effort only). This can become problematic in the case where the catch is 0 for a particular year. A technical point in cases where the catch is 0. In this case the MLE for the fishing mortality rate is also 0, but the derivative is undefined because the observed standard deviation is also 0. In such cases, estimates of the standard error for the fishing mortality rate in a year with 0 catch will be undefined. One option to explore for simulation studies (i.e., when using the -sim argument) is to modify the original data file and add a small, insignificant amount, of catch to ensure the simulation model generates some data, otherwise spurious results may occur in simulation studies.

4.7.2 Penalties

Currently the code for the penalties is as follows:

```
dmatrix d_sp_comp = trans(trans(data_sp_comp).sub(sage,nage)).sub(t1,t2);
nll(1) = dmvlogistic(d_sp_comp,pred_sp_comp,resd_sp_comp,sp_tau2,minp);

// Mulitvariate logistic likelihood for composition data.
double cm_tau2;
dmatrix d_cm_comp  = trans(trans(data_cm_comp).sub(sage,nage)).sub(t1,t2);
nll(2) = dmvlogistic(d_cm_comp,pred_cm_comp,resd_cm_comp,cm_tau2,minp);

// Negative loglikelihood for egg deposition data
dvector std_egg_dep = TINY + column(data_egg_dep,3)(t1,t2);
nll(3) = dnorm(resd_egg_dep,std_egg_dep);
```

```
// Negative loglikelihood for milt mile day
dvector std_mileday = TINY + column(data_mileday,3)(t1,t2);
nll(4) = dnorm(resd_mileday,std_mileday);

// Negative loglikelihood for stock-recruitment data
dvariable std_rec = log_sigma_r;
```

The penalties are implemented in a phased approach, where in the initial phases of parameter estimation, the penalties initially have small standard deviations. This increases the overall efficiency of the non-linear search routine to help resolve the overall scaling. Then in the terminal phase, these penalties are relaxed (increased variance) such that they provide little or no influence on the gradient structure. A similar strategy is also used with the recruitment deviation parameters.

5 Simulation example

Built into the assessment model code is a command line option that conducts a simulation experiment. In this experiment, the input parameter values are known, and simulated observations are generated with known error distributions. The model then proceeds to estimate the model parameters using built-in optimization procedures. Any potential biases can be detected by comparing the estimated values with the true values. This is best done using a Monte Carlo experiment where the distributions of estimated parameters are compared with the true values used in simulating the data.

For this simulation example, the data from the Sitka herring stock will be used. I first fit the model to the Sitka data to obtain the following parameter input file. Next, save this file as `ham.pin`, and by default the model will read these parameters when the application is first initialized. These parameters saved in the `ham.pin` file can be modified to specific values defined by the user.

```
# Number of parameters = 106  Objective function value = -1067.91  Maximum gradient component =
    0.000146673
# theta[1]:
-0.503402545573
# theta[2]:
4.19024390238
# theta[3]:
5.65105702886
# theta[4]:
6.34259911617
# theta[5]:
1.28108178211
# theta[6]:
0.614742804759
# log_rinit_devs:
 -0.369099857438 0.0432573918353 -0.232452148129 0.450637740947 0.107656872785
# log_rbar_devs:
 -1.71838525570 -1.67116045858 -0.785130704400 -2.32065685005 -2.43830223672 -3.16312286545 -1.59961940954
      -0.183637375474 1.32240176081 -0.137209104183 -1.28834862444 -0.411898580727 1.04415812632
     0.262636124152 -1.13914483251 -0.0891842359559 1.80064719129 -0.255869889506 -1.74476380697
     -1.03396568671 1.88568689495 -0.546576451091 -1.48967523844 -0.994032025259 0.524164900236
     0.347541168834 1.14799261913 1.19578236045 -0.221319825319 0.862572980916 1.10468337111
     0.751162194596 2.06816736234 0.159887259205 0.751872098920 0.987472284687 1.40415756341 1.64452226242
      1.40874746193 1.41059964860 0.330241657355 -0.0165228217769 1.04703009587 -1.94748796339
     1.22905663405 0.504830220587
# mat_params[1]:
 4.50000000000 7.00000000000
# log_m_devs:
 0.00000000000 0.00000000000 0.00000000000 0.00000000000
# log_slx_pars[1]:
 1.09861228867 -0.693147180560
# log_slx_pars[2]:
```

```
 1.50743640058  -0.331659075481
# log_slx_pars[3]:
 1.77240606047  0.0107003869354
# log_ft_pars:
 -2.98070476911  -3.11738199477  -3.11884859564  -3.03984330553  -1.82381258039  -2.08980608571  -8.91395057202
      -4.69390362291  -3.29132526822  -2.76846814111  -2.71491299020  -2.30490013629  -2.16575297377
      -2.17788654497  -1.92424610026  -2.10886395022  -2.54065392177  -1.84657016221  -1.51265689328
      -2.50451696325  -3.26161715876  -2.52673542945  -1.85876833453  -2.47927928999  -2.90263367564
      -1.68711715035  -1.73349724696  -2.41633464261  -2.21695495405  -2.99506697086  -1.90955934394
      -2.10428618377  -2.71340347735  -2.38404572145  -2.38594313472  -2.52440253674  -2.37870137084
      -2.19838420627  -2.25348830854  -2.09952696950  -2.00882977224  -2.15518189265  -3.04941561245
      -1.69527908160  -2.08737142634
```

Next, run the model using the `-sim` flag using the number 1 as the random number seed. At the command line you can run:

```
./ham -sim 1
```

When you execute the above command, the following sequence summarizes the events that occur inside the simulation model:

1. Set random number seed for random variables.
2. Read in the model data.
3. Read in the initial parameter values from `ham.par`.
4. Run the simulation model:
   (a) Call the population dynamics subroutines.
   (b) Call the observation models to generate predicted observations.
   (c) Add random errors to the simulated observations.
   (d) Overwrite the input data in memory with the simulated data.
5. After the simulated data has been created, the model proceeds with its normal routines for conducting parameter estimation.
6. Simulated observations are printed to the report file.
7. Estimated parameters in the `ham.par` file should be comparable with the true parameter (`ham.pin`) values that were used in generating the simulated data.

This approach to simulation-estimation is called "self-testing" where both the same model is used to generate the data, as well as, in fitting the model to data. Arguably, this approach should generate exact parameter estimates in the case where data are generated with no simulated observation errors.

5.1 Observation error

For the purposes of demonstration, Figure 5.1 shows the results of conditioning the simulation model on the Stika herring data. In these simulation-estimation experiments, all of the simulations were conditioned on the same fishing mortality rates and recruitment vectors

that were estimated from fitting the model to the Sitka data. The only difference between the 8 simulation models are the random numbers in the observation errors.

The simple test should result in estimates of spawning stock biomass and fishing mortality rates that fall within the 95% confidence interval that was used to generate the data. The results in Figures 5.1 and 5.1 indicate that this is in fact the case.

5.2 Mixed error

In a mixed error model, the model includes process errors in the form of deviations from average recruitment, and these are drawn from a normal distribution with a mean 0 and standard deviation equal to σ_R. When conducting simulations of this type, the model is conditioned on the instantaneous fishing mortality rates that are specified in the parameter input file (`ham.pin`) if the file exists in the current working directory. If a parameter input file does not exist, then the initial values specified in the control file are used for the leading model parameters and the annual instantaneous fishing mortality rates are set at a default value of 0.2 each year.

It's important to note here that the scaling parameters (initial recruitment and average recruitment) are sufficiently large such that the population is not driven to extinction based on the initial parameter values. Moreover, the current iteration of the simulation model generates recruits from an average recruitment and log-normal deviations. As such, the simulated data are not informative about the underlying stock-recruitment relationship. Additional modifications should be made to the simulation model, where the expected value of recruitment in each simulation year is a function of R_0 and the steepness (or slope at the origin) of the stock-recruitment relationship.

No mixed error simulations are shown in this technical document, but the same procedure as outlined in section 5.1 is used to conduct the simulation experiments.

6 Example Assessment: Sitka herring

For this example, the data from the Sitka herring stock were used in this assessment. This example is just that, an example. This is not intended to be used for any decision making purposes.

6.1 Input data

The catch data shown in Figure 3 are shown with the estimated 95% confidence intervals associated with the log standard error associated in the input data file. For this example the assessment model is conditioned on effort, which simply means a vector of fishing mortality rates are estimated by fitting the model to the observed catch. The previous assessment simply subtracted the catch, which implies the catch is known and assumed to have no measurement error.

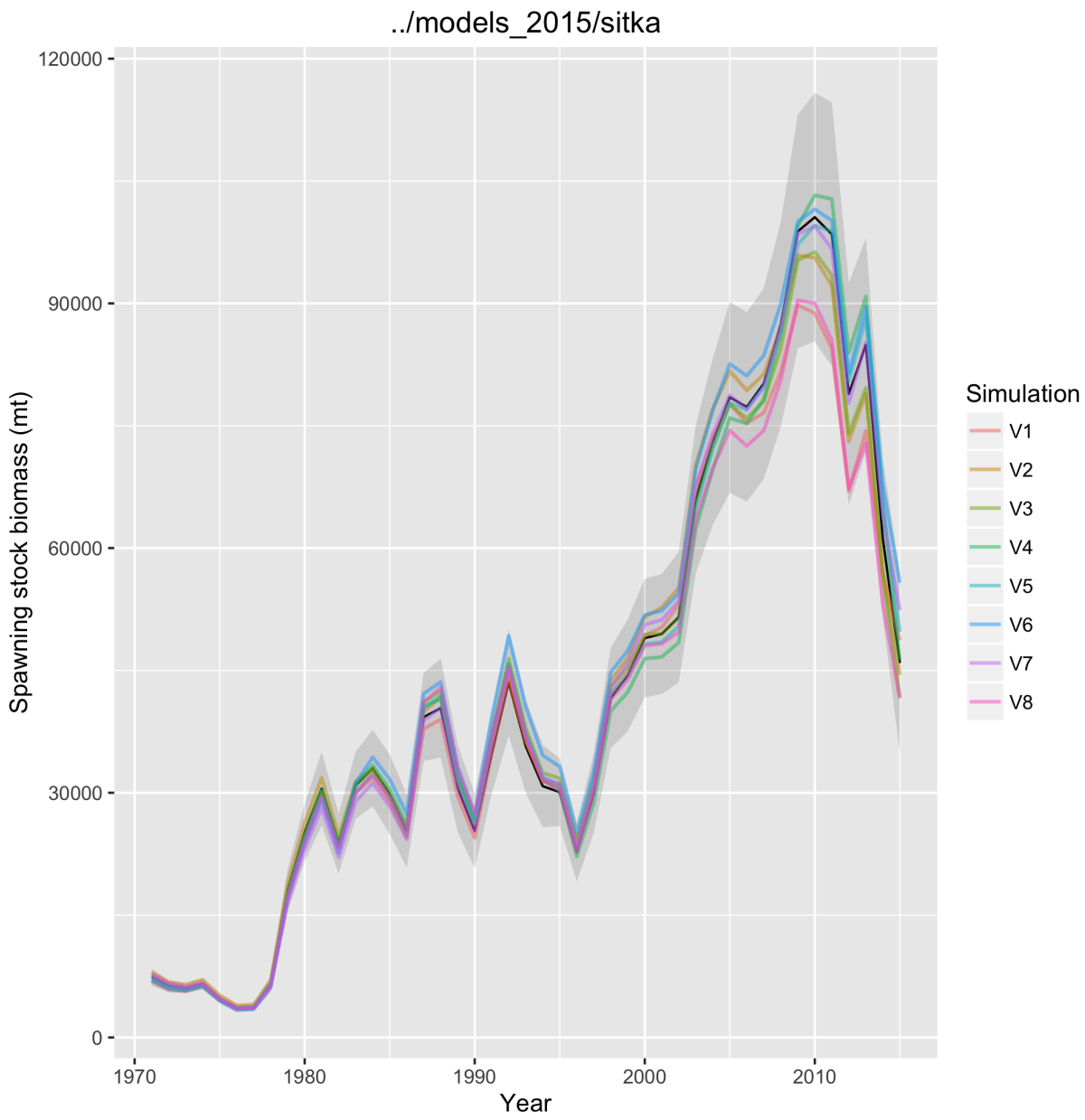

Figure 1: Estimates of spawning stock biomass where only the observation errors differ among simulations. Shaded region corresponds to the 95% credible interval for the true distribution of spawning biomass, and the colored lines correspond to estimates based on simulated observations in the egg survey and catch-age sampling.

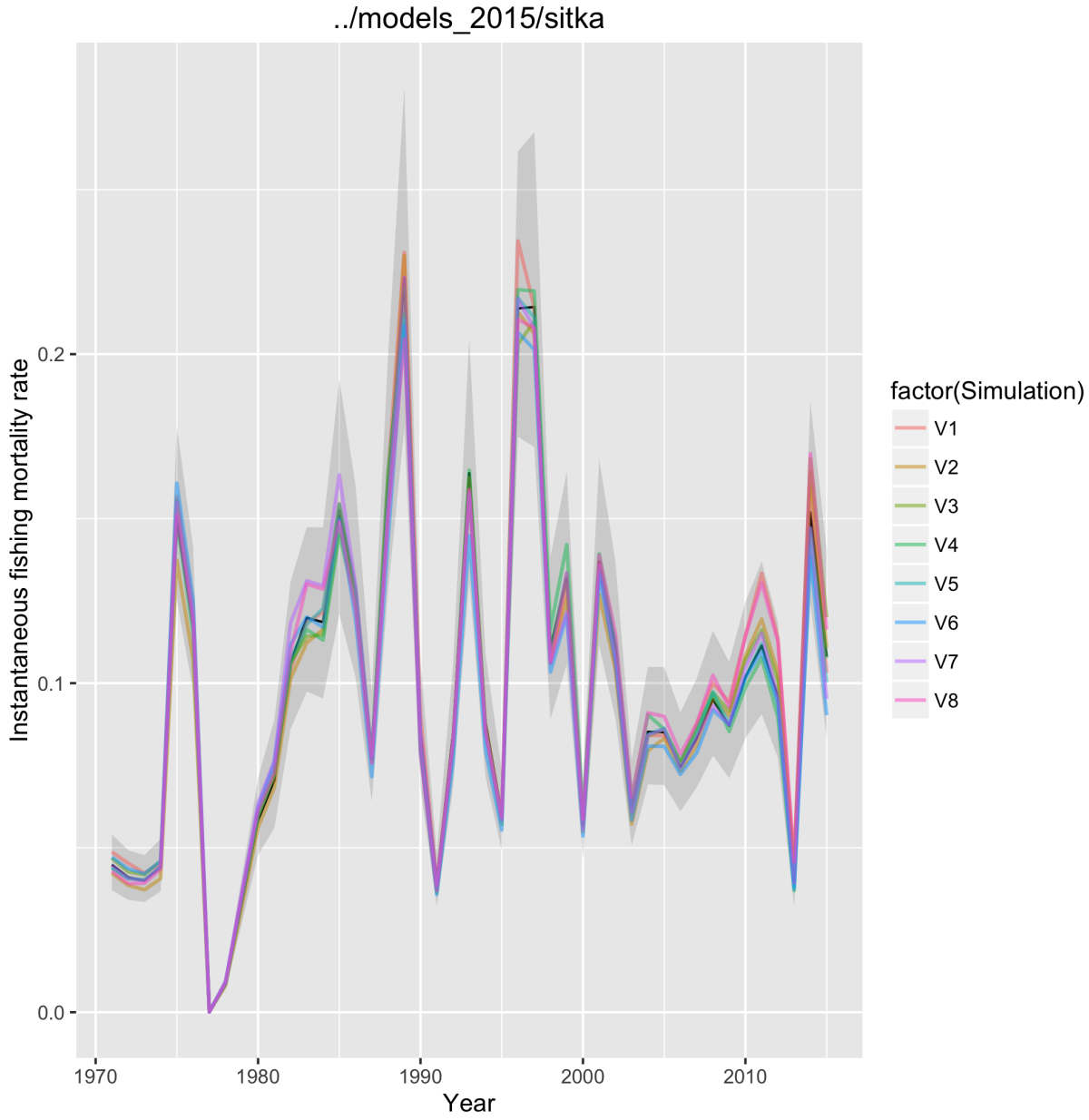

Figure 2: Estimates of the instantaneous fishing mortality rates where the only observation errors differ among simulations. Shaded region corresponds to the 95% credible interval for the true distribution of fishing mortality, and the colored lines correspond to estimates based on simulated observations in the egg survey and catch-age sampling.

The primary index for fitting this model is the survey egg deposition data. The time series for the Sitka herring stock is shown in Figure 4 and are plotted on a logarithmic scale. There has been nearly a 10-fold increase in this index between 1970's and the 2010's. Note that the model assumes a 50:50 sex ratio, and the data are scaled in trillions of eggs, assuming the catch is in metric tons.

The empirical weight-at-age data for the spawn survey are shown in Figure 5). The commercial weight-at-age data are shown in Figure 6. These are user input data and are used to convert numbers-at-age to weight-at-age.

Figures 7 and 8 are bubble plots showing the proportions-at-age in the age-composition data for the spawn survey samples and the commercial samples, respectively. Each distinct color represents a specific cohort over time, and the area of each circle is proportional to abundance.

Table 2: Notation and equations for population dynamics model.

Model parameters

$$\theta = \{\ln(M), \ln(\bar{R}), \ln(\ddot{R}), \ln(\alpha), \ln(\beta), \vec{\delta}\} \tag{T2.1}$$

Initial States ($i = 1980$)

$$\iota_j = \begin{cases} \exp(-M * (j - \min(j))) & \text{where } 3 \leq j \leq 7 \\ \dfrac{\exp(-M * (j - \min(j)))}{1 - \exp(-M)} & \text{where } j = 8 \end{cases} \quad \text{survivorship} \tag{T2.2}$$

$$N_{i,j} = \ddot{R} \exp(\delta_{i-j}) \iota_j \qquad i = 1980, \forall j \qquad \text{initial numbers-at-age} \tag{T2.3}$$

$$N_{i,j} = \bar{R} \exp(\delta_i) \qquad \forall i, j = 3 \qquad \text{age-3 recruits} \tag{T2.4}$$

Dynamic States ($i > 1980$)

$$Q_{i,j} = \frac{N_{i,j} S_{i,j}}{\sum_j N_{i,j} S_{i,j}} \qquad \text{vulnerable proportions} \tag{T2.5}$$

$$\bar{w}_i = \sum_j w_j Q_{i,j} \qquad \text{average weight of catch} \tag{T2.6}$$

$$C_{i,j} = \frac{\hat{c}_i Q_{i,j}}{\bar{w}_i} \qquad \text{where } \hat{c}_i \text{ is the observed catch (mt)} \qquad C_{i,j} \text{ catch-at-age} \tag{T2.7}$$

$$\acute{N}_{i,j} = N_{i,j} \exp(-F_{i,j}) \tag{T2.8}$$

$$N_{i+1,j+1} = \acute{N}_{i,j} \exp(-M_{i,j}) \tag{T2.9}$$

Spawning stock biomass

$$B_i = \sum_j (N_{ij} - C_{ij}) \omega_{ij} w_j \qquad \text{Spawning stock biomass} \tag{T2.10}$$

Stock-recruitment

Table 3: Data and types of likelihoods implemented

Data	normal	log-normal	multivariate-logistic	multinomial
Commercial Age-comps			X	
Spawn Survey Age-comps			X	
Egg deposition		X		
Aerial survey		X		
Catch		X		
Stock-Recruitment		X		

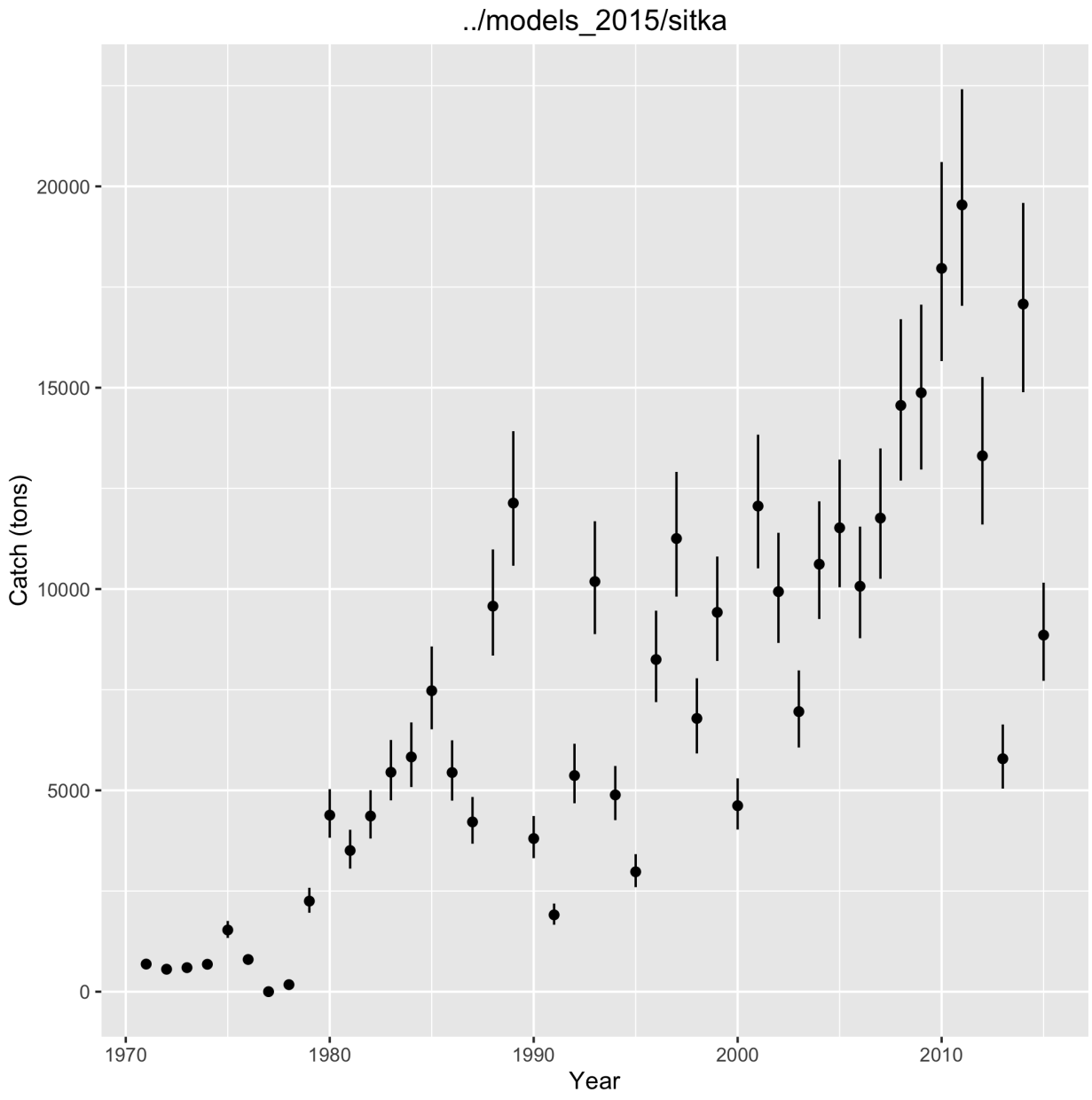

Figure 3: Herring removals from the Sitka stock. Error bars are based on the log standard error defined in the input data file.

