Atlantis User's Guide
Part II: Socio-Economics

Asta Audzijonyte, Rebecca Gorton, Isaac Kaplan,
Jacob M. Kasper, Elizabeth A. Fulton

April 2017
Current Update: June 2025

TABLE OF CONTENTS

NOTE. The table numbering begins within 1 in each part of the manual, there is no continuity of numbering of figures and tables across the two documents even though the chapter numbers flow from one document to the other. This will be updated in future versions of the manual when the editors figure out how to drive Microsoