HCS program for simulating harvest control rules. Program description and instructions for users

Version HCS13_2
Manual revised March 2013

Dankert W. Skagen
www.dwsk.net

1. Overview

The ***hcs*** program ('harvest control simulation') is a general purpose program for stochastic simulation of management decision rules (harvest control rules). Several versions have been distributed over time, this manual is for the version ***hcs13_2.***

The ***hcs*** program is a single stock, single fleet, age disaggregated tool for simulating harvest control rules. It can simulate a quite wide range of rules, and has numerous options to parametrise the rules. The underlying population model (operating model) has a wide range of options for recruitment, included regime shifts, and for variable and density dependent weights and maturities. Deriving data for decisions from the operating model is done with an algorithm that imitates the effect of some sources of errors in analytic stock assessments, but there is no full assessment in the loop. The program is run as a bootstrap with randomly drawn stochastic terms in each realization, and the distributions of results are derived as frequency distributions in the set of realizations.

hcs13_2 can scan automatically over ranges of harvest rule options and parameters, as well as some stochastic distribution parameters. The results are partly presented as annual means and percentiles, partly as detailed data for generating distributions, but also as assembly tables allowing performance measures to be compared across options. Since there is no full assessment in the loop, the program is fast - each scenario takes 10-30 seconds with 1000 iterations. This should make it a versatile tool for searching for optimal design of rules and for sensitivity testing.

The ***hcs*** program originally was developed as an experimental tool, to study generic properties of harvest control rules, but has been adapted to assist in the design and evaluation of harvest control rules for real stocks. It has options for generating input data either according to basic biological dynamics or by using data observed for real stocks, and it can generate initial stock numbers by 'priming' the stock with a fixed fishing mortality as an alternative to taking such data as input.

Some version history
The core of the program was developed for a generic study of fixed quota regimes in 2006[1]. Since then, it has been continuously extended and adapted, and has gradually developed into a general simulation tool. Since 2010, versions are numbered by year and version within the year. Hence version 13_1 is the first revision in 2013. The

1) Skagen , D.W. (2007)
Management strategies for reducing variation in annual yield: when can they work?
ICES Journal of Marine Science. 64: 698-701

versions are to some extent backwards compatible, but not fully. Input files for previous versions may have to be revised to use the latest version.

In the early development, *ad hoc* versions, were made for the study of specific harvest rules. For simulating proposed rules for mackerel in spring 2008, a version named ***hcm***. with a special recruitment algorithm, was made. For harvest rules for Blue whiting (2008), a gradual reduction of a target F was asked for, as well as a different rule for applying a constraint on TAC variation. A version with these features had the acronym ***hcs_bw***. A later revision (***hcs_ad***; August 2009) has several extensions, in particular has options for periodic and/or spasmodic recruitment, and some additional printouts of age distributions. A revised version ***hcs10*** from 2010 kept most of these extensions, and in addition included new options for the transition phase when introducing a harvest rule, and for including density dependent weights and maturities. Parts of the code was rewritten to make it better structured. The version called ***hcs10_2*** had two additional features: A more flexible growth model, which also includes stochastic variation, and harvest rules with a plateau level of the target at intermediate SSBs. The input files were redesigned and the output has been slightly amended. Version ***hcs10_3*** had two new features: An option to draw occasional large year classes as successes in a Bernoulli trial, and an improved specification of noise in weight and maturity. Version ***hcs11_1*** in addition had the option to switch between recruitment regimes, either randomly or in a predetermined way. Version ***hcs11_2*** had several bugs fixed, but was otherwise similar to 11_1. In version ***hcs12_1*** the decision model was restructured, to allow more structured designs of harvest rules. The following versions had more harvest rules, some more flexible output and several bugs fixed. New in version ***13_2*** is additional harvest rules, moore options for the recruitment in projections, and a revised definition of risk, in accordance with revised recommendations from ICES (WKGMSE 2013).

As a rule, input files are ***not*** backwards compatible. This is because new revisions have required additional input to cover additional options and features.

Modifying the program
For each new stock or management plan, some modification of the program is almost unavoidable. Some options that are hard coded for convenience (number of iterations, number of years, number of ages) can be changed very easily in the include file that is part of the program code. The program is quite modular, and some modifications (but not all) should be relatively easy to write. Introducing new rules is relatively simple; however, a number of printout specifications will also have to be modified which is less trivial. Modifying the program requires some insight in Fortran77, and that a compiler is available. The latter should be a minor problem nowadays, good Fortran compilers both for Unix/Linux and Windows (and probably for Mac, but there I have no experience) can be downloaded free on the internet (see e.g. http://gcc.gnu.org/wiki/GFortran) .

Input-output
All input and output is through ascii-files, with hard coded file names. There are two standard input files called ***bio.inn*** and ***opt.inn***, and two optional input files (***canum.inn*** and ***nin.xxx*** - the actual suffix indicates the format - see the input section. There is a wide range of output options. Which files to produce is specified in the ***opt.inn*** file.

Distribution
The current distribution of ***hcs13_2*** is distributed as a zip-file, which contains the code, this manual, an .exe file for running under Windows and a set of input files that can serve as a template. For running under Linux, the program has to be compiled which can be done with any Fortran77 compiler. I have used the ***gfortran*** compiler from GNU for both Linux and Windows. It . Running under some variants of Windows requires the presence of a file *libgcc_s_dw2-1.dll* which is included in the distribution of the program. The zip-file can be downloaded from my home page www.dwsk.net, or by contacting me on dankert@dwsk.net.

The program is distributed according to the open source principles. Hence it is distributed free of charge, the code is distributed with the program and it is free to use for everyone. It has been tested quite extensively, but there is no guarantee that it is completely free of bugs, and the author cannot take any responsibility for consequences of program bugs, misunderstandings when using the program or any other errors. If results are published, reference to the source would be appreciated, and if modifications are made by future users, that should be clearly stated,

the modified program should be re-named and there should be reference to the original.

Organizing of the manual
Section 2 is a complete documentation of all algorithms and models. Section 3 on input-output describes how to specify both the options to be used, and actual values for parameters, as formal descriptions of file formats and contents. There are some cross references between Sections 2 and 3. Section 4 contains some practical advise. The manual is not intended as a textbook in management simulation - it is assumed that the user is familiar with that field.

2. Models and options.

The program is designed the standard way for harvest rule simulation programs. It consists of a population (operating) model that generates yearly true stock numbers at age, an observation ('assessment') model that transfers the stock numbers into noisy, 'observed' numbers, decision rules by which a TAC is derived according to the observed stock (projected forward if relevant) and an implementation model that translates the TAC into actual removals. These removals are then input to the population model for the next time step. The outline is shown in Figure 1 and the components are further described below.

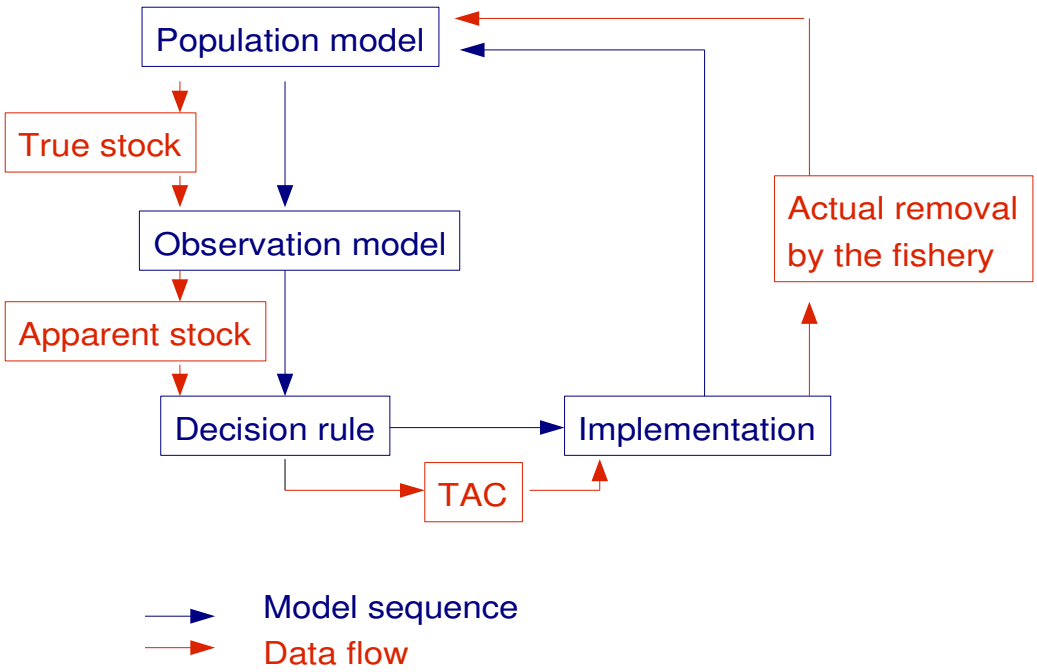


Figure 1. *Outline of the simulation loop in the hcs programs.*

2.1 Population model (operating model):

This model projects the vector of stock numbers at age (*N(a,y)*) forwards in yearly time steps *y* with a given fishing mortality *F* and natural mortality *M*:

*N(a+1,y+1)=N(a,y)*exp(-M(a,y)-F(a,y))*.

The oldest age *A* is a plus-group, modelled as a dynamic pool:

*N(A,y+1)=N(A,y)*exp(-M(A,y)-F(A,y))+N(A-1,y)*exp(-M(A-1,y)-F(A-1,y)).*

The fishing mortality is separable:

*F(a,y) = Fy(y)*Sel(a).*

The selection is normalized so that the *Fy* is scaled as the average *F* over a standard range of ages. Although the selection at age is assumed to be constant in the decision process, the realized fishing mortalities (and hence the realized selection at age) can vary because it is derived by applying 'real' catches at age coming from the implementation model, where noise may be included at individual ages.

Each year, a new year class is entered at the youngest age, according to a stock-recruit model. There are many options for stock-recruit models, as described later on. If the youngest age is greater than 0, recruits are still entered at age 0, but projected forward with zero mortality and weight until they reach recruiting age. Initial stock numbers at age (numbers present at the start of the simulation) can be obtained either by priming the population with a fixed F and stochastic recruitments and weights, or by taking numbers from a file. When taking numbers from a file, there are several options for adding noise to the initial numbers - see Section 3.2.3.

Weights and maturities at age can be entered as such, or derived according to a von Bertalanffy growth model and a logistic maturity ogive (see section 3.2.1 - ***p-option)*** . They can be made dependent on the total biomass (TSB) in the year before, on year class strength, have periodic fluctuations and have stochastic variation added, with truncated normal or lognormal distributions.

Note on total biomasses:
HCS operates with two kinds of total biomass:

- **TSB** from a lowest age specified in the bio.inn file and using stock weights. This is the general TSB, that is reported in the results, and used for density dependence and other biological relations.
- **B+,** from a lowest age specified in the opt.inn file, using catch weights. This is used in harvest rules where a decided harvest rate is converted to a TAC. Likewise, B+ is imitated when a biomass survey is used as decision basis in a harvest rule.

2.2. Details of the population model

2.2.1 Variable growth, including density dependence.

Weight if individuals in the catch (catch weight) and in the stock (stock weight) are handled separately, but are partly linked, as described below. The catch weights are used to convert catches in numbers to catches in weight and vice versa, while stock weights are used to convert stock numbers to biomasses.

The default is weights and maturities at age being constant over time. However, the following options are available for the annual weights and maturities:

- Dependence on the total biomass
- Dependence on year class strength
- Periodic trends

- Random variations

All the parameters in these functions, including the reference (standard) weights, are specified in the ***bio.inn*** file.

The starting point is the deterministic reference weights (catch weights or stock weights, here commonly termed *w0*) The final weights are then calculated as

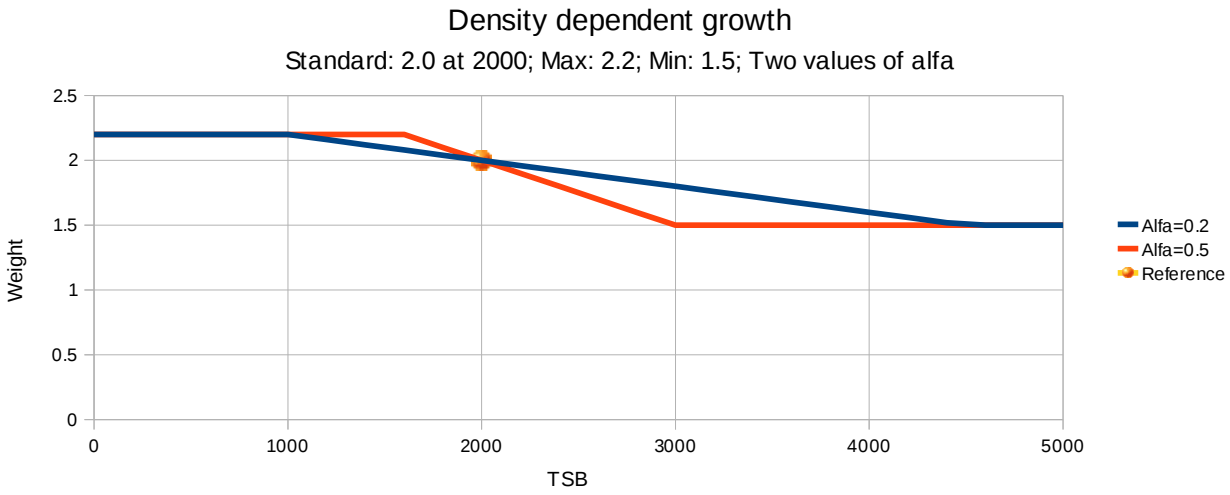
$w(a,y) = w0(a)*TSB\_factor(y)*Ycl\_factor(y-a)*Trend\_factor(y)*Random\_factor(a,y))$

Each of the factors is 1 if no modification takes place.

TSB_factor* and *Ycl_factor are linear functions with bounds. The *TSB_factor* applies in a year to all ages and is a linear function of the total biomass *TSB* in the previous year. Following Kovalev and Bogstad (2005), it is expressed by a slope α_{TSB} and a reference TSB termed *B0*, at which the factor is 1.

$TSB\_factor = 1+\alpha_{TSB}*(TSB-B0)/B0$

The model is illustrated in the figure below, with an arbitrary default weight = 2.0 in this example..



The year class factor *Ycl_factor* is derived the same way. It depends on the recruitment value *R* of the year class, and the reference is R0

$Ycl\_factor = 1+\alpha_{Ycl}*(R-R0)/R0$

The *Ycl_factor* follows the year class at all ages.

Both the TSB factor and the year class factor are truncated at Factor_low and Factor_high:

$Factor = max(min(Temp\_Factor;Factor\_low);Factor\_high)$

The ***Trend factor*** is a cosine function that applies to all ages in a year. It is specified by an amplitude, a period and a phase:

$Trend\ factor(year) =1+ ampl*cos(2\pi*(year-phase) /period)$

The amplitude should be < 1, to avoid negative weights.

The ***Random_factor(a,y)*** is the product of two normally distributed random terms:

1. A year factor, common to all ages.
2. An age factor, specific for each individual age and separate for each of the weights and the maturities.

Both the year factor and age factor components of the random factor are normally distributed with mean 1 and a sigma as specified. Both are updated in each simulation year, independently of the previous values. Hence, growth rates are not modelled explicitly, only the weights themselves.

This algorithm applies to both weights in the catch and weights in the stock. Each of these have a specific *w0(a)* and a specific α_{TSB}. and a specifically drawn age component in the *Random_factor(a,y),* while all other factors in the calculation, including random terms are shared between the two. This is to ensure that catch weights and stock weights change in some synchrony.

Reference: Y. Kovalev & B. Bogstad 2005: Evaluation of maximum long-term yield for North-East Arctic cod. Proceedings of the 11th Russian-Norwegian Symposium , Murmansk, 15-17 August 2005, PINRO Press. Murmansk 2005 .

2.2.2 Variable maturity, including density dependence

The algorithm for weights is not suited for maturity (fraction mature at age). All fish should mature sooner or later, so functions for density dependence that cannot reach one at very high age are not feasible. Instead, we assume that the effect of a large TSB is to delay maturation. A direct modelling of maturation, involving the probability that a fish that is still not mature will mature in the coming year, can become rather complex. A more simplistic approach is taken here, where the fraction mature as a function of age is just moved along the x-axis. For example, if the maturation is delayed, the proportion mature at age *a* becomes the proportion specified for a lower age. The move is expressed as a delay factor, such that the proportion mature at age *a* is taken as

Reference proportion mature at the age a / Delay_factor.

The delay factor is the product of a *TSB-sensitive term*, a year class *Ycl_factor* and *Random* multipliers as described for the weights above. The sensitivity of the age delay to TSB is expressed by a sensitivity factor α_{delay}: Then

$$Delay\_factor = (1+\alpha_{delay}*(TSB-TSBref)/TSBref)*Ycl\_factor(y-a)*Trend\_factor(y)*Random\_factor(y,a))$$

The reference maturity ogive is input or derived at age, and connected with straight lines between the ages.

The α_{delay} and the age component in the *Random_factor(y,a)* are specific for the maturation, the other factors that go into the calculation of the delay factor are shared with the weights calculation. Hence, maturity and weights are to some extent altered synchronously. However, the usage of the delay factor implies that maturation age is inversely related to growth, i.e. rapid growth leads to early maturation. Further, there is a minimum and maximum proportion mature at each age, specified in the ***bio.inn*** file.

Updating of weights and maturity takes place when projecting the true stock one year ahead. The weights and maturities used in the decision process are always the most recently derived true values, which also are assumed valid for the whole projection period if the observed stock is projected forwards as part of the decision process.

It is strongly recommended to check that the specification of the weights and maturities actually leads to the expected results. A printout option is available that presents the annual weights and maturities at age for each of the first 10 iterations in the simulation (print option 25, file Wema).

2.2.3. Recruitment models.

HCS allows for a wide variety of recruitment regimes, and for shifting between them, to mimic regime shifts if wanted.

Regimes and regime shifts.
Recruitment regimes are characterized by the set of recruitment function specifications as outlined below.
HCS13_1 allows for several regimes and shifts between them.

On the ***bio.inn*** input file there is first one line to specify the number of regimes and the way to shift between them. Then each regime is specified in a block of 4 lines, Regime shifts are specified by the years where they take place and the set of recruitment function specifications of each regime.

There are two options for deciding on regime shifts:

- Predetermined years and regimes: The years where shifts take place are specified as input. The regimes will apply in the order they are specified on the input file. The number of shifts must be consistent with the number of regimes, i.e. number of shifts = number of regimes -1. This option is suitable for examining the robustness of a rule to shifts in the recruitment regime.
- Random years and regimes: The years are drawn randomly, according to the geometric distribution of 'time x to next success' in a Bernoulli trial:
  $Prob(x\ years) = 1/mean * (1-1/mean)^{x}$, where the mean is specified.
  At each shift, the regime to be applied is drawn randomly amongst those available. This option is suitable for examining the effect of uncertainty with regard to future recruitment regimes.

In both cases, the simulation will always start with the first set of recruitment parameters, which should be consistent with the initial stock numbers.

Note that in ***hcs***, recruitment is always calculated at age 0. If recruiting age is higher, the year classes are just carried forward with no mortality or weight until they reach maturation age. This implies that when a regime shift takes place in one year, it will not materialize in the recruitment until those born after the shift have reached recruitment age, which can be several years later.

To use the same regime throughout, set the number of regimes to 1.

Recruitment function specifications for each regime.
Each recruitment regime is characterized by a set of specifications. These define the recruitment as a product of several multipliers:

- A deterministic stock recruitment function *R0(SSB)*, which can be Hockey stick (in which constant recruitment is a special case), Beverton-Holt or Ricker. The Hockey stick and Beverton – Holt also have an option for depensation.
- A periodic component multiplier $\xi period$, as a cosine function with amplitude, period and phase.
- Occasional strong year classes (sometimes termed spasmodic recruitment), as a multiplier $\xi spasm$ in selected years
- Random noise with a specified distribution with mean value = 1 and a dispersion parameter, as $\xi noise$, drawn for each new recruitment.

Each years recruitment is the product of these:
$R(y) = R0(SSB) * \xi period(y) * \xi spasm(y) * \xi noise(y)$

Each of the components are described in detail below:

The *Deterministic S-R functions* are (with parameters *a_par* and *b_par*)

1. Hockey - stick: *S>b_par: R = a_par*
   *S<b_par: R = a-par*SSB/b_par*
2. Beverton-Holt: *R = a_par*SSB/(SSB+b_par)*
3. Ricker: *R = a_par*SSB/b_par*exp(1-SSB/b_par)*

Note the form of the Ricker function, the *a* parameter is the maximum recruitment and the *b* parameter is the biomass at the maximum recruitment.

Depensation.
A depensatory stock recruitment function is available for the Hockey stick and Beverton-Holt functions. The depensation is implemented as a power term.

1. Hockey - stick: *S>b_par: R = a_par*
   S<b_par: R = a_par(SSB/b_par)^{power}*
2. Beverton-Holt: *R = a_par*SSB^{power}/(SSB^{power}+b_par^{power})*

A power term = 1.0 is neutral. A power term >1.0 implies depensation towards the origin.

Periodic fluctuations in the recruitment
Periodic fluctuations is included through a multiplier $\xi period$, specified by an amplitude, period and phase.

$$\xi period(Year) = 1.0 + Amplitude * \cos(2\pi * (Year - Phase)/Period)$$

The amplitude should be in the interval $0.0 \leq Amplitude \leq 1.0$. The period is in years. The phase is a year (relative to the starting year of the run) where the periodic function has its maximum. The option can be used both to include periodic fluctuations in the recruitment, and to generate a trend. To make a downward trend, the period should be about the double of the time span for the simulations, and the phase about -1/4 of the period. The recruitment will then start at the standard level specified in the recruitment function, and be reduced to a fraction *1-Amplitude* of the standard level at the end of the simulation period.

To exclude periodicities, set the amplitude to 0. The period should not be 0.

Spasmodic recruitment
This is an option to include occasional strong year classes, that come with more or less regular intervals. If asked for, the program sets up - for each iteration - a list of years where the recruitment is multiplied with a specified fixed magnitude factor $\xi spasm$. This is done by drawing random intervals, specified by a *mean* and optionally a *sigma*. The first interval starts in the first 'spasmodic' year, which may be a year in the past (stated relative to the starting year). In these years with high recruitment, the multiplier $\xi spasm$ applies.

There are three options for deriving random intervals:
1. According to a log-normal distribution with specified *mean* and *sigma*.
2. According to the geometric distribution of 'time x to next success' in a Bernoulli trial:
   $Prob(x\ years) = 1/mean * (1-1/mean)^{x}$
3. According to a flat (boxcar) distribution, with specified *mean* and relative *range*, centered at mean and covering *(mean-range*mean, mean+range*mean)*

The first and third options are best suited to the situation where the intervals are supposed to be more or less regular, in particular if the purpose is to investigate how dependent management is on the regular occurrence of big year classes. The second is the right formulation when examining the effect of the uncertainty caused by sporadic year classes occurring as random events with a probability of *1/mean.*

To exclude spasmodic recruitment, set the magnitude factor to 1.0.

Random noise
Noise around the deterministic stock recruit function (*ξnoise* - last bullet point above) can be normal or log-normal. In both cases the multiplier *ξnoise* is derived from a random number *x* with standard normal distribution. The variability of the multiplier is specified by a dispersion parameter *ps* which is a CV with the normal distribution and a sigma with the log-normal distribution:

1. Normal distribution: $\xi noise = (1+ps*x)$
2. Lognormal distribution: $\xi noise = exp(ps*x-ps^2/2.0)$

Truncation
The random noise multiplier *ξnoise* can be truncated, to avoid randomly drawn recruitments outside what may be considered as a realistic range. There are two options for how the truncation is done, either on the primary standard normal number *x* or on the final number *ξnoise.* For each option there is a lower and upper bound.

The rule is to draw a new random number if:
Option 1: $x > trunc\_upper$ or $x < trunc\_lower$
Option 2: $\xi noise > trunc\_upper$ or $\xi noise < trunc\_lower$

Note that applying constraints also means that the distribution is no longer normal (or lognormal), and that asymmetric constraints will alter the mean recruitment. *Therefore, it is strongly recommended to inspect the distribution of actual recruitments (which can be obtained with printout option 20) and compare it with the recruitment distribution you want to imitate.* Another check can be obtained by comparing the output from the Y/R calculations with the original stock-recruit relation (see Section 3.4)

All the options described above can be combined, for example a trend with sporadic strong year classes, and random noise on top of that. The full set of options is specified separately for each regime, hence it is for example fully possible to have one regime with spasmodic recruitments and another without.

2.2.4. Selection at age.

Selection at age is entered as input ***(bio.inn*** file). When implementing a TAC, the resulting noisy catch numbers at age will lead to a different (realized) selection, which is what the operating model will use to reduce true stock numbers according to mortality. The original selection is used everywhere else, including in the decision model and there is no options to change it over time.

2.2.5 Initial numbers

These are the numbers in year 0, which is the start of the simulation. When working with a specific stock, these numbers will typically be taken from the last assessment of the stock, and be similar to those used for the intermediate year in a short term prediction. Alternatively, HCS can provide initial numbers by running the stock to a stochastic equilibrium with a fixed fishing mortality, here called ***priming*** with fixed F.

When numbers from an assessment are used, there may be uncertainty to these numbers. This uncertainty should be regarded as representing the plausible range of values for the true present state, which conceptually is different from observation and implementation error. However, many analysts prefer to let the range of plausible initial numbersbe represented by the uncertainty in the present assessment. If that uncertainty is poorly known, a fair assumption about the plausible range of initial numbers may be that it has the same uncertainty that is assumed for future assessment.

HCS allows for 3 options for specifying uncertainty in initial numbers (option 1 is priming):

2. Numbers at age are entered together with a variance-covariance matrix of the logarithms to the numbers
3. Numbers at age are entered together with CVs, assuming a lognormal distribution, and no correlations
4. Numbers are entered, and the observation model is used to draw random starting numbers for each bootstrap replica.

The option to be used is linked to the format of the nin.xxx file, where the actual suffix indicates the format (see Section 3.2.3. When priming, the fixed F is implemented without error, and the replicas vary due to variation in recruitment, growth and maturity.

2.3. Observation (assessment) model:

This model transforms the vector of true *N*-values at age to observed *N*-values at age. Note that the term observations here refers to the stock numbers at age that typically are estimated in an assessment. The usage of the term is sometimes different in other contexts. The algorithm in ***HCS12_1 and later*** differs from that in previous versions.

The starting point is true stock numbers in a reference year (typically the year before the TAC year). These numbers are modified by stochastic multipliers. The multipliers are derived as an autoregressive process along the year classes, described below, to mimic the influence of noisy input data in an assessment. If stock numbers according to a mimicked assessment are needed further back in time, such numbers are derived through a VPA, using observed catches and starting with the stock numbers in the reference year. The observed catches are input for the years prior to the starting year of the simulation (file ***canum.inn***), else they are derived by applying noise multipliers to the implemented true catch numbers at age.

This procedure should bring in some of the noise structure that will occur in a full assessment. In particular, the effect of noise in the tuning data (surveys or CPUE), which will fade away over a number of years, is mimicked, and if the history is needed, it will come from a VPA which is updated each year. This is not a substitute for a full assessment in the simulation loop, but is probably about as close one may come without actually fitting a parametric population model, which would increase the computing time drastically.

The annual noise in the 'tuning' data is assembled in a matrix (age,years) of noise terms *x(a,y)*. Each year *y*, a new year column (i.e. year) is added to that table. The terms are derived as followed.

- Draw a random number x_1 (year factor) from a lognormal distribution (with bias correction) with an input year factor CV.
- For each age *a*: Draw a random x_{2a} number (age factor) from a lognormal distribution (with bias correction) with age-specific CVs
- Take the product $x(a,y) = x_1 * x_{2a}$ as entries in the noise matrix.

When imitating an assessment in year *y*, observed stock numbers *Nobs(a,y)* at age *a* in year *y* are derived as follows:

- Take *N(a,y)* from the true stock
- Derive for each age *a* a number $\xi(a) = \alpha_0 * x(a,y) + \alpha_1 * x(a-1,y-1) + \alpha_2 * x(a-2,y-2) + ... \alpha_i * x(a-i,y-i)$, for as many years backwards as can be assumed to be influenced by a noisy survey observation, or to the first year the year class appears. The maximum number *i* of years will depend on how fast the VPA converges, but *i = 4 or 5* probably should be sufficient.
- The observed stock numbers in year y are then taken as $Nobs(a,y) = N(a,y) * \xi(a)$

Hence, the influence of previous observation noise follows the year classes. The coefficients α_i are input. As a guideline, one may argue that the influence is related to the fraction of the N back in time that is needed to account for the current survivors. That will decline backwards in time according to *exp(-Cum(Z-M)):* the *N=S*exp(CumZ)* and the amount needed for the survivors is *S*exp(CumM)*. The α_i are scaled internally so that the sum of the

applied α_i is 1.0. If this algorithm is not to be used, and the drawn $x(a,y)$ is used directly, i.e. $\xi(a) = x(a,y)$, the number of α_i -terms should be set to 1 and the α_0 to 1.0

If the observation model values for *N* are needed backwards in time, the following VPA-algorithm applies:

- Build a catch matrix *Cobs*
  - For years prior to the start of the simulation, observed catch numbers at age are input.
  - For each new year *y*, derive observed catches from the true catches at age *C(y,a)* in year *y* from the implementation model.
    - Draw a random ζ_1 number from a lognormal distribution with an input CV.
    - For each age *a*: Draw a random ζ_{2a} number from a lognormal distribution, and take the product $\zeta(a,y) = \zeta_1 * \zeta_a$
    - Take $Cobs(a,y) = C(a,y) * \zeta(a,y)$

- Calculate stock numbers backwards with Popes equation, using the present and the previous *Cobs* values: $Nobs(a-i,y-i) = Nobs(a-i+1,y-i+1) * exp(M(a-1)) + C(a-i,y-i) * exp(M(a-1)/2)$. The still existing year classes are started with the *N*-values in the observation model. Older year classes are started with *N*-values derived from the catches in *Cobs* at the oldest true age and mortalities equal to those at the penultimate true age.

The *Nobs* matrix is revised each year. The older versions are kept. Accordingly, the *Nobs* matrix has the dimension (*age, y1, y2*) and contains perceived values for year *y1* as estimated in year *y2*.

In version 13_2, a bug in this routine was corrected, which may lead to slightly different results from previous versions.

2.4 Implementation model:

This model transforms a TAC coming from the decision model to an actual catch, both in tonnes and in numbers at age, and derives the true fishing mortality accordingly.

- The TAC is translated into catch numbers at age by a searching routine that finds the overall F leading to the TAC when applied to the true population, assuming the standard selection at age and the currently valid weights at age.
- Then noise is added to the catch numbers at age as age dependent log-normally distributed random noise, with CVs specified individually for each age.
- The *realized total catch* is the decided TAC multiplied by a log-normally distributed year factor, biased if requested.
- The numbers caught are adjusted with a common factor to give a sum of products of the catches and catch weights equal to the realized total catch.
- Realized fishing mortalities are derived from the resulting catches in numbers at age and the true stock numbers.

If in this procedure, some true stock numbers at age are too small to allow the realized catches, new catches are derived assuming a fishing mortality specified as *Fmax*. The TAC remains for reference, but the true stock numbers are reduced according to Fmax. This procedure is only included to prevent the program from crashing, and is not intended as a part of the harvest rule. Accordingly, *Fmax* should be set at an unrealistically high value. A flag is set that indicates that the stock has collapsed in this iteration.

2.5 Decision model:

General outline

The decision process leads to a *decided TAC for year y*. The process is strictly structured in several steps which are outlined here and described in detail below. The underlying idea is to find a rational and structured way to

design harvest rules. In this framework, first a primary TAC is set according to a rule, then this primary TAC is modified through additional rules. This leads to a sequential decision process.

In each step, a decision is taken according to a **basis**, which is the information that goes into the decision. Most often this will be the state of the stock expressed as SSB at some time, but in principle, any relevant information can be taken into account. Furthermore, the basis does not have to be a single number, it may well be a vector, where each component has some role in the decision process. For example, the decision can depend on a survey measuring the total biomass, supplemented with the trend in a recruitment survey. HCS has some options for deciding on bases, but not everything thinkable. The first limitation is that it must be derived from the simple population model that is used here. Hence, length distributions, area distributions or climatic variables cannot be included. Nevertheless, by combining the present opportunities, a quite wide range of decision bases can be explored.

The rule itself is a parametric function of the decision basis. The parameters are both standard levels for for example F, and trigger or reference points related to the basis, which can lead to different decisions depending on the basis. The repertoire of rule functions in ***hcs*** has evolved gradually as various people have launched new ideas.

The outcome of the rule can be a TAC, but more often, it is some other measure, for example F, that has to be t**ranslated** into a TAC.

When the primary TAC has been calculated, it may be modified by additional rules. This is a very common feature in management plan, and the purpose is most often to reduce inter-annual variation in the TACs. HCS has two such rules, a 'filer rule' and a 'percentage rule', that are applied in sequence. Each of these are conditional on the state of the stock. Hence, both have a basis, that is derived with algorithms similar to those used to derive the basis for the primary TAC, but of course independently of the basis for the primary TAC.

Finally, there is the option to set a minimum and/or a maximum TAC.

In summary, the decision sequence is:

First step (primary TAC):

- A **basis** derived from the perceived state of the stock, projected forwards in time as needed.
- A **rule** derives a **measure of exploitation** from the basis.
- If needed, this measure is **translated** into a TAC.
- If the basis depends on the TAC (e.g. the basis includes a biomass after the TAC has been taken), the process is repeated in an iteration loop until convergence

This gives a **primary TAC**

Second step (modifying the TAC): The primary TAC can be modified through a sequence of constraint rules to give the final TAC. Optionally a TAC can be set for several years starting in year *y*.

Details of these steps are given below.

Projection in the decision process.
The perceived stock numbers are obtained from the observation model by 1. Jan in some previous year referred to as the *last assessment year* - the last year with estimates of *N* at the start of the year from an assessment. The typical case when setting a TAC for year *y* is that the assessment provides stock numbers at the start of year *y-1* (the 'intermediate' year). If needed, and that is the typical case, the stock numbers are projected forwards, starting with the stock numbers in the last assessment year, and the projection is carried on as far as needed.

This projection is deterministic, with specified assumptions for catches or fishing mortalities. Weights and maturities are the true values from starting year for the projection. Recruitments are according to the deterministic stock-recruit function in the regime valid in the recruitment year, with no periodic or spasmodic variation. Year classes that have been born, but are not recruited yet, are included with their true values. Strong or weak year classes are known if and only if they have already been born. If there are several recruitment regimes, the regime

in the year where the projection starts is assumed. Hence, future shifts in recruitment are not accounted for. In version 13_2, there is also an option to take the recruitment as the mean of previous recruitments, as calculated by the observation model in the starting year of the projection.

If the decision basis includes years even later than the TAC year *y*, or refers to some time during the TAC year *y* (e.g. SSB when spawning is later than 1. Jan), the basis for decision will depend on the TAC. If so, the projections are repeated iteratively to get a TAC in accordance with the decision basis. These projections assume that the decided or temporary TACs are taken. If the projection period goes beyond the last TAC year, the same F as in the last TAC year is assumed.

2.6 Details of the decision process:

The timing of various values in the decision process are always stated as years ***relative to the first TAC year, which here is referred to as year 0.*** Hence, if there is an intermediate year, the last assessment year will be -1

As outlined above, the decision process is a two steps sequence: A basic rule and a sequence of TAC modifying rules ('stabilizers') with possible exceptions.