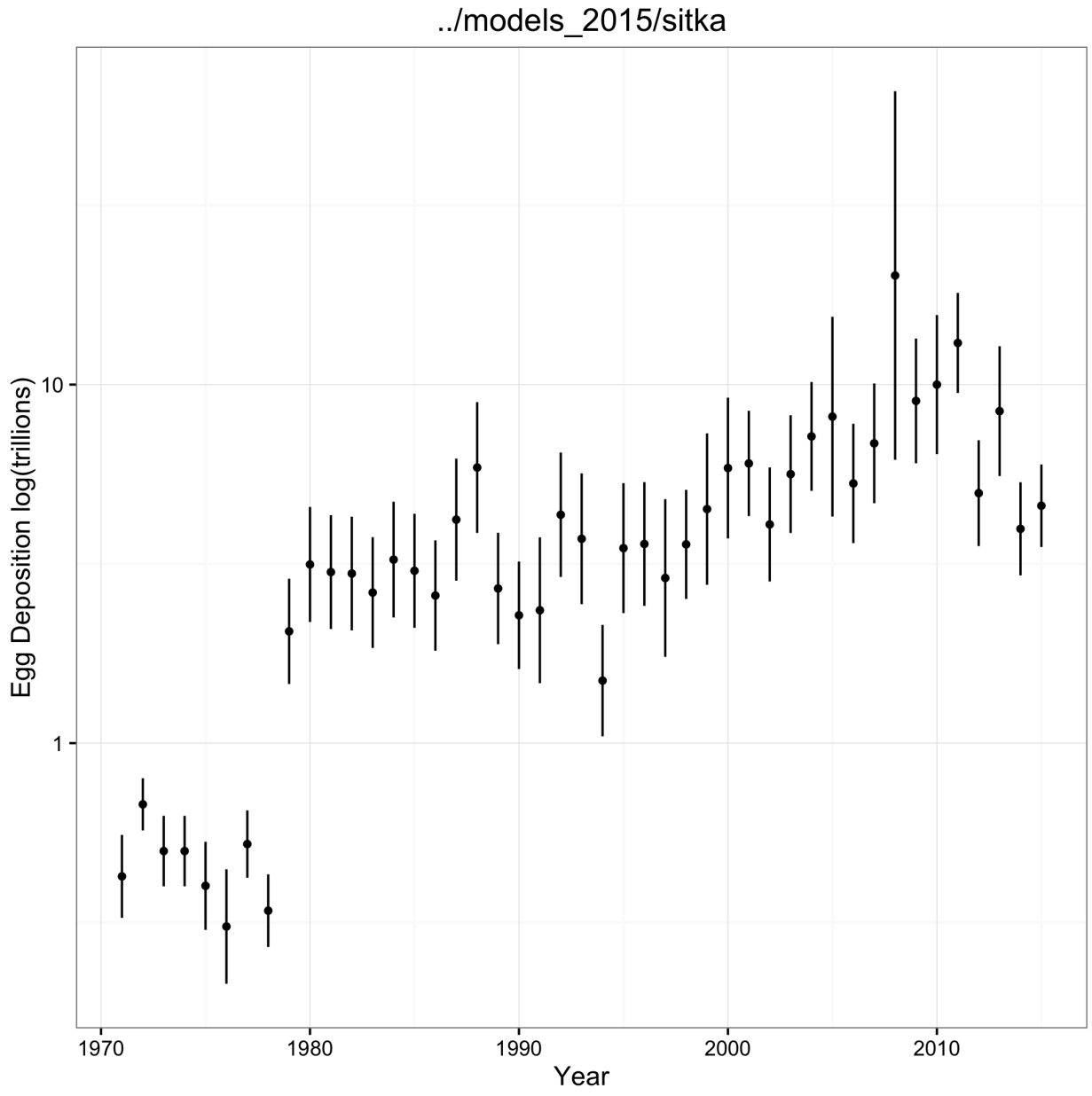

Figure 4: Egg survey index for Sitka herring. Note that these data are plotted on a log scale.

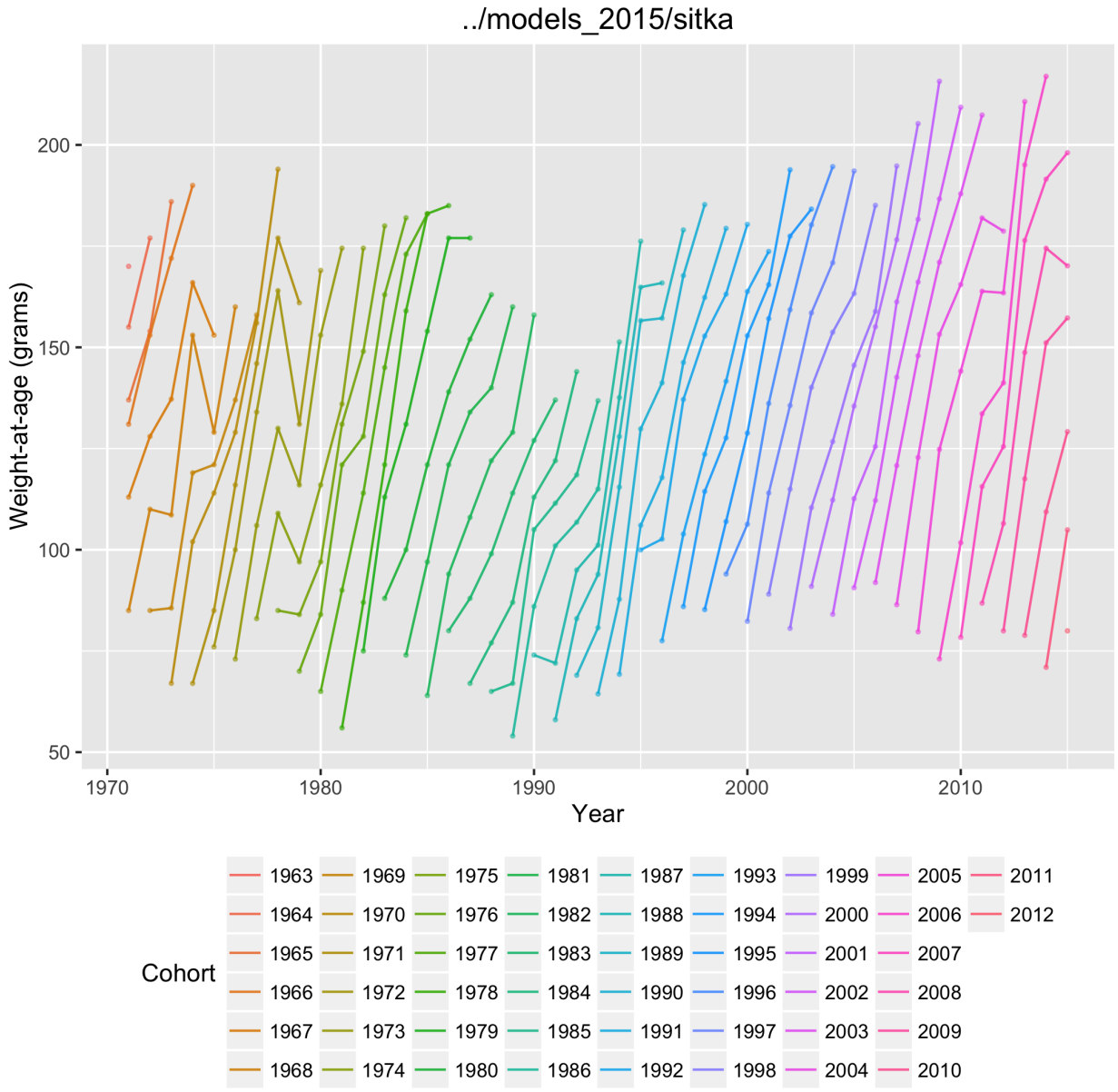

Figure 5: Empirical weight-at-age data for Sitka spawn survey.

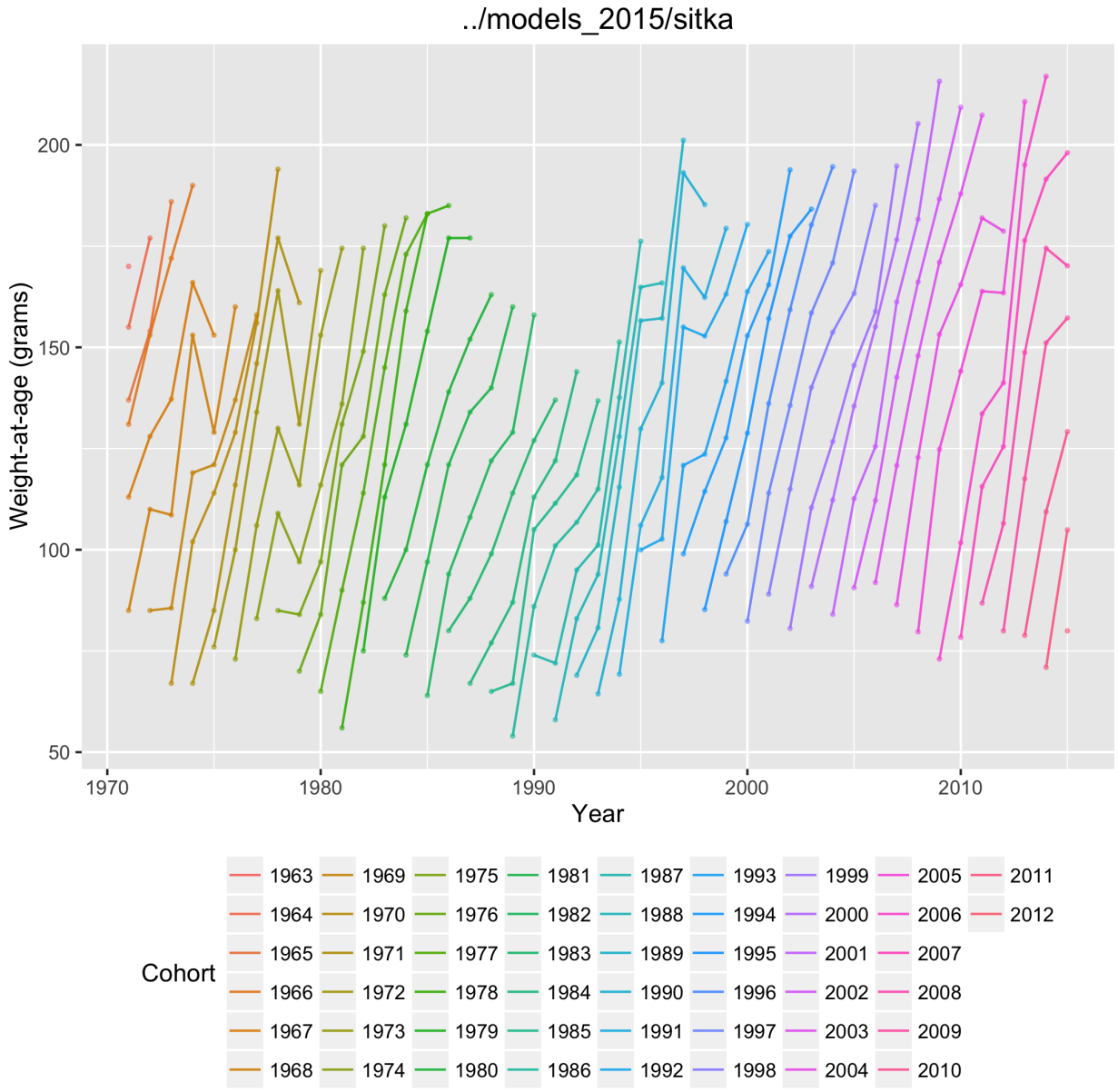

Figure 6: Empirical weight-at-age data for Sitka Commercial fishery samples.

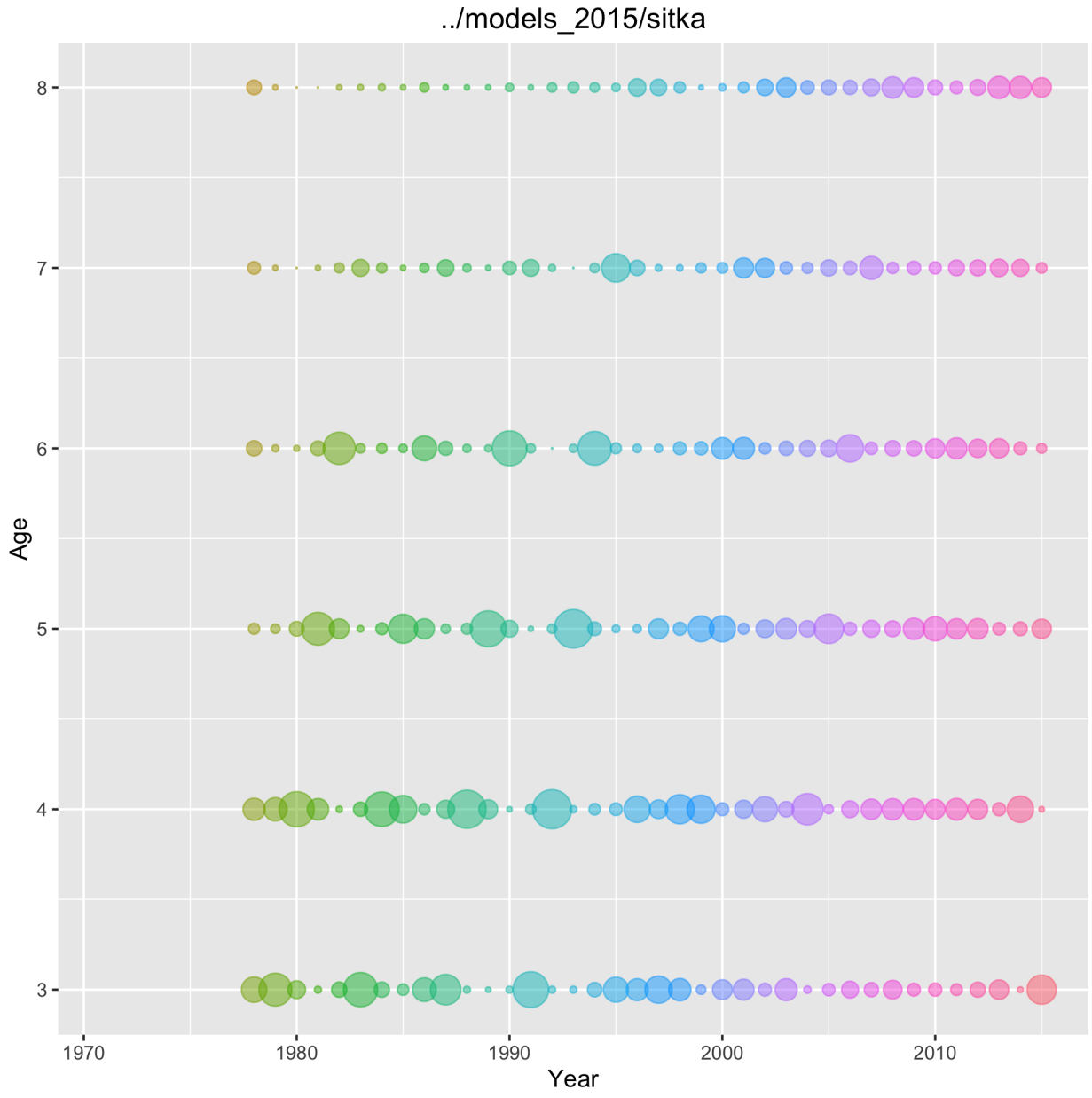

Figure 7: Age-proportions by year from the spawn survey samples.

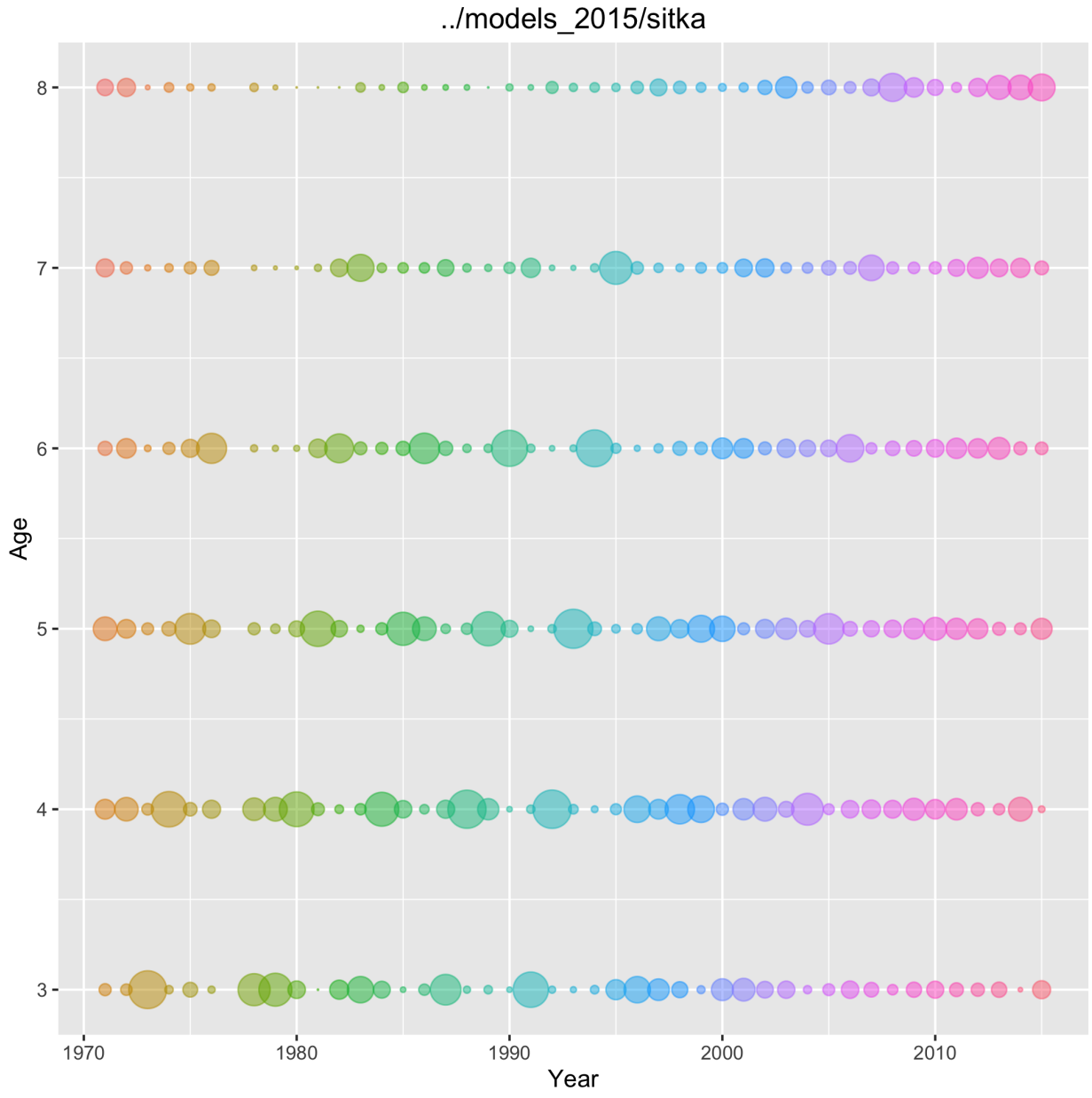

Figure 8: Age proportions by year from the commercial fishery samples.

6.2 Model outputs

Model output from the age-structured herring model are stored in a number of output files that are produced automatically by ADMB. The maximum likelihood estimates (MLE) of model parameters with standard deviations and correlations are found in the `ham.cor` file. User defined outputs are in the `ham.rep` file, and if you wish to add additional outputs that currently do not exist, these outputs can be added to the `REPORT_SECTION` of the tpl file and the code must be recompiled.

The following output figures were generated from a series of R-scripts (available on the project repository) that were developed during the course of model development.

Estimates of mature spawning stock biomass (male and female combined based on the maturity ogive) are shown in Figure 9, along with the approximate 95% confidence interval.

Estimates of the average annual instantaneous fishing mortality rates each year are shown in Figure 10. Recall that age-specific selectivity each year is scaled to have a mean value of 1.0 to ensure both parametric and non-parametric selectivity models remain continuous and differentiable. If asymptotic estimates of fishing mortality rates are desired, then the series shown in Fig. 10 is multiplied by the maximum selectivity each year.

Figure 11 compares the trends in the asymptotic fishing mortality rates versus the average fishing mortality rate. The trends in the asymptotic estimates of fishing mortality suggests that fishing mortality rates have been increasing in recent years, but this trend is associated with changes in selectivity, where younger age-classes are becoming less vulnerable to the gear. To examine this issue further you would look for changes in selectivity over time where older age-classes are more selected that younger age-classes.

Again, in this example the stock assessment model is conditioned on fishing effort. This means that a vector of annual fishing mortality rates are estimated by jointly fitting the model to the observed catch data (Figure 12). This differs significantly from the previous model version where the observed catch was assumed to be measured without error, and removed from the population using a difference equation.

The residual difference between the observed and predicted catch is shown in Figure 13. In this case the residuals appear to have a non-random pattern that emerges due to the minor differences between the trends in F associated with the catch and abundance index, and the trends in Z that are inferred from the age-composition data. Furthermore, the pattern changes from random from 1970-1979, to all negative from 1980 to 1999, and flips to positive from 2000 to 2015. These blocks also correspond to the selectivity blocks in the control file for the Sitka herring stock.

The primary information that the model is being fit to is the egg deposition index (Figure 14). The egg deposition survey data are treated as absolute abundance information. In other words there is no additional scaling parameter that is estimated for the purposes of comparing only trend information. These data provide information about population scaling, so units associated with catch, weight-at-age, and maturity, are critical.