2.6.1 Step 1. Find a primaryTAC according to the basic rule:

The general template for a basic rule is that it derives a ***measure v*** of exploitation as a ***function*** of a ***basis vector*** and a vector of ***parameters***. The program has several rules, each identified by a number, and the collection can be extended by coding new rules. The rule is blind in the sense that it does not consider the meaning of the entries in the basis vector and parameter vector, it is only a formula that treats them as numbers. This framework was designed to allow the user to study a wide range of harvest rules, by specifying measure, basis or parameters freely.

a) Specify the basis.
The basis is a vector, where each entry is some information that should be taken into account when deciding the future exploitation. Each entry is derived by defining *a type*, *a time frame* and *a calculation method*, which are stated as input in the options file. For the time being, the following elements are available.

- Type:
  - 1. SSB according to observation model
  - 2. TSB according to observation model, derived using the catch weights.
  - 3. Perceived past recruitment according to observation model
  - 4. A 'survey' recruitment indicator (e.g. a noisy recruitment survey, derived by adding noise to the true recruitment)
  - 5. Result of a spawning biomass survey, derived by adding noise to the true SSB.
  - 6. Result of a total biomass survey, derived by adding noise to the true TSB.
- Time frame:
  - First and last year (relative to the first TAC year)
- Calculation:
  - 1. Average in the period
  - 2. Smallest in the period
  - 3. Slope in a linear regression over the period. The slope is relative to the mean, i.e. if the regression equation is $y = b*x + a$, the slope here is *b/ymean.*

b) Specify the rule:
The functions are numbered. To each function, there is associated a dimension of the basis vector and of the parameter vector, and the function form is coded.

The parameters are input, and the dimension of the input parameter vector must be in accordance with the

requirement of the selected function. The parameters can be included in the scanning loop.

Functions available at present:

1. ***'Alpha-rule':***

Basis: 2 dimensions
Rule parameters:
Btrig1,Btrig2, vstd, alpha1 and *alpha2*

- If Basis(1) < Btrig1: *v = vstd*(1.0-alpha1*(btrig1-Basis(1))/btrig1)*. If that leads to *v<0*, set *v = 0*
- If Basis(1) > Btrig1 and Basis(2) < Btrig2: *v = vstd*
- If Basis(1) > Btrig1 and Basis(2) > Btrig2: *v=vstd*(1.0+alpha2*(Basis(2)-btrig2)/btrig2). If that leads to v<0, set v = 0*

Note that the Basis(2) is used to modify the *v* relative to the *vstd*, and only when *Basis(1)* is above the *Btrig1* and *Basis(2)* is above the *Btrig2*. Note that *Basis(1)* and *Basis(2)* are derived independently of each other. For example *Basis(1)* can be an SSB and *Basis(2)* a measure of recruitment.

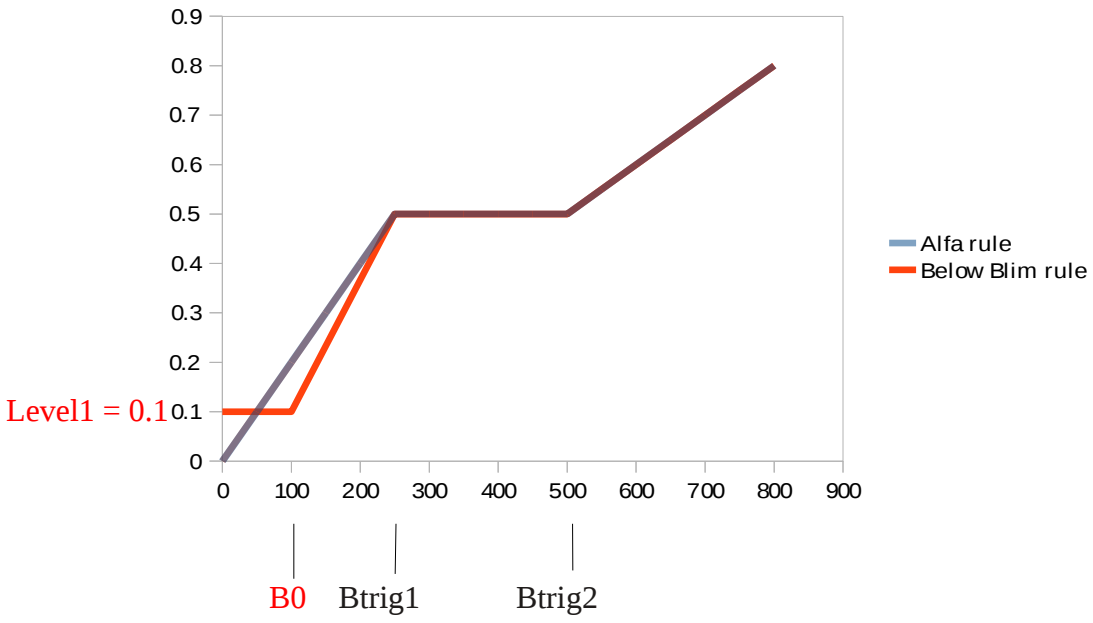
2. 'Small v below Blim':

Basis: 2 dimensions
Rule parameters:
B0, Btrig1,Btrig2, vstd, Level1, alpha1 and *alpha2*

The difference from rule 1 is at low SSB. There is a third trigger, called *B0*, below which *v = Level1*, which represents the 'lowest possible exploitation'. Typically, it will be *Blim*, but it is independent here. Between *B0* and *Btrig1*, the v is linearly changing with *Basis(1)*. Above Btrig1, the rule is similar to rule 1. Note that the *B0* ref. point refers to *Basis(1)*.

- If Basis(1) < B0: *v = Level1*
- If B0 < Basis(1) < Btrig1: *v = vstd*(1.0+Level1*(Basis(1)-btrig1)/(btrig1-B0))*
- If Basis(1) > Btrig1 and Basis(2) < Btrig2: *v = vstd*
- If Basis(1) > Btrig1 and Basis(2) > Btrig2: *v=vstd*(1.0+alpha2*(Basis(2)-btrig2)/btrig2)*

The rules 1 and 2 are illustrated in the graph below. The x-axis is SSB or B+ . The *v* (for example F) is on the y-axis. The rules and their respective alpha1 have different colours.



3. 'Alpha-rule with variable level':
Basis: 3 dimensions
Rule parameters:
Btrig1,Btrig2, vstd, alpha1, alpha2, gain

The parameter *vact* is an intermediate variable, that represents the 'standard' level of *v* conditional on the third dimension *Basis(3)* of the basis- presumably a slope.

- *vact = vstd*(1+gain*Basis(3))*
- If Basis(1) < Btrig1: *v = vact*(1.0-alpha1*(btrig1-Basis(1))/btrig1)*. If that leads to *v<0*, set *v = 0*
- If Basis(1) > Btrig1 and Basis(2) < Btrig2: *v = vact*
- If Basis(1) > Btrig1 and Basis(2) > Btrig2: *v=vact*(1.0+alpha2*(Basis(2)-btrig2)/btrig2). If that leads to v<0, set v = 0*

Rule 4. Smooth transitions: Upper part of logistic function below B1 and above B2
Basis: 2 dimensions
Rule parameters: *Btrig1,Btrig2, vstd, slope1, slope2, vmax.*

In the formulas below, *vact* and *v0 are i*ntermediate variables.

- If Basis1<Btrig1) — *vact=1.0/(1.0+exp(-slope1*(Basis1/Btrig1)))*; *v0=1.0/(1.0+exp(-slope1))-0.5*; *v=vstd*(vact-0.5)/v0*
- If (Basis1>Btrig1 and Basis2 >Btrig2) — *vact=1.0/(1.0+exp(-slope2*(Basis2-Btrig2)/Btrig2))*; *v=vstd+(vact-0.5)*(vmax-vstd)*
- Else — *v=vstd*

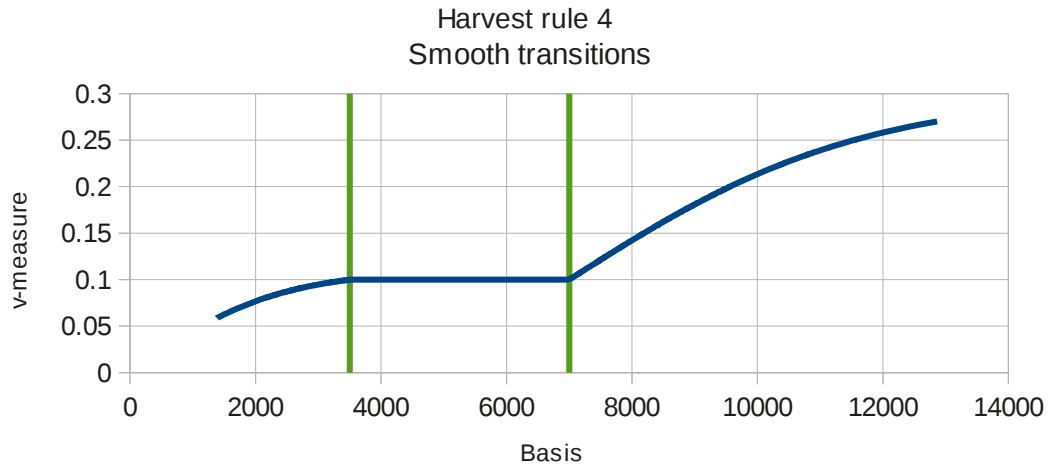


Figure: Rule 4 with B1 at 3500 and B2 at 7000

5. 'Alpha-rule' with two levels of vstd:
Basis: 3 dimensions
Rule parameters:
Btrig1,Btrig2, RefR, vstdlow,vstdhigh, alpha1 and *alpha2*

This rule is equal to rule 1, except that it has two standard values for *v*, depending on the third basis *Basis(3)*

Basis(3) is intended to be a recruitment indicator, but can in principle be any measure. It is also related to rule 3, but rule 5 has only two levels of *vstd*, while rule 3 has a continuum. In the formulas, *vstd* is an intermediate variable.

- If RefR < Basis(3) *vstd=vstdlow*
- If RefR > Basis(3) *vstd=vstdlhigh*
- If Basis(1) < Btrig1: *v = vstd*(1.0-alpha1*(btrig1-Basis(1))/btrig1)*. If that leads to *v<0*, set *v = 0*
- If Basis(1) > Btrig1 and Basis(2) < Btrig2: *v = vstd*
- If Basis(1) > Btrig1 and Basis(2) > Btrig2: *v=vstd*(1.0+alpha2*(Basis(2)-btrig2)/btrig2). If that leads to v<0, set v = 0*

6. 'Small v below Blim' with two levels of vstd:
Basis: 3 dimensions
Rule parameters:
B0, *Btrig1*,*Btrig2*, *RefR*, *Level1*, *vstdlow*,*vstdhigh*, *alpha1* and *alpha2*

This rule is equal to rule 2, except that it has two standard values for *v*, depending on the third basis *Basis(3)*. *Basis(3)* is intended to be a recruitment indicator, but can in principle be any measure. In the formulas, *vstd* is an intermediate variable.

- If Basis(3) < RefR *vstd=vstdlow*
- If Basis(3) > RefR *vstd=vstdhigh*
- If Basis(1) < B0: *v = Level1*
- If B0 < Basis(1) < Btrig1: *v = vstd*(1.0+Level1*(Basis(1)-btrig1)/(btrig1-B0))*
- If Basis(1) > Btrig1 and Basis(2) < Btrig2: *v = vstd*
- If Basis(1) > Btrig1 and Basis(2) > Btrig2: *v=vstd*(1.0+alpha2*(Basis(2)-btrig2)/btrig2)*

7. 'Alpha-rule with two slopes (sardine rule)':
Basis: 1 dimension
Rule parameters: *Btrig1*,*Btrig2*, *alpha1*, *alpha2*, *vstd*

- If Basis(1) < Btrig1: *v = alpha2*(1.0+(Basis(1)-btrig1)/btrig1)*. If that leads to *v<0*, set *v = 0*
- If Basis(1) > Btrig1 and Basis(1) < Btrig2: *v=alpha2+((Basis(1)-btrig1)/(btrig2-btrig1)*(btrig2-btrig1)*.
- If Basis(1) > Btrig2: *v = vstd*

Note that the *alfa2* is the *v* when *Basis(1) = btrig1*, while *alpha1* is the slope below *btrig1*.