The residuals for the egg deposition pattern are shown in Figure 15. The model is not able to fit the 1988, 1994, and 2008 survey data points (corresponding to the largest 3 residuals).

Another feature built into the assessment model is to jointly fit a stock-recruitment

model to the estimated age-3 recruits and estimated spawning biomass. This could also be done outside the model, but the resulting estimates of uncertainty in reference points would be biased because uncertainty in the independent variable (spawning biomass) is not propagated. The residual fit to a Ricker stock-recruitment curve is shown in Figure 16.

Residual fits to the catch-at-age data for the spawn survey samples are shown in Figure 17. The area of each circle is proportional to the residual difference between the observed catch-at-age proportion and the predicted catch-at-age proportion. The residual patterns for the commercial age-composition proportions is shown in Figure 18. Ideally, the pattern of residuals would be completely random with respect to both age and time dimensions. Some patterns to watch out for that could be a sign of model-misspecification are blocks of residuals all of the same sign (+ve or -ve) that might be indicative of a change in behavior or a change in regulations that result in a behavioral change.

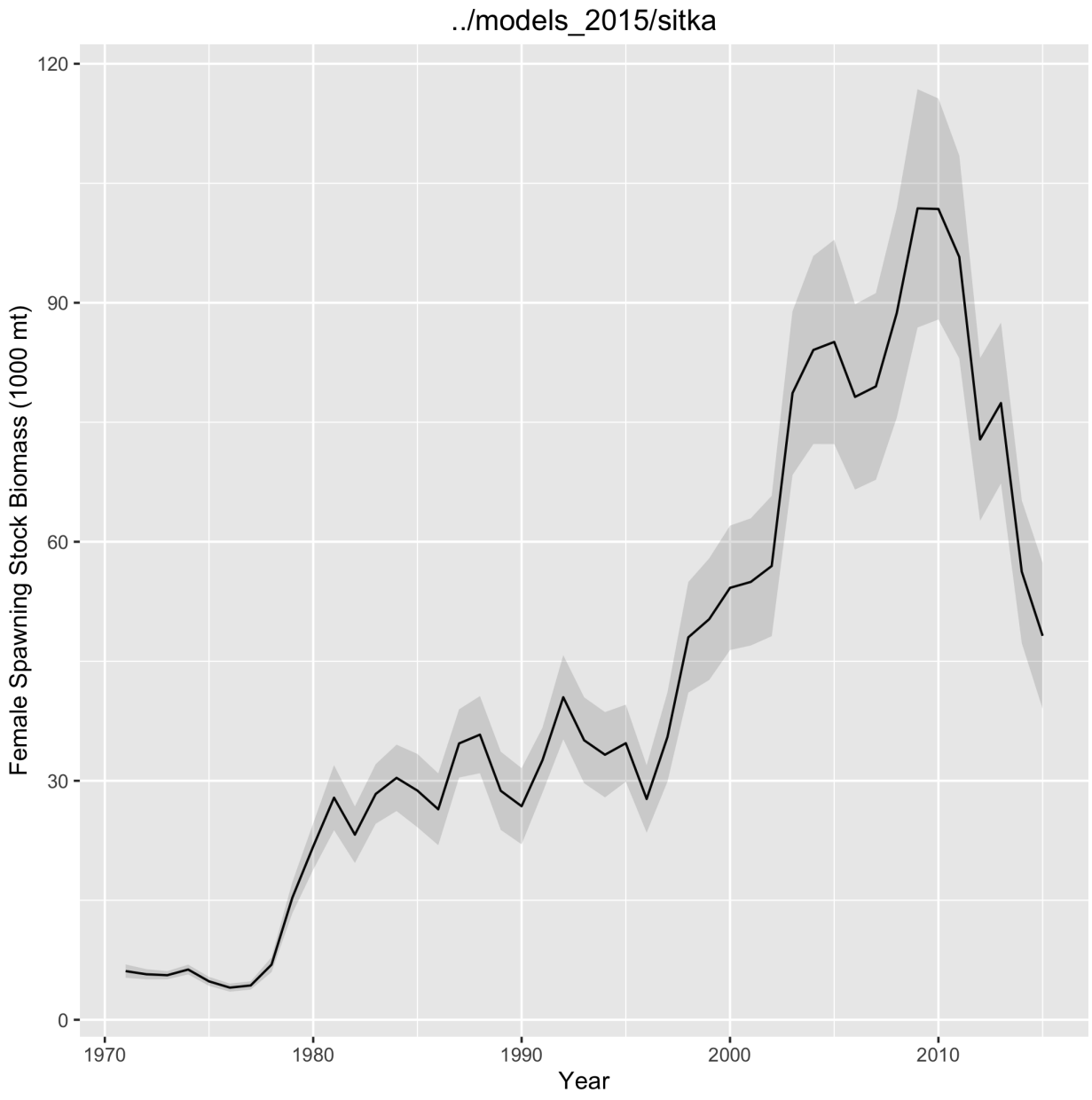

Figure 9: Estimates of mature spawning biomass at the time of spawning, (post-fishery) and the 95% confidence interval shown in the shaded region.

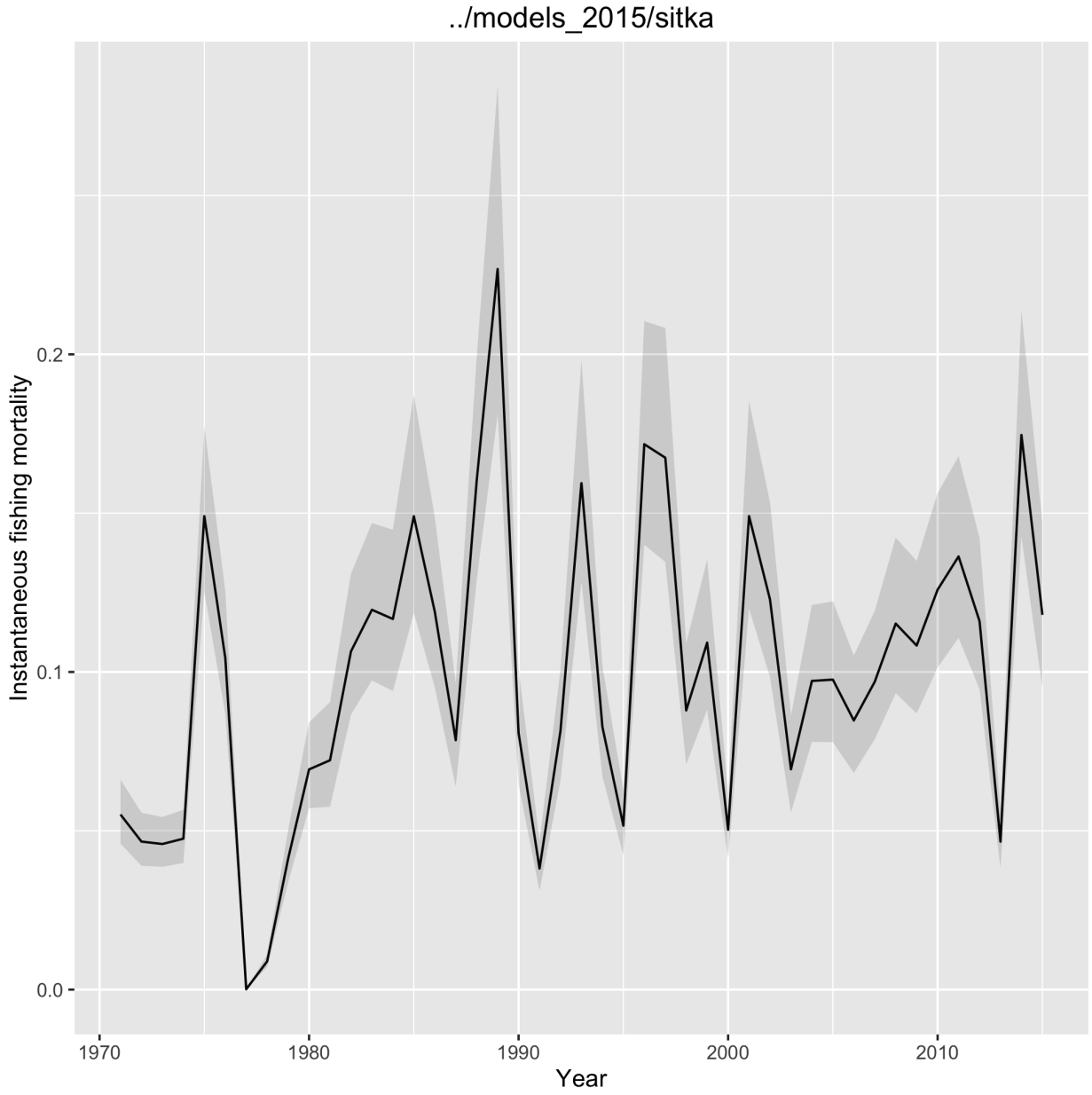

Figure 10: Estimates of the mean instantaneous fishing mortality rate with the 95% confidence interval.

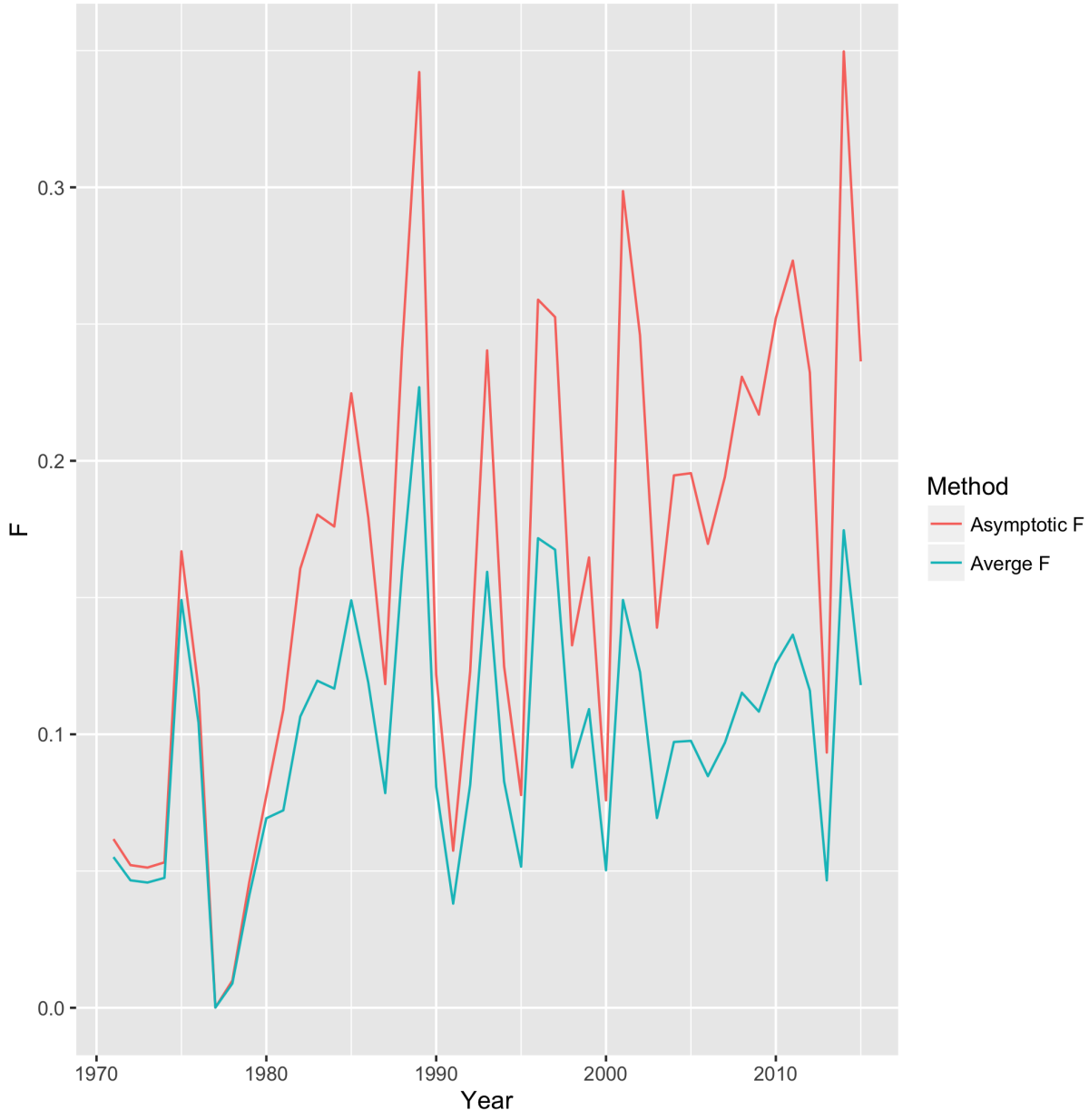

Figure 11: Trends in average age-specific fishing mortality rates versus the asymptotic trends in fishing mortality rate. The trends differ slightly due to changes in selectivity over time.

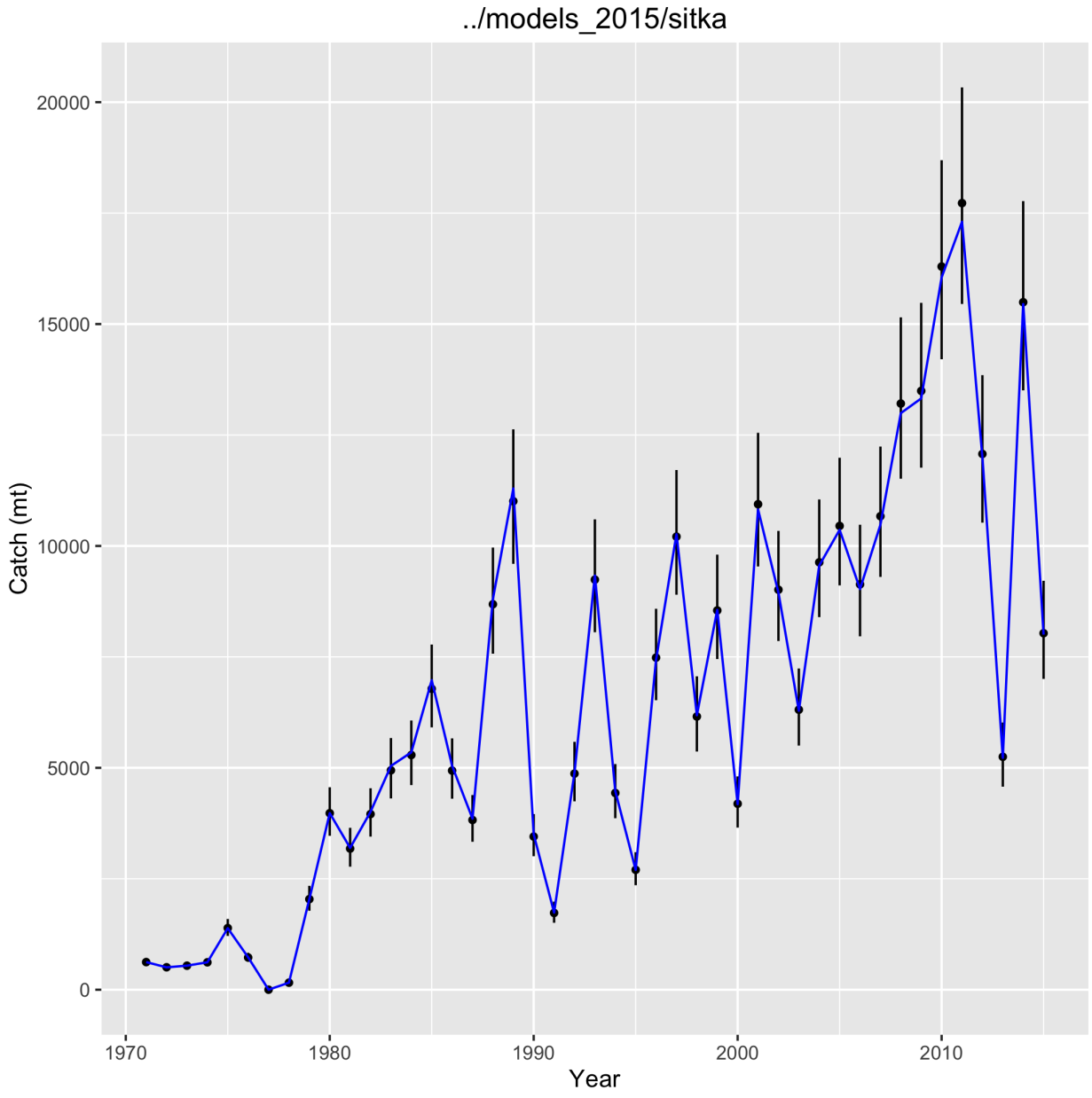

Figure 12: Observed and predicted catch in the Sitka herring fishery.

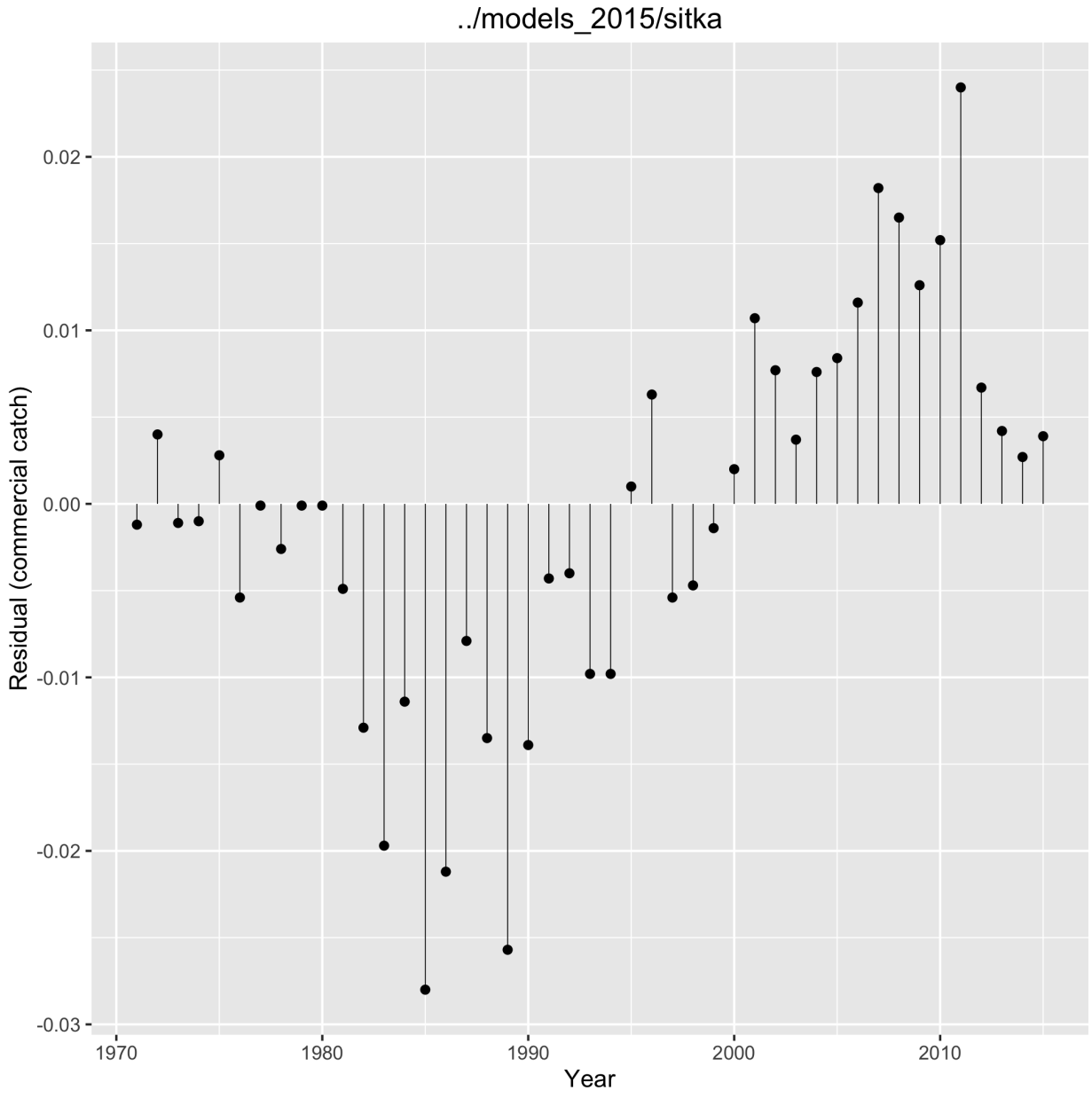

Figure 13: Residual fit to the commercial catch data.