8. 'Alpha-rule with bounds':

Basis: 2 dimensions
Rule parameters:

Btrig1,Btrig2, vstd, alpha1 and *alpha2,lower_bound, upper_bound*

- If Basis(1) < Btrig1: *v = vstd*(1.0-alpha1*(btrig1-Basis(1))/btrig1)*. If that leads to *v<0*, set *v = 0*
- If Basis(1) > Btrig1 and Basis(2) < Btrig2: *v = vstd*
- If Basis(1) > Btrig1 and Basis(2) > Btrig2: *v=vstd*(1.0+alpha2*(Basis(2)-btrig2)/btrig2).*
- *If v < lower_bound, v=lower_bound*
- *If v > upper_bound, v=upper_bound*

Note that the Basis(2) is used to modify the *v* relative to the *vstd*, and only when *Basis(1)* is above the *Btrig1* and *Basis(2)* is above the *Btrig2*. Note that *Basis(1)* and *Basis(2)* are derived independently of each other. For example *Basis(1)* can be an SSB and *Basis(2)* a measure of recruitment.

c) Translate the rule outcome (v) to TAC
Following the calculation of *v*, it is translated into a TAC, depending on the meaning given to it. The following meanings can be translated at present:

1. *v* is an F-value.
2. *v* is a Harvest rate, defined as TAC = HR*B+ in year 0 according to the observation model.
3. v is a Harvest rate, defined as TAC = HR* Basis(1), which can be of any type, including a survey (basis type 5 or 6)
4. *v* is the TAC itself (no translation needed)

Note that if basis is of type 5 or 6 (survey biomass) a HR is translated to a TAC as *v*basis*. In all other cases, the perceived stock numbers from the observation model are used. If v is a harvest rate, and the

Iterative search:
If the value of *v* is sensitive to itself, i.e. if it influences the basis, the process a-c is iterated until convergence. Typically, this happens when the basis includes a biomass after (parts of) the TAC has been taken.

2.6.2 Step 2. Apply constraint:

The constraint rules basically use the primary TAC for year *y* together with previous TACs. The constraint rules have the following elements that are evaluated in sequence (starting with the primary TAC derived above) and can be used alone or in combination:

1. *'Filter rule*': The TAC for year *y* is a weighted average of the primary TAC for year *y* and the decided TAC for year *y-1*, with a weighting factor called *TAC filter gain*:
   *TAC(y) = (1-gain)*TAC(y-1) + gain*primary TAC(y)*
   Optionally, the constraint may only apply if a basis exceeds a trigger level. The basis is defined the same way as above (but with its own parameters). If the basis is a trend, the filter only applies when the trend is lower than a trigger level, if it is a biomass it only applies if it is higher than the trigger level.
   Setting the gain to 1 disables the filter. To ensure that the filter always applies, use SSB as basis and set the trigger level to 0.
2. *'Percentage rule':* The TAC is not allowed to change by more than *x%* per year.
   The percentage rule (Rule 2) has two options for applying the rule, which in this case is stated specifically.
   1. If the constraint *always* applies, it is simply applied.
   2. If the constraint *only* applies when a basis type measure is exceeds a trigger level. The basis measure can be defined the same way as in the rules for setting the primary TAC, but can only have one dimension. There are two options:
      - If the basis is a trend: Constraint if Basis >Trigger
      - Else: Constraint if Basis< Trigger.

   A negative percentage disables this step.
3. The TAC shall not exceed a lower and an upper bound. In particular with rules that allow increasing

exploitation at high biomasses it is strongly recommended to have at least a maximum TAC. To disable this step, set the lower bound to 0 and the upper bound to an incredible high number.

2.6.3. Additional management options.

Multi-annual TACs
If the TAC is to be valid for more than one year, there are two options for how to change the TAC.

1. Abrupt: The new TAC is set for all years in the whole period
2. Gradual: The TAC is changed linearly until the new value is reached in the last year of the period.

Transition rule towards a target in the initial phase.
When a harvest rule is introduced, it may imply a quite substantial change from the current exploitation. Managers sometimes would like to make that change gradual. ***hcs11_2 and later*** versions have the option to change the rule target gradually from the current value (i.e. that observed for year 0) to the final target. This is specified as a 'number of years to reach target'. If this number is greater than 1, the target management measure value *v* to be applied in each year is derived according to a 'Target reduction rule'. When the target has been reached, the ordinary rules as described above take over.

The rule that is implemented is that the value of the target *vstd* is reduced linearly from the initial value to the final one over the specified number of years. This is done by setting up a table for future target values when the rule is introduced. Hence, the change is not relative to the currently perceived rule parameter, only to the initial one.

Note that this arrangement is different from some rules that have been proposed by EU in connection with the introduction of FMSY, where the reduction is relative to each years estimate of the previous years fishing mortality.

Some comments to the constraint rules:
The rules provide great flexibility, way beyond what is relevant in most cases. In practice, one will have either the option to set the TAC as a weighted average of the present temporary TAC and the previous TAC (referred to as the filter rule), or apply a constraint on the percentage change. Combining these two will imply a sequence of two constraints, if the filter is not enough, then there is a percentage rule on top of it. For most practical purposes, this is over-kill. Also, the option to change the TAC gradually when a multi-annual TAC is used, is rather similar to applying the filter rule. Combining the two means that the change will be linear but the target will be determined by the filter rule. Again, -1 1this is probably over-kill.

The *'only'* option in the exception rule for the percentage constraint covers the situation where the constraint applies when and only when the constrained catch leads to Basis above the trigger level. This is a common feature in proposed harvest rules, and may be necessary to ensure sufficiently swift action if the stock abundance goes down, but leads to the paradox that the TAC can get trapped at a low level when rebuilding the stock. This is a quite well known weakness of the percentage type of constraints. When this situation has appeared, managers have sometimes chosen to deviate from the rule, if that still leads to a satisfactory biomass. This tactics has not been implemented here, mostly because it is hard to foresee how managers will solve such problems.

2.6.4. Decision models vs. standard practice

The standard practice when setting a TAC varies a good deal, and the procedures in ***hcs*** may differ from these practices. In particular, the following points may matter:

Recruitment assumptions:

In ***hcs,*** the assumption is that existing year classes are known with error form the observation model. Year classes that have been born, but are not recruited are included in the stock numbers at age and carried forward with no mortality until they recruit. These numbers are not altered by the observation model. Hence, if recruitment is at a higher age than zero, year classes that have been born but have not reached recruitment age at the time where the

projection starts, will keep their 'true' values. Future year classes in projections are derived using the stock-recruit function for the regime that was valid at the start of the projection, with the SSB in the projected stock, but with no stochastic terms and no periodicities or spasmodic recruitments. Alternatively, it is taken as the mean over a number of previous years (as assessed in the last assessment year) from the observation model. Thus, the regime present in the last assessment year is assumed to be known, and to be continued into the future. The influence of management decisions on future recruitments is taken into account through the stock-recruit function, but the common practise to assume future recruitments as a geometric mean of recent recruitments according to the most recent assessment, or to derive recruitments from pre-recruit survey data, is not reproduced. Neither is there any mechanism for substituting the youngest ages in the assessment with mean recruitments.

Weights and maturities:

The weights and maturities used in the decision process are always the most recently derived true values, which also are assumed valid for the whole projection period. There is no averaging of weights and maturities that go into the decision model, and no assumptions are made about future change in weight. In particular, trends or year class effects are not continued.

Selection at age
HCS uses a standard selection at age, which is input, in the decision process. The 'true' selection at age in the operating model may deviate from this, because it is derived from the actual catches at age coming from the implementation model. The common practise to derive a selection for projections and decisions from an assessment is not implemented. Neither is there any opportunity to change selection over time so far.

3. Instructions for use

3.1 Setting up a simulation.

3.1.1 Preparation

To run a set of simulations, you will need to prepare the following files:

- bio.inn
- opt.inn
- nin.xxx (the suffix depends on the format)
- canum.inn

The format and contents of these files is described in detail in Section 3.2 below. The example files that are provided with the program can be used as templates to get all necessary lines in place, but they do not represent any kind of standard set up - there is no such thing! The nin.xxx and canum.inn are in principle optional, but in most cases you will need them.

These files are all ascii-files, and should be prepared with a 'clean' editor with no hidden tabulators etc. Make sure that there are no empty lines at the end of the file.

The opt.inn file has a list of outputs, described in Section 3.3 . Pick the ones you will need there. Note that on file 12, you will also have tot specify years. The upper year should not be above the maxyear (hard coded in the hcscom13_1.i file).

3.1.2 Running the program

Under Linux, the program is best run from a terminal window. The executable and the input files must be in the same working directory, and the output also goes there. Under Windows, the same applies, and it is recommended to run from a DOS window rather than just clicking on the executable in the file manager, to be able to see error messages. Under Linux, the program will have to be compiled before running. How that is done depends on the

compiler. With the ***gfortran*** compiler, the command will be *gfortran hcs13_1,f*. The program is then started by the command *./a.out.* Under Windows, the program is started by typing hcs13_1.exe in a DOS window in the working directory.

3.1.3 Modifying ranges.

The actual age range is input, on the biological data file. Note that the oldest age is always regarded as a plus age. There is a maximum age that is hard coded as the parameter *maxage* on the ***hcscom13_1.i*** file, any oldest age not exceeding that is valid.
The year range and the number of iterations are hard coded on the top of the ***hcscom13_1.i*** file as *maxyr* and *niter* parameters respectively. To change the year range or number of iterations, these parameters have to be changed and the program has to be recompiled. No changes are required elsewhere. However, the *maxyr* should not be set higher than 98, because some printout formats are adapted to 2-digit years.

3.2 Input.

There are two, optionally 4 input files: *bio.inn*, *opt.inn* and optionally a *nin*. file and a *canum.inn* file. The file formats are described here. All file names are standard and hard-coded. On the files, numbers must be space separated. On the *bio.inn* and *opt.inn* files, explanatory extra lines are permitted. Such lines must start with # **in position 1**. Extra lines are not permitted in nin-files or canum.inn file. Comments can be added after the numbers on each line - the program stops reading a line when it has got what it expects. Extensive use of comments is strongly recommended, the files themselves are rather cryptic and a wrong number in the wrong place can have drastic consequences and often causes the program to crash. Here, a formal specification of the formats is given, together with some examples. Note that different examples refer to different stocks.

The format is generally not backwards compatible with older versions of HCS. However, the format of the nin-files has never changed and the only changes in the other files from ***hcs12_1*** to ***hcs13_1,*** is that when requesting output to file 12 in the list of outputs on the opt.inn file, years must be specified from version 12_5 onwards, and version 12_5 also has the CV of a biomass survey as an additional screening parameter, which requires an additional line. Version 13_1 is fully compatible with version 12_5, but has an additional harvest rule (rule 7).

In the descriptions of the files below, line numbers refer to lines that are not comment lines. Entries are in this font.

3.2.1. Biological data (File bio.inn).

The input file for biological data is called ***bio.inn.*** There are two alternatives for the biological data, indicated by the letter ***i*** or ***p*** in the appropriate line (see below). The first part of the file (before the i- or p-line) contains specifications of basic bounds, of recruitments, and of general specifications for weights and maturities. The lines below the i- or p – line specifies biology by age, i.e. natural mortality, selection, weights and maturities, that can either be specified as such (i-option) or derived from a growth model (p-option).