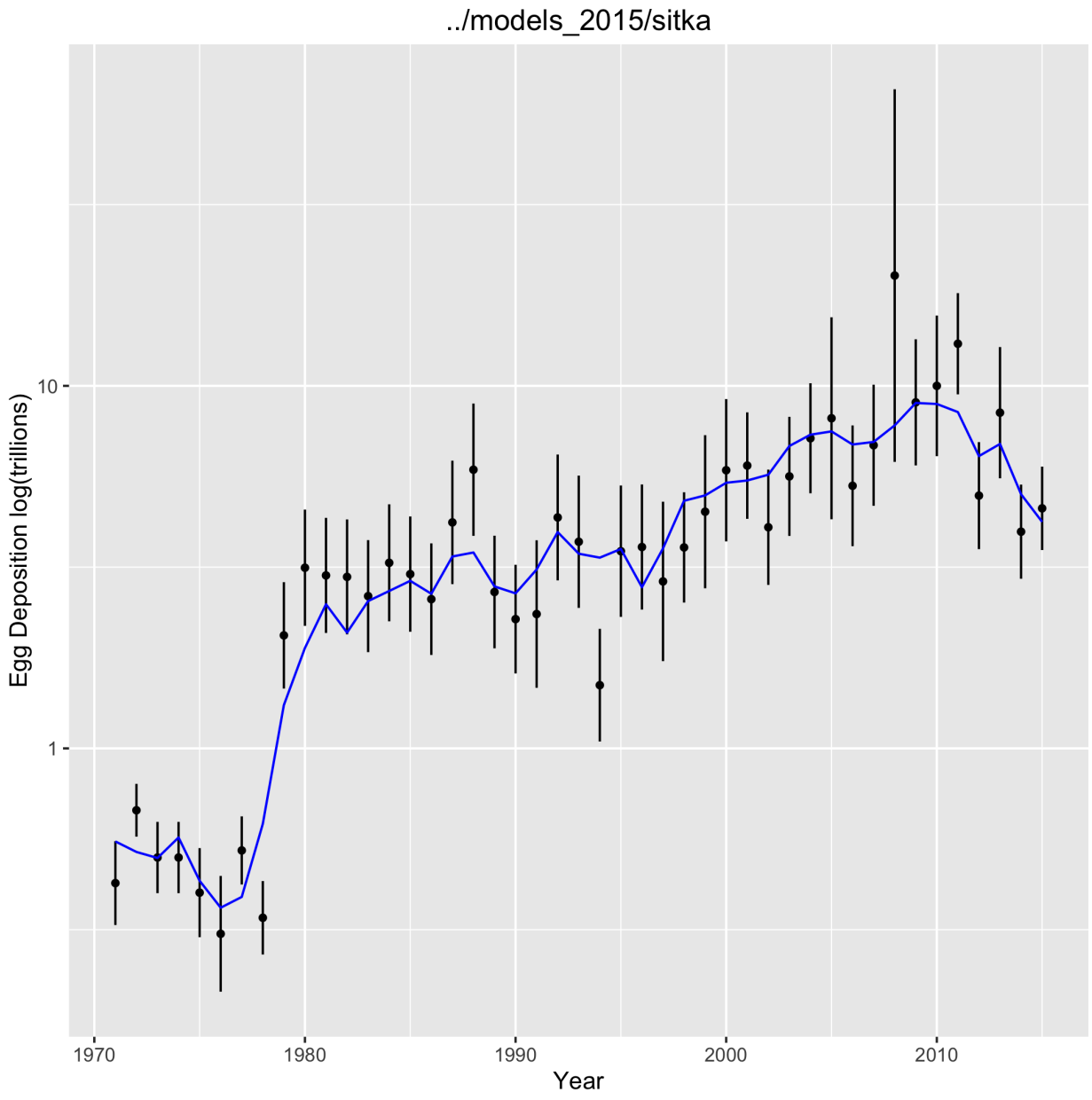

Figure 14: Fits to the egg survey index for Sitka herring. Note these data and predictions are plotted on a log scale.

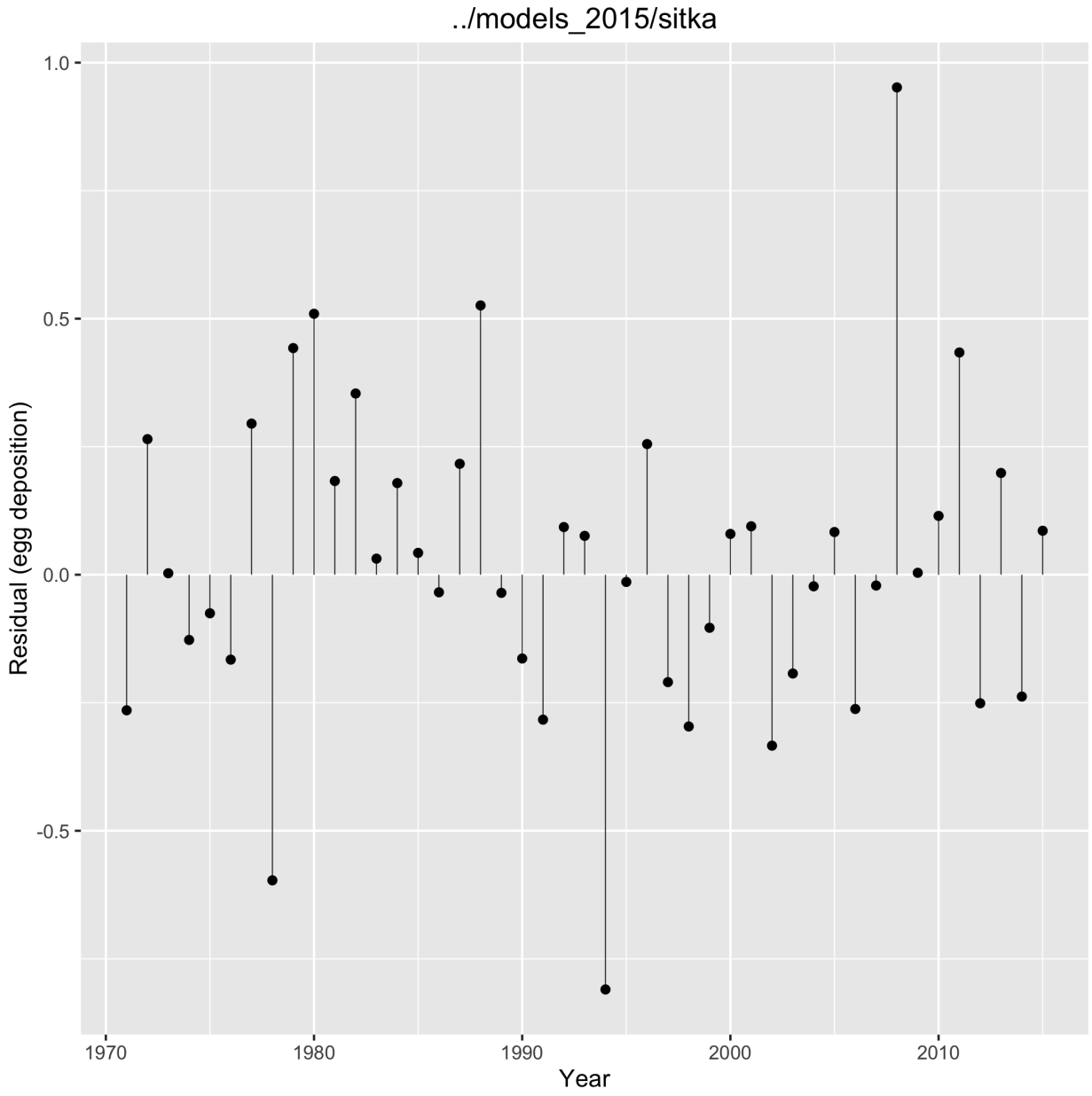

Figure 15: Residual fit to the egg deposition data.

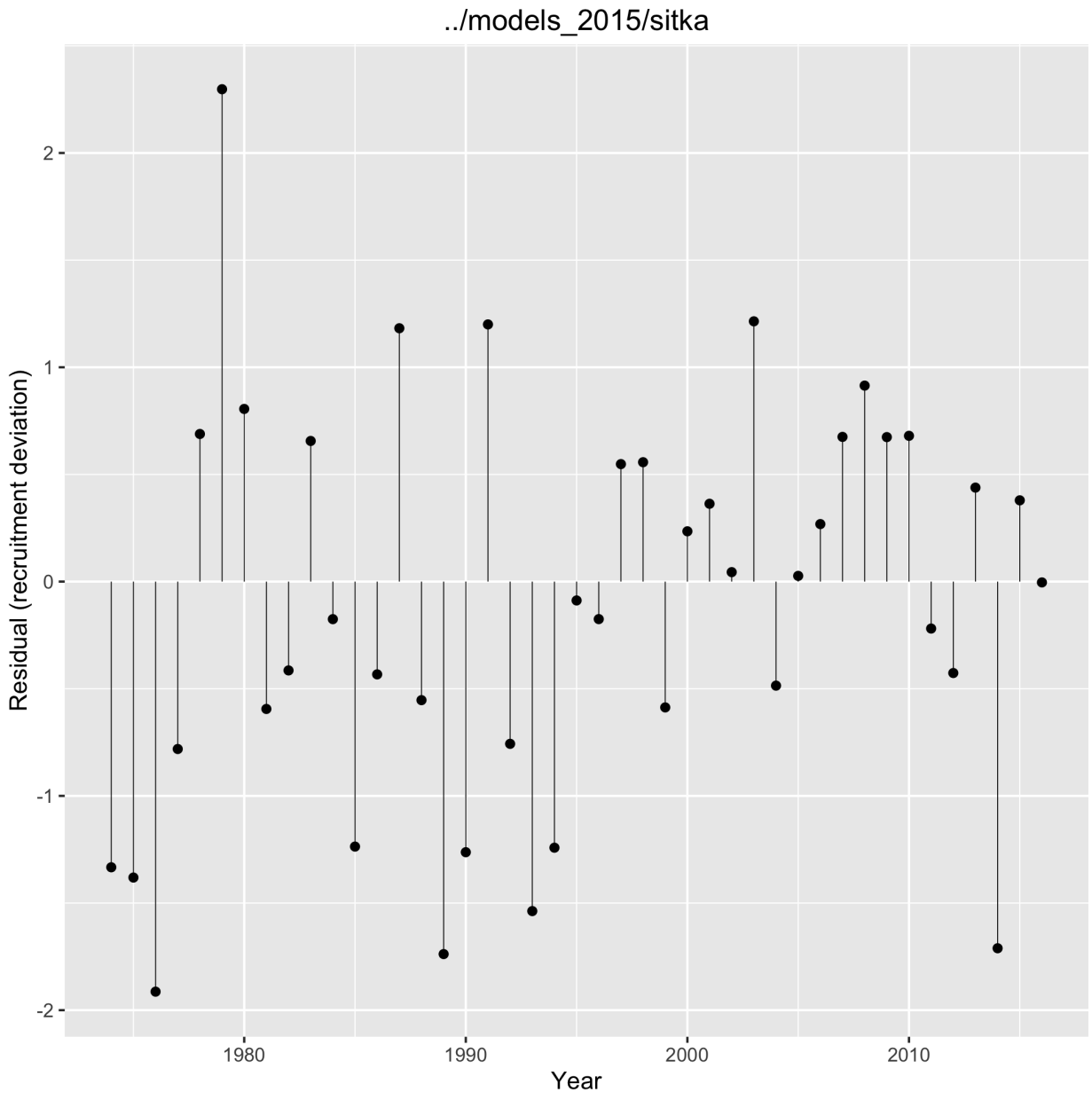

Figure 16: Residual deviations between the log of annual age-3 recruits and the Ricker stock recruitment relationship.

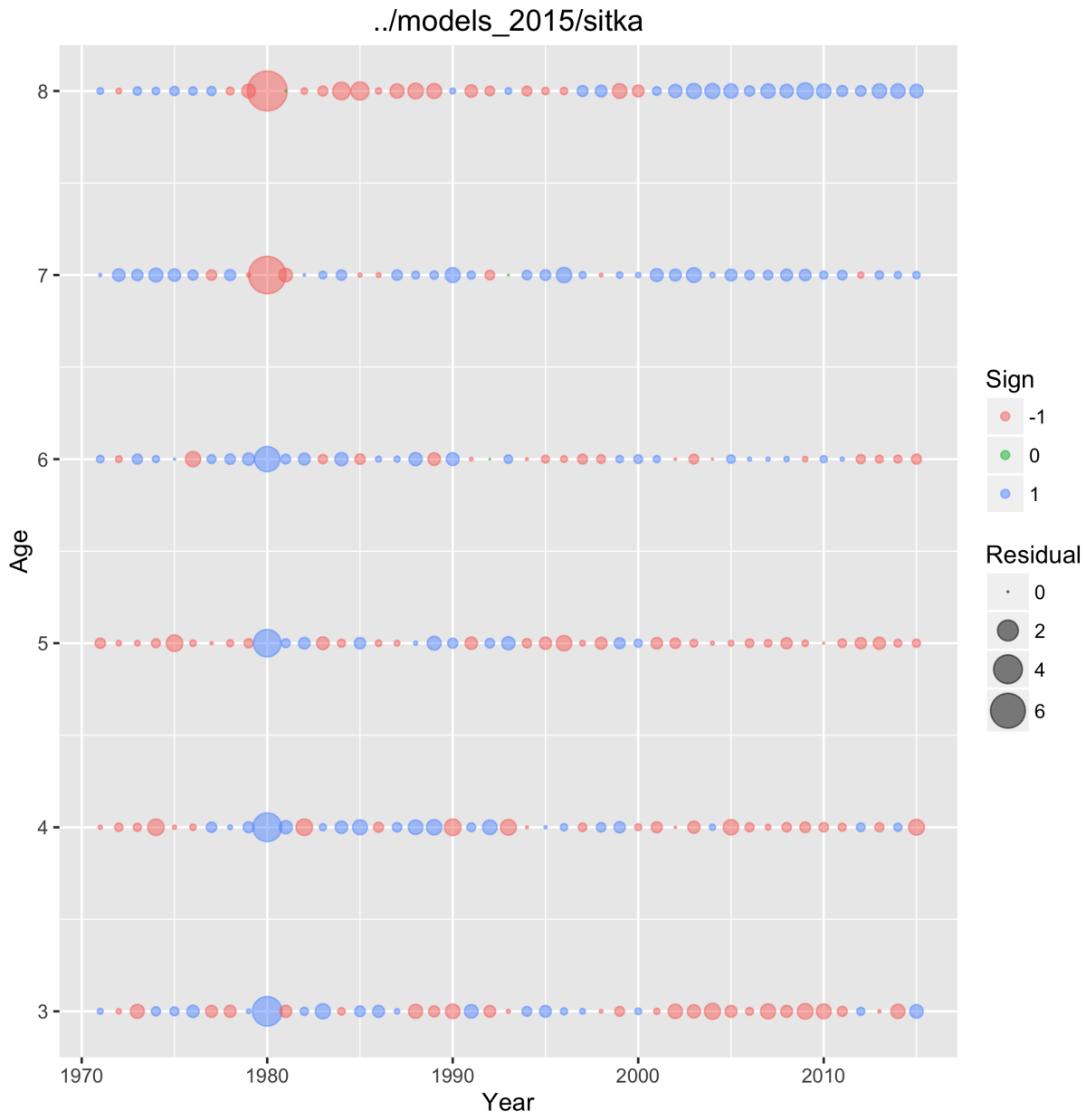

Figure 17: Residual fits to the spawn survey age composition data.

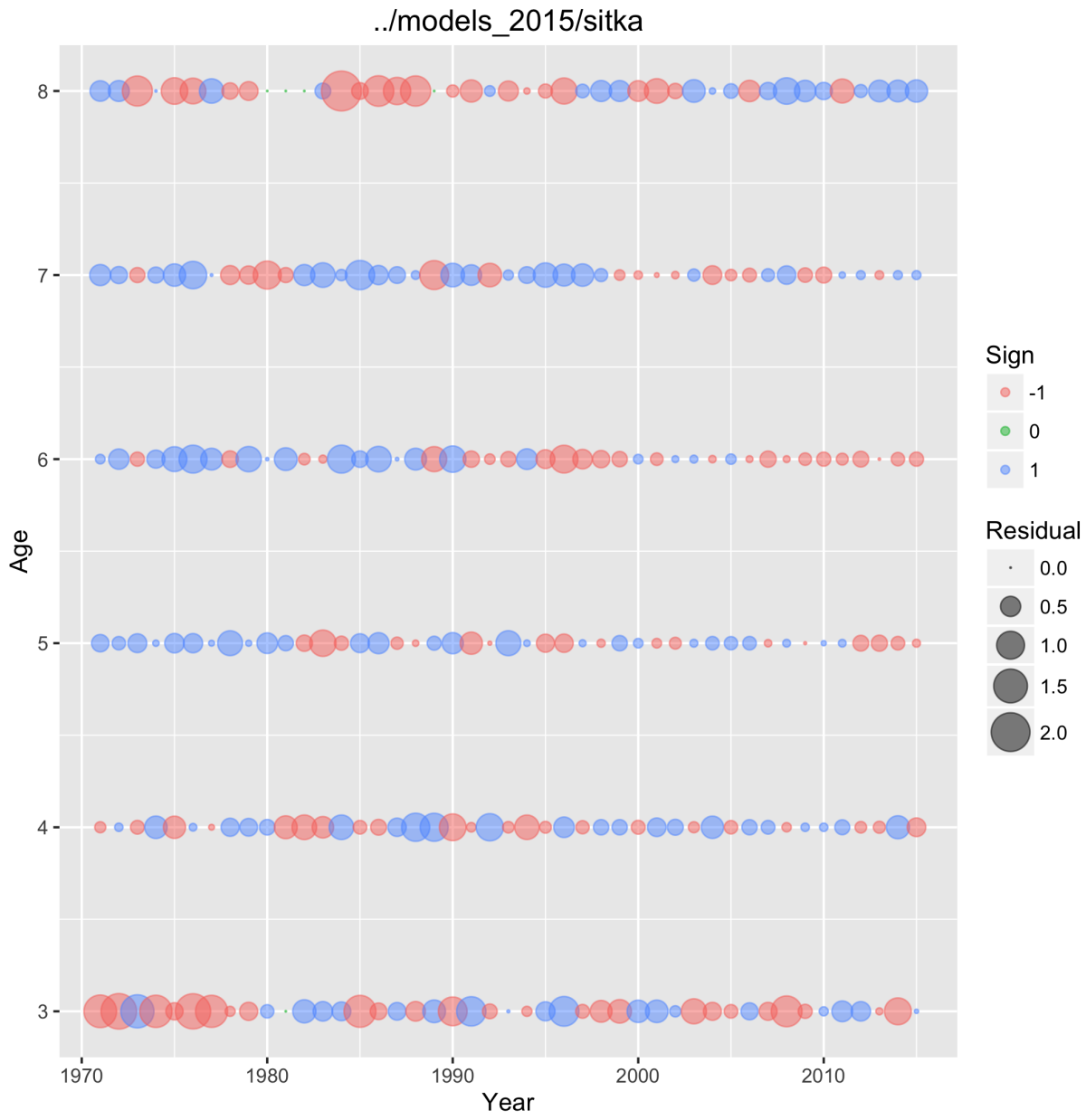

Figure 18: Residual fits to the commercial age composition data.

7 Summary

Although age-structured models tend to estimate dozens, if not 100s, of more parameters than the simple surplus production or biomass dynamics models, The key policy parameters that define stock productivity and scale (i.e., intrinsic rate of growth and carrying capacity) only involve 3 basic parameters, unfished stock size, natural mortality, and the steepness of the stock-recruitment relationship. Estimates of unfished biomass, or its analogue unfished recruitment, defines the population scaling. This parameter is primarily informed by the scale of the catch data. The natural mortality rate defines "residency", or the number of years and average individual will persist in the population and survive to contribute to future generations via spawning events. Trend information in composition data can jointly inform natural mortality rates in a stock assessment model, but these estimates are also conditional on structural assumptions about model selectivity. The steepness of the stock recruitment relationship is more related to population resilience, but in this case we are specifically referring to how strong the density-dependent juvenile survival rate from egg to age-3 recruit is. The stronger the compensatory response is, the more resilient the stock is to the effects of fishing.

If reliable estimates of unfished stock size, natural mortality rates, and the steepness of the stock recruitment relationship can be obtained from the age-structured assessment model, then fisheries reference points can easily be developed conditional on assumptions about fisheries selectivity. The code herein readily provides the means to calculate MSY or SPR-based reference points. Furthermore, uncertainty in these reference points can also be quantified by either sampling from the joint posterior distribution and computing a distribution for MSY, or FMSY. Or use the delta method to obtain asymptotic estimates of uncertainty using the inverse of the Hessian matrix (e.g., an `sdreport_number` in ADMB).

A number of significant changes were introduced into this new code, with the primary goals of: improving numerical stability, providing a more modern statistical framework for quantifying uncertainty, and changes to the data input and control files to allow for rapid exploration of alternative model structures without having to recompile the code, or have several versions of the code, that are all prone to programmer errors. I've also tried to provide comments in the code that direct a programmer or the next generation of analyst to add additional options for selectivity, or different stock-recruitment curves and, alternative likelihood functions for composition data. This is an active area of research in fisheries stock assessment, and the documented code provides an interface in which to explore alternatives.

As an example of flexibility, let's say a reviewer wanted to know how the assessment model differs if you were to treat the egg survey index as a relative index instead of absolute. One option that does not involve making any changes to the code, is to put the egg index in the mile milt day input. In this case, the model will only fit the trends in the egg abundance rather than treating them as absolute. More importantly, no potentially dangerous code changes were necessary to make the comparisons.

Below are some of the major structural differences and a few warnings to the user to watch out for. For example: is the objective function continuous and differentiable, how to avoid getting stuck in local minima, is fishing mortality really increasing or is selectivity just

changing?

The previous age-structured model for Alaska herring stocks assumed the catch was known without error. In this parameterization each year the observed catch was subtracted from the mature spawning biomass using a difference equation. One potential pitfall with this approach is that during the non-linear optimization to find the maximum likelihood estimates of the model parameters, it is possible that the population can go negative. In such circumstances, the search routine can easily get stuck in local minima because the gradient of the objective function is not continuous. My recommendation is not to condition the model on catch, but to fit the model to the catch and estimate the fishing mortality rates directly.

When allowing estimated model parameters to vary over time (e.g., time-varying selectivity, or time-varying natural mortality rates), estimated trends in the asymptotic fishing mortality rates may differ slightly with trends in the average fishing mortality. The fishing mortality rates may appear to be stable, but this statement is only true if trends in selectivity are also invariant over the same time period.

Lastly, if you feel confident that the data are informative such that you would like to try and estimate the variance parameter for the recruitment deviations (σ_R), you'll likely discover that the model may tend to converge to either an observation error only model (i.e., $\sigma_R \to 0$), or less likely a process error only model (only if the user specifies very small observation errors in the data file). There are a number of options that might be considered to address this statistical "errors-in-variables" problem. The simplest approach is jointly estimate an additional variance term (a feature commonly implemented in Stock Synthesis), or preferably integrate over the random variables (recruitment deviations) using numerical methods (e.g., MCMC). For example, a common observation is that MLE estimates of σ_R in this model will be less than the median estimates of σ_R obtained from random samples of the joint posterior distribution. If the data are not that informative, or there is a lot of conflicting data or model misspecification that leads to greater uncertainty, informative priors for σ_R will probably be required to obtain convergence. The alternative to MCMC is a mixed-effects, or random-effects, models which are becoming more popular in the last 5 years.

References

Francis, R. (2011). Data weighting in statistical fisheries stock assessment models. *Canadian Journal of Fisheries and Aquatic Sciences*, 68(6):1124–1138.

Mace, P. M. and Doonan, I. (1988). *A generalised bioeconomic simulation model for fish population dynamics*. MAFFish, NZ Ministry of Agriculture and Fisheries.

Schnute, J. and Richards, L. (1995). The influence of error on population estimates from catch-age models. *Canadian Journal of Fisheries and Aquatic Sciences*, 52(10):2063–2077.

Walters, C. and Ludwig, D. (1994). Calculation of Bayes posterior probability distributions

for key population parameters. *Canadian Journal of Fisheries and Aquatic Sciences*, 51(3):713–722.