Both options

Lines 1-5: Basic settings

- Line 1: Year 0 (Initial year for reference in tables etc). The TAC for year 0 is known, year 1 is the first year where a TAC is set according to the rules. The initial numbers (nin.xxx file) are at the start of year 0.
- Line 2: Youngest_age Oldest_age. The oldest age is always a plus-age.
- Line 3: Lower_age Upper_age in Reference age range for fishing mortality. Fishing mortality is reported as the average over this age range.
- Line 4: Youngest_age for calculating TSB for use in models for density dependent weight and maturity. This age is independent of the age for calculating TSB in harvest rules (opt.inn file).
- Line 5: Prop_F Prop_M Proportion of F and M before spawning

Lines 6 onwards: Specification of recruitment

- Line 6: Recruitment regime shifts. Two options:
    1. Number of regimes, Code for shift =1: input shift years, Years where shift takes place
    2. Number of regimes, Code for shift = 2: random shift years, Mean interval, Year where current regime started.
- Line 7 onwards: Blocks of 4 lines with recruitment options, one for each regime. The number of blocks has to match the number of regimes.
  The 4 lines in each recruitment block are:

1. **First recruitment line:** Keys for recruitment options:
   S-R-function Distribution Truncation type
    - Code for the S-R function: 1= Hockey stick, 2 = Beverton- Holt, 3=Ricker
    - Code for distribution type: 1=Normal, 2=Lognormal
    - Option for truncation: 1 or 2, see second recruitment line

Details of the S-R functions
The S-R functions are (with parameters *a_par* and *b_par* set in line 2)

1: Hockey - stick: $S>b\_par:\ R = a\_par$
$S<b\_par:\ R = a\text{-}par*SSB/b\_par$
2: Beverton-Holt: $R = a\_par*SSB/(SSB+b\_par)$
3: Ricker: $R = a\_par*SSB/b\_par*exp(1-SSB/b\_par)$

Note the form of the Ricker function, the *a* parameter is the maximum recruitment and the *b* parameter is the biomass at the maximum recruitment.

2. **Second recruitment line:** Parameters in recruitment function:
   a-parameter, b-parameter, ps, trunc_upper, trunc_lower, Depensatory gain power

ps - parameter.
Noise around the deterministic stock recruit function can be normal or log-normal. In both cases it is implemented through a multiplier $\xi noise$ derived from a random number x with standard normal distribution, and a dispersion parameter *ps* which is a CV with the normal distribution and a sigma with the lognormal distribution:

Normal distribution: $\xi noise = (1+ps*x)$
Lognormal distribution: $\xi noise = exp(ps*x)$

Truncation (lower and upper bound)
The interpretation depends on the truncation option and on x or $\xi noise$ (defined above).
Rule: Draw a new random number if:

Option 1: $x > trunc\_upper$ or $x < trunc\_lower$
Option 2: $xnoise > trunc\_upper$ or $xnoise < trunc\_lower$

Depensatory gain power:
Extends the stock recruit function with a power term, to include depensation at low SSBs:

If Hockey – stick: $S>b\_par:\ R = a\_par$
$S<b\_par:\ R = a\_par*(SSB/b\_par)^{power}$

If Beverton-Holt: $R = a\_par*SSB^{power}/(SSB^{power}+b\_par^{power})$

The power is not implemented for the Ricker function, the power term is ignored there.
A power term = 1.0 is neutral. A power term >1.0 implies depensation towards the origin.

3. **Third recruitment line:** Periodic recruitment variation of the a-parameter:
   Amplitude (≤ 1) period phase

   The a-parameter is periodic according to

   a_param(year) =standard_a_param(1.0+Amplitude* cos(2p*(Year-Phase)/Period))*

   The phase is the actual year where the cosine function has its maximum, and is not linked to the start of the recruitment regime

4. **Fourth recruitment line:** Spasmodic recruitments:
   Distribution code Mean_interval Sigma _interval, Magnitude _factor, First_spasmodic_year

   The first spasmodic year is relative to the year of regime change.
   Distribution code:
   1: Lognormal intervals,
   *2: Binomial intervals (the sigma is dummy), Prob(x years) = 1/mean * (1-1/mean)* ^{x}
   *3: Boxcar distribution of intervals – the sigma parameter is used for the relative range: The interval is drawn randomly in the range (mean-sigma*mean, mean+sigma*mean), rounded to the nearest integer*

Weight and maturity properties

The following 5 lines must be in the sequence below, the line number depends on the number of recruitment blocks:

- B0: Reference biomass for density dependent annual effects on weights and maturities
- R0: Reference recruitment for density dependent year class effects on weights and maturities
- Sigma for normally distributed yearly noise multiplier on weights and maturities.
- Amplitude (≤ 1), period, phase for periodic trends in weights and maturities
- 3 parameters for year class effects on weights and maturities (see population model description): Lower_bound, Upper_bound, alpha

Weights and maturities at age
Two options - input or parametric model

- Code for biological data option: i: Input data; or p: Parametric model

i-option:
There shall be 5 blocks of lines (one block for each bullet point below, within each block there is one line per age. The age range must be in accordance with the age range specified in line 3:

- Natural mortality: Age Value
- Selection at age: Age Value
  Note that the selection is normalized internally in the program, so the scaling does not matter.
- Weight at age in the catch: Age Standard_value Lower_bound Upper_bound Slope (alpha-value) Sigma for age-specific noise
- Weight at age in the stock: Age Standard_value Lower_bound Upper_bound Slope (alpha-value) Sigma for age-specific noise
- Maturity age delay factor: Age Standard_value Lower_bound Upper_bound Slope (alpha-value) Sigma for age-specific noise

  The Lower bound, upper bound and slope refer to the density dependence of weights and maturities (see . With the i-option these are specified for each age.

p-option:

One line for each bullet point:

- Natural mortality (assumed equal for all ages)
- Length at infinity (meters)
- Length at entrance (at youngest age)
- von Bertalaffy k
- Foulton condition factor ($W = cond*L^3$)
- Slope of maturity at age logistic function (Length with 50% maturity is set automatically at 2/3 of Linf)
- sel50:Length at 50% selection in the fishery
- slopesel *:* Slope of selection at age
  The selection is computed according to the length at age as a fraction of the selection at Linf:
  Selection(age)=(1.0+exp(-slopesel(Linf-sel50)))/ (1.0+exp(-slopesel*(Length(age)-sel50)))*
- Lower bound, Upper bound , alpha (slope): Density dependence parameters for weights and maturities (common to all ages). The bounds are relative to the standard value.

Example of a bio.inn file with the i-option:

```
# Blue whiting: Data from WKPELA 2012; taken from http://130.226.135.24/bluewhiting/
2011        Year 0
1 10 First and last age
3 7   Age range for average F
1    First age in TSB
0.0 0.0   Fraction of F and M before spawning
#Recruitment model specification
#1 1 8 16 1=One regime
4 1 8 16 23 4=Four regimes, 1=fixed years, shift in year 8, 16 and back in 23
# 3 2 7.0 -5 Tre regimer, stokastiske år, 7 års middelintervall, start tellingen for 5 år siden
# Medium year classes, spasmodic
1   1  2 Recruitment model distribution truncation_type
7340.0 1500.0 0.25 0.5  1.5 1  A-par, B-par, CV, trunc_lower, trunc_upper,  Depensatory gain, >=1
0.0 30 -15 Recruitment periodicity: Amplitude (0 is neutral), period, phase
3 6.2 0.6 3.66 -3  Spasmodic recruitment (Dist. type, interval, range interval, Factor (1=neutral), first year).
# Very large year classes
1   1  2 Recruitment model distribution truncation_type
38939.0 1500.0 0.35 0.2  2.0 1  A-par, B-par, CV, trunc_lower, trunc_upper,  Depensatory gain, >=1
0.0 30 -15 Recruitment periodicity: Amplitude (0 is neutral), period, phase
2 3.4 0.3 1.0 -1  Spasmodic recruitment (Dist. type, interval, sigma interval, Factor (1=neutral), first year).
# Recruitment failure
1   2  2 Recruitment model distribution truncation_type
5814.0 1500.0 0.38 0.2  2.85 1  A-par, B-par, CV, trunc_lower, trunc_upper,  Depensatory gain, >=1
0.0 30 -15 Recruitment periodicity: Amplitude (0 is neutral), period, phase
2 3.4 0.3 1.0 -1  Spasmodic recruitment (Dist. type, interval, sigma interval, Factor (1=neutral), first year).
# Medium year classes, spasmodic
1   1  2 Recruitment model distribution truncation_type
7340.0 1500.0 0.25 0.5  1.5 1  A-par, B-par, CV, trunc_lower, trunc_upper,  Depensatory gain, >=1
0.0 30 -15 Recruitment periodicity: Amplitude (0 is neutral), period, phase
3 6.2 0.6 3.66 -3  Spasmodic recruitment (Dist. type, interval, range interval, Factor (1=neutral), first year).
# Weights and maturities: Here common parameters, individual below
3000.0              B0: TSB corresponding to reference values (density dep)
20000.0             R0: Recruitment of reference year class (year class factor)
0.0                 Sigma for random year factor noise
0.0  50.0 -10.0     Amplitude, period and phase for periodic trends
0.5 2.0 0.0     Lower and Upper and Alfa for year class factor
```

Subsequent lines are specific for the i-option

```
# Code for biological data source
i
# Natural mortality at age
1 0.20
```

2 0.20
3 0.20
4 0.20
5 0.20
6 0.20
7 0.20
8 0.20
9 0.20
10 0.20
Selection at age
1 0.143
2 0.206
3 0.420
4 0.793
5 1.212
6 1.341
7 1.233
8 1.225
9 1.026
10 1.026
weight at age in the catch (average 2008-2010)
Note: To keep standard values fixed, set lower below 1 and upper above 1, and alfa = 0
1 0.0493 0.0 10.0 0.0 0.1 Standard value; Lower value; upper value; Alfa factor Sigma at age
2 0.0717 0.0 10.0 0.0 0.1
3 0.0950 0.0 10.0 0.0 0.1
4 0.1110 0.0 10.0 0.0 0.1
5 0.1258 0.0 10.0 0.0 0.1
6 0.1447 0.0 10.0 0.0 0.1
7 0.1617 0.0 10.0 0.0 0.1
8 0.1833 0.0 10.0 0.0 0.1
9 0.2086 0.0 10.0 0.0 0.1
10 0.2605 0.0 10.0 0.0 0.15
weight at age in the stock
Note: To keep standard values fixed, set lower below and upper above std. value, and alfa = 0
1 0.0493 0.0 10.0 0.0 0.1 Standard value; Lower value; upper value; Alfa factor Sigma at age
2 0.0717 0.0 10.0 0.0 0.1
3 0.0950 0.0 10.0 0.0 0.1
4 0.1110 0.0 10.0 0.0 0.1
5 0.1258 0.0 10.0 0.0 0.1
6 0.1447 0.0 10.0 0.0 0.1
7 0.1617 0.0 10.0 0.0 0.1
8 0.1833 0.0 10.0 0.0 0.1
9 0.2086 0.0 10.0 0.0 0.1
10 0.2605 0.0 10.0 0.0 0.15
Maturity at age
Note: To keep standard values fixed, set lower below and upper above std. value, and alfa = 0
1 0.11 0.0 10.0 0.0 0.0 Standard value; Lower value; upper value; Alfa factorSigma at age
2 0.40 0.0 10.0 0.0 0.0
3 0.82 0.0 10.0 0.0 0.0
4 0.86 0.0 10.0 0.0 0.0
5 0.91 0.0 10.0 0.0 0.0
6 0.94 0.0 10.0 0.0 0.0
7 1.00 0.0 10.0 0.0 0.0
8 1.00 0.0 10.0 0.0 0.0
9 1.00 0.0 10.0 0.0 0.0
10 1.00 0.0 10.0 0.0 0.0

Example of lines specific for the p-option

p
0.2 Natural mortality
Growth and maturity
1.0 Length at infinity (meters)
0.05 Length at entrance
0.25 von Bertalaffy k
8.0 Foulton condition factor
5.0 Slope of maturity at age function (50% is at 2/3 of Linf)
#Selection at age

4.0 Age at 50% selection
5.0 Slope of selection at age
Parameters for density dependent weights and maturities Lower bound Upper bound alpha
0.5 1.2 0.5

3.2.2. Run options (File opt.inn):

This file contains all specifications for the harvest rule, and some other options. Many parameters are stated with a lower, upper value and step value. The program will scan over these values. Setting the upper equal to the lower means only this value is used. The interval should not be zero.

General specifications

Line 1: Code for initial stock numbers at age. Options (see examples below):

1. Priming with a fixed F
2. Reading stock numbers at age and variance-covariances from a file (expects a *nin.inn*-file)
3. Reading stock numbers and CVs from a file (expects a *nin.num*-file)
4. Reading stock numbers from a file (expects a *nin.obs* - file), and apply the observation model to these stock numbers.

Line 2: Priming value for F (if initial numbers option is 1) or value of TAC in year 0 (if initial numbers option is 2, 3, or 4)
Line 3: Maximum possible F (to protect the run if there is not enough fish for the TAC)
Line 4: Last assessment year :Delay between last obs. year and decision year (, normally a negative number)
Line 5: Interval between decision years
Line 6: Reference Blim. This value is used to calculate risk to Blim, but it is not used as parameter in the harvest rules.
Line 7: Youngest age to be included in the calculation of B+ (total biomass from the youngest age onwards) for use in harvest rules. It is independent of the age used for calculating TSB related to density dependent growth on the bio.inn file and reporting in the Yield per recruit routine

Definition of basis for decisions on primary TAC

Each Harvest rule has a number of bases (entries in the basis vector), different for the different rules.

Line 8:
Number of entries in the basis vector. The necessary number is related to the harvest rule.
Additional bases are calculated, but not used. Necessary numbers:

Rule 1,2 and 4:	2 bases
Rule 3 and 5:	3 bases
Rule 6:	4 bases
Rule 7:	1 base
Rule 8:	2 bases

Further lines: One for each basis element:
Element number, type first_year last_year calculation

Codes:
The first and last year are relative to the TAC year.

Type: Version 13_2 has 6 types:

1. SSB according to assessment
2. TSB according to assessment
3. Recruitment according to assessment
4. Recruitment survey independent of assessment
5. Biomass survey measuring SSB independent of assessment
6. Biomass survey measuring B+ independent of assessment

Calculation:

1. Mean
2. Minimum
3. Slope in linear regression (relative to the mean)

Stating the decision rule:

Next line - stating the decision rule to be applied:
Decision rule number, Measure

The Decision rule numbers are listed below with the parameters.

Measures:

- 1: F
- 2: HR (based on assessment)
- 3: HR (based on survey)
- 4: TAC

Following lines: Parameters in decision rule - one line for each parameter
lower, upper step:

For each parameter the line states a lower value, upper value and step for scanning. The meaning of the parameters is described in detail in Section 2.6.1. The sequence of the parameter lines described below must be followed.

Parameter lines by rule:

- Rule 1 (Alpha-rule):
  - Btrig1
  - Btrig2
  - Vstd
  - Alpha1
  - Alpha2
- Rule 2 (Alpha rule flat below B0):
  - B0
  - Btrig1
  - Btrig2
  - Vstd
  - Level1
  - Alpha2
- Rule 3 (Alpha rule with variable level):
  - Btrig1
  - Btrig2
  - Vstd
  - Alpha1
  - Alpha2
  - Gain
- Rule 4 (Alpha rule with smooth transitions):
  - Btrig1
  - Btrig2
  - Vstd
  - Slope1
  - Slope2
  - Vmax
- Rule 5(Alpha rule with variable vstd)
  - Btrig1
  - Btrig2

        - Refr
        - Alfa1
        - Alfa2
        - Vlow
        - Vhigh
    - Rule 6 (Alpha rule with variable vstd, flat below B0)
        - B0
        - Btrig1
        - Btrig2
        - Refr
        - Alfa2
        - Vlow
        - Vhigh
    - Rule 7 (Alpha rule with two slopes)
        - Btrig1
        - Btrig2
        - Refr
        - Alfa1
        - Alfa2
    - Rule 7 (Alpha rule with bounds)
        - Btrig1
        - Btrig2
        - Vstd
        - Alfa1
        - Alfa2
        - Lower bound
        - Upper bound

Definition of constraint rules.

There are 3 kinds of constraints. All must be specified in sequence on the file. If they do not apply, use parameters that disable them. For each constraint, the following are specified:

- One line with the
  constraint number.
  If the number is 2, an additional number is needed (1=Always, 2=Only)
- Then, lines with parameters:
    - For constraint number = 1 (filter):
        - Filter gain (lower, upper, step)
        - Trigger for derogation (lower, upper, step)
        - Basis for derogation:
          Type first_year last_year caclualtion
          Types and calculations are as for defining basis for the primary TAC, see above. Only one basis permitted
    - For constraint number = 2 (percentage):
        - Percentage (lower, upper, step)
        - Trigger for derogation (lower, upper, step)
        - Basis for derogation:
          Type first_year last_year caclualtion
          Types and calculations are as for defining basis for the primary TAC, see above. Only one basis permitted.
    - For constraint number = 3 (min-max)
        - Minimum TAC (lower, upper, step)
        - Maximum TAC (lower, upper, step)

Next lines – additional specifications of decision rules :

- Implementation of multiannual TAC (Abrupt=1, Gradual=2). Dummy if annual TAC.
- Gradual transition towards target: Number of years to reach target lower, upper, step).

Year factor bias and variances in observations and implementation

- Bias in the observations model (lower, upper, step)
- Bias in the implementation model (lower, upper, step)
- CV for the year factor in the observations model (lower, upper, step)
- CV for the year factor in the implementation model (lower, upper, step)
- CV recruitment survey (lower, upper, step)

Further lines: ***CV for noise in age factors***: (one line for each age, no more, no less):

Age CV_for_observation_noise_age_factor CV_for_noise_in_observed_catches_at_age CV_for_implementation_noise_age_factor

One line with coefficients for autoregressive model of observed stock numbers

Number_of coefficients, individual coefficient values:

Coefficients for autoregressive model of observed stock numbers

One line with the number of years for calculating mean recruitment in projection

Number_of_years

Following lines – prinout options:

Keys for printout options (one line for each file - the file names are hard-coded):

- First number: Internal code (don't change it!); Second number:1 = print, 0 = do not print.

The files are (with their internal code number for reference):

11	'results' :	Main results file
12	'Years y1 - y2': y1 y2	Averages and percentiles over the years y1 to y2. **Specify the years on the line**
14	'First_5' :	Averages and percentiles over the years 1-5
15	'traj':	Individual trajectories (first 20 replicas)
16	's0dist':	All SSBs in year null
17	'Fdist':	All F-values (by iteration and year)
18	'IAVdist' :	All IAV values (by iteration and year)
19	'todssb' :	All SSB-values (true,obs,assumed by iteration and year)
20	'srpairs' :	All stock-recruit pairs (by iteration and year)
21	'All_years' :	Averages and percentiles over all years
22	'Yieldrecr' :	Yield & SSB per recruit
23	'Catch_dist' :	Equilibrium age distribution in the catch (in yield per recruit)
24	'SSB_dist' :	Equilibrium age distribution in the SSB (in yield per recruit)
25	'Wema':	Yearly weights and maturities at age for the first 10 replicas
26	'Comprecr':	Compare true and assumed recruitments

The output files as explained further in the output section (3.3) below.

Example of an opt.inn file

```
# General conditioning
4     Initial data code: 1=priming, 2-4=data from nin-file
93.876    Priming F or Tac0
3.0    Maximum F-factor
-1    lastassn: Delay between obs. year and first TAC year.
1     idur: Interval between decision years
1500.0 Reference Blim
```

```
1 Youngest age for TSB
# HCR design
# Basis
2 Number of entries in the basis-vector
1 2 -1 1 1 Basis 1: 2=TSB, time -1 1, 1=avg
2 2 -1 1 1 Basis 2: 2=TSB, time -1 1, 1=avg
# Rule
1 3 Rule number 1=Alfa-rule,; 1=F, 2=HR-rule, 3=TAC
# Function number 1 has 5 function parameters, stated as min, max and interval
3000 4000 250  Btrig1
5000 6000 1000 Btrig2
400 550 50 v-standard (here TAC)
1.5 2.5 0.5 Alfa1
2.0 4.0 0.5 Alfa2
# Constraint rules: All rules 1-3 must be on the file.
1 Constraint rule 1 (filter)
1.0 1.0 0.5 Filter 1 (one year behind): 1.0: No filter
0.3 0.3 0.1 Trigger for filter derogation
2 -1 1 3 Basis for filter derogation: 2=TSB, time -1 1, 3=trend
# Constraint rule 2: Percentage
2 2 Constraint rule 2 (percentage), Option 2 'only' Two parameters, one basis
-5 -5 20 Percentage: Negative = ignored
2000.0 2000.0 1000.0 Trigger for percentage derogation
1 -1 1 2 Basis for percentage derogation: 2=TSB, time -1 1, 2=minimum in period
# Constraint rule 3
3 Constraint rule 3: max and min TAC: Negative: No bound
100 100 100 Minimum TAC
2500 2500 500 Maximum TAC
# Implementation of multiannual TAC
1      iabrgrad: (1: Abrupt, 2: Gradual) Dummy if interval=1.
# Gradual transition to target
1 1 4              Number of years to reach target (1 is from the outset)
# Obs model specification
0.0 0.0 0.2        Bias obs model
0.0 0.0 0.2        Bias implementation
0.30 0.30 0.04     CV year factor obs model
0.0 0.0 0.2        CV year factor implementation model
0.2 0.2 0.2        CV recruitment survey
0.6 0.6 0.1        CV biomass survey
# Age factors for uncertainty (obs,  catch and impl). Check that you have the right ages!
# Taken from sam.cor, as adviced by Jose de Oliveira. There is no documentation
# that these are the right numbers. Correlations are not included.
1 0.75 0.1 0.1          Age, cv agefact.obs model, cv agefactor catches, cv agefact.implementation
2 0.48 0.1 0.1
3 0.29 0.1 0.1
4 0.17 0.1 0.1
5 0.19 0.1 0.1
6 0.18 0.1 0.1
7 0.23 0.1 0.1
8 0.24 0.1 0.1
9 0.31 0.1 0.1
10 0.37 0.1 0.1
# Coefficients for autoregressive error in survivors
# Derived assuming F=M=0.2
5 0.3 0.25 0.20 0.15 0.1 Number to be read and values
3 Number of years for calculating average recruitment (0:use SR function instead))
11 1 print 'results' : Main results file
12 1 10 20 print 'Years_x_y' : Averages from year x to year y. Years specified
14 1 print 'First_5'  : Averages years 1-5
15 1 print 'traj' : Individual trajectories
16 0 print 's0dist' : All SSBs in year null
17 0 print 'Fdist'        : All F-values (iteration and year)
18 0 print 'IAVdist' : All IAV values (by iteration and year)
19 0 print 'compbas' : True and assumed basis, but only for the TAC year and the first 100 iterations
20 0 print 'srpairs' : All stock-recruit pairs
21 1 print 'All_years' : F, Y and SSB in the years 6 to 15
22 1 print 'Yieldrecr' : Yield & SSB per recruit
```

23 0 print 'Catch_dist' : Equil. age distr. in catch
24 0 print 'SSB_dist' : Equil. age distr. in SSB
25 0 print 'Wema': Weights and maturities at age
26 0 print 'Comprecr': Compare true and assumed recruitments

3.2.3. Files with initial numbers (nin-files - optional)

There are several options for input of initial numbers. The input file depends on the option stated in the first line of the ***opt.inn*** file. The program expects a file with the right name and contents. If the option is 1, no file of this kind is needed.

- The first number on each line is the age. The second number is the stock number at age. The rest of the line depends on the options below.
- If the youngest age is higher than 0, and the initial data are taken from a nin-file, SSBs for the years prior to the initial years are stated at the end of the file, starting with the earliest year. This is to have a basis for drawing recruitments for the year classes that were born, but not recruited at the start of the simulation.

(***nin.inn*** - corresponding to option 2)
Initial stock numbers at age, and a matrix with sigmas and correlations. The numbers at age are plain numbers, which are converted to logs internally by the program. The diagonal elements on the matrix are sigmas in a log-normal distribution, the other elements are correlation coefficients (see example below).
This format is adapted to taking values from ICA (ica.out for the numbers and ica.vc for the correlations)

Example - nin.inn

```
 0 3096.400  2.571  0.038  0.021  0.012   0.011  0.013  0.013  0.013  0.014  0.014  0.015  0.016   0.000
 1  666.910   0.038 0.250  0.019  0.011   0.010   0.012  0.012  0.012  0.013  0.013  0.014  0.014  0.000
 2 1306.100   0.021  0.019  0.039  0.010   0.009  0.011  0.011  0.011  0.012  0.012   0.013  0.013  0.000
 3 1645.200   0.012  0.011 0.010  0.017   0.005   0.007  0.007  0.007  0.007  0.008  0.008  0.008  0.000
 4 3923.600   0.011 0.010  0.009  0.005  0.013   0.007  0.007  0.007  0.008  0.008  0.008  0.009  0.000
 5 1431.000   0.013  0.012  0.011  0.007  0.007  0.014  0.009  0.009   0.010  0.010   0.010  0.011  0.000
 6  245.570   0.013  0.012  0.011 0.007  0.007  0.009  0.014  0.009   0.010  0.010   0.011   0.012  0.000
 7  343.940   0.013   0.012  0.011 0.007  0.007  0.009  0.009 0.014   0.011  0.011  0.012   0.012   0.000
 8  174.690   0.014   0.013  0.012  0.007  0.008  0.010  0.010  0.011  0.015  0.012  0.013   0.013  0.000
 9  106.100   0.014  0.013  0.012  0.008  0.008  0.010  0.010  0.011  0.012  0.016  0.014   0.014  0.000
10   74.954   0.015  0.014  0.013  0.008  0.008  0.010  0.011  0.012  0.013  0.014  0.018  0.016   0.000
11   44.756   0.016 0.014  0.013  0.008  0.009  0.011  0.012  0.012  0.013  0.014 0. 016  0.020   0.000
12   58.783   0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.020
```

SSBs for previous years are not needed here since recruitment is at age 0 for the stock in this example.

(***nin.num*** - corresponding to option 3)
Initial stock numbers at age and CVs at age, assuming a log-normal distribution. The format is converted internally to the format of the nin.inn - file, with zero covariances and CVs on the diagonal of the variance-covariance matrix.

Format:

Age Number CV
.
.
SSBs for previous years, starting with the most distant year

(***nin.obs*** - corresponding to option 4)
Initial stock numbers at age only. To get random numbers later on, the observation model is used.

Format:

Age Number
.
.
SSBs for previous years, starting with the most distant year

Example of a nin.obs file. Here, since this is an example with youngest age >0, there are lines at the end with previous SSBs.

```
3     6000
4     3058
5    16441
6     6963
7    16602
8      700
9      730
10    2905
11    2922
12     377
13      80
500
500
500
```

3.2.4. File with historical catch numbers at age (canum.inn) - optional

Contains catch numbers at age for the years prior to the year defined as year 0. This file is needed by the observation model if years prior to year 0 are used in the decision process. It is assumed that the numbers represent ages starting at the age stated as the youngest age and ending with the age stated as the oldest age in the bio.inn file. The year sequence does not matter, and redundant years are ignored. The year is specified by the calendar years. Years not needed are ignored.

Format:

```
Year Number lowest age ..... Number oldest age
.
.
```

Example of a canum.inn file. In this case, the lowest ages stated in the bio.inn file is at 1, so the numbers start there. The oldest age is 10 in this example

```
2009  61 156 232 595 1596 1157 592 252  89 49
2010 350 223 160 209  646  992 703 257 707 44
```

3.3.Output.

The output is to files. All the file names are hard coded. The user specifies which of these files to produce in the opt.inn file (see above). In addition, one more file is generated which is a scratch file called 'kladd'. It is used mostly for dumping intermediate results during debugging, and the routine versions of the program leave it empty.

Some output files have one section for each run option (set of of rule parameters). Each section is headed with a listing of all options for that run. Summary files (***First_5, Years y1-y2, All_years***) have one line for each option, with all option parameters and some selected summary statistics for that option). These files are best read by importing them into a spreadsheet.

A third group of files come from the analysis of yield per recruit.

Note on risks.
Risk is defined here as the percentage of trajectories where the SSB falls below the value stated as reference Blim. On the results file, where yearly results are presented, the risk is the percentage in that year. On the summary files,

it is the maximum of the yearly percentages of throughout the period covered. This is also the case for the probability that Basis(1) is below Btrig. However, the prob crash is always the percentage of trajectories that have crashed in the year in question or earlier. In the summary files, it is the cumulated probability by the last year in the time interval.

Some headers on the files are somewhat cryptic and are explained here.

Results-file:
This is the printout of the main conditioning options and results (catch, F, SSB, recruitment and IAV). It hopefully is mostly self-explanatory. The percentage inter-annual variation IAV is defined as *[TAC(y)-TAC(y-1)]/[(TAC(y-1)+TAC(y)) /2]*100, i.e.* the change from one year to the next relative to the average in those years. The mean IAV in the top table is the mean absolute IAV, while the table with fractiles is the signed IAV. The probability of crashing the stock is the cumulated probability that either the stock is below 1/10 of Blim or that the decided catch would require a fishing mortality higher than the possible maximum.
There is one set of tables for each run option, and the conditioning is listed at the top.

First_5, Years y1-y2, All_years:
These files have one line for each scenario, with the conditioning for the scenario and mean values over the indicated period, and over all bootstrap replicas.
The first columns are the rule parameters, the actual parameters depend on the rule.

The results:

Cmean C10 C50 C90	Mean and 10, 50 and 90 percentiles for realized catch over all the years in the time period and all the bootstrap replicas.
Fmean F10 F50 F90	Mean and 10, 50 and 90 percentiles for realized fishing mortality over all the years in the time period and all the bootstrap replicas.
Smean S10 S50 S90	Mean and 10, 50 and 90 percentiles for SSB over the years in the time period of all the bootstrap replicas.
IAV:	Mean (over all years in the time frame and all bootstrap replicas) of the absolute inter-annual variation in percent: *abs{[TAC(y)-TAC(y-1)]/[(TAC(y-1)+TAC(y)) /2]*100}*.
Risklim:	See note above
Riskbreak:	The probability that the Basis(1) is below the first breakpoint in the harvest rule (Btrigger 1). Calculated as Blim.
Crash:	Probability that the agreed TAC could not be taken due to shortage of fish, or that the SSB was below 1/10 of Blim at any time before the end of the time frame

Traj-file:
Time trajectories of essential results. On this file some results are printed for all years for each bootstrap replica. Only the first 20 replicas are printed for each scenario. The variables are:

- TAC as decided
- Basis values as seen by the decision model (Basis 1 if there are several). It is the measure (SSB, TSB or Recruitment) that is presented for the actual year
- IAV as defined above
- True basis values, corresponding to the decision basis values above.
- True F
- Annual surplus production: TSB(y+1)-TSB(y) + TAC(y)

S0distr - ***file***: Initial SSB-values
This file lists all SSB values in year 0. It may be useful for looking at the distribution of initial conditions.

Fdistr - ***file***: Assumed and true fishing mortalities.
For every iteration and year, it prints iteration number, year, decision F and true F. The decision F is the F corresponding to the decided TAC and the observation model N-values.

IAVdistr - ***file***: Distribution of IAV
Iteration number, year, IAV-value in percent

Compbas: Comparing all the bases used for decision with the corresponding true values. If the basis is taken over several years, the value presented is for the TAC year. Only results for the first 100 iterations are presented.

Srpairs - ***file***: Stock-recruit pairs
Iteration number, year True SSB and the recruitment generated by that SSB

The files 'Yieldrecr', 'Catch_dist' and 'SSB_dist' are generated by the yield per recruit routine, as described below.

wema: Prints weights and maturities by age and year

Warning: Some of the files, in particular those in the lower part of the list, may become very large when screening over a large number of scenarios. They are intended to control distributions and ensure that the model is properly conditioned, and it is recommended not to use them in routine screening.

3.4. Yield per recruit

Yield per recruit is calculated routinely with the data on the bio.inn file, for a range of fixed *F*-values from 0 up to *Fcrash* (or to 1.0 if *Fcrash* > 1.0). The yield per recruit routine is run before any of the simulations. If only the yield per recruit is of interest, the run can be broken (CtrlC) when the first option run starts. If multiple recruitment regimes are specified, the calculations are repeated for each regime.

The results that are presented also take into account the stock-recruit relation and the dependence of weights and maturities on TSB. Hence, for each level of *F*, the equilibrium SSB and catch is presented, scaled to the equilibrium recruitment (according to the deterministic stock-recruit function) and weights and maturities. If a pure Yield per recruit calculation is wanted, make a run where the b_parameter in the stock-recruit function is set to zero and there is no density dependence of weights and maturities.

Periodic fluctuations in recruitment are not taken into account, the *a* and *b* parameters in the stock-recruit function are used as they stand. However, if spasmodic recruitment is stated, the scaling is to the long term average recruitment. Depensatory recruitment is not taken into account - the power term is set to 1 in this routine. Density dependent weights and maturities are accounted for - the equilibrium weights and maturities are according to the equilibrium TSB, but year class effects and time trends are not taken into account.

The a-parameter in the stock-recruit relation is adjusted to take asymmetry in the recruitment distribution induced by truncation into account. This is done by integrating the probability density function of recruitments over the interval specified by the truncation, and take the mean as sum of products of probability and stochastic variable. As a diagnostic on truncation, a stock-recruit curve can be plotted with the SSB and recruitment are that are printed for each F-level, and compared with the stock-recruit relation originally stated.

In addition to yields, SSB and TSB, the mean age of the catch and spawning stock are calculated as the sum of age_times_biomass of fish at age in the catch or spawning, divided by the total catch or SSB.

A search is made to find *F0.1* where the slope of the yield curve is 1/10 of the slope at the origin. An approximate *Fmax* is at the F where the catch is largest. Since the scanning is stopped at *Fcrash*, the highest catch may be just there. If *F0.1* is above *Fcrash*, -1 is returned. Note that F0.1 and Fmax refer to the production curve, taking the

stock-recruitment and the density dependence into account.

The output is on 3 files if asked for:

22 'Yieldrecr': Yield & SSB per recruit, adjusted for the stock-recruit relation:
On top, the input data are listed.
Then for each F:
F-factor; Yield SSB; %of virgin biomass; Y/SSB; Mean age of catch;
Mean age of the spawning stock; Recruits

In addition, F0.1 and Fmax are printed on this file, with values of F, Yield and SSB.

23 'Catch_dist': Equilibrium age distribution in the catch
24 'SSB_dist' : Equilibrium age distribution in the spawning stock

3.5. Screen output.

When the program is started, F0.1 and Fmax, with corresponding catch and SSB are displayed.
For each option run, all option parameters are listed (sequence as in the input) when each option is started. If file type 21 has been asked for, some average results for all years are displayed when the run is done. This is mostly to allow the user to monitor the progress.

4. Some practical advice:

1. It sometimes happens that the program crashes when reading the input files. The most common error is additional or missing lines at the end of the file, for example missing historic SSBs at the end of the nin.xxx when the recruiting age is higher than 0, or simply empty lines at the end of a file. Other common causes are missing lines, missing spaces or too few entries on some line. If you change the number of ages, make sure that all ages (and no more) are represented on the input files. In particular, the number of lines specifying observation noise at age in the opt.inn file must be adjusted. Fortran is extremely sensitive to such things, it takes everything literally and never understands what we really mean to say. And if you get frustrated because the program will not read your files, you are not the first one with that experience.

2. Some error messages refer to the line in the program where the error occurs. It may sometimes be useful to look up that line in the program code, just to see what goes on there. When running under Windows, it is useful to start the program in a dos-widow, else the error messages disappear immediately.

3. It is very easy to make mistakes when setting up the run options, weight and maturity models and stock recruit functions. **Never trust that options on an old file are** OK **if you don't quite understand them. Read the definitions, use the manual, check every line carefully, and use the output opportunities to control that you get what you want.**

4. Be careful with the distribution and truncation of recruitments. Asymmetric limits give a skewed distribution. A useful procedure is to take recruitments from the file 'srpairs' (number 20) into a spreadsheet, make a cumulated distribution and compare that with the cumulated distribution of historical recruitments that you want to reproduce, Often, some trial and error is needed to get it right.

5. Resist the temptation to scan over everything. The best advice is to be problem-oriented and systematic. With numerous options, the number of runs becomes very large, and even if the program is fast (each option will normally take 10 - 30 seconds on a modern PC) hundreds of options do take time. Afterwards, you will experience that presenting this vast amount of information in an understandable way is a major challenge.

6. All output file names are hard coded, and all files are over-written without warning. Orderly book-keeping saves much work, waiting for re-runs and frustration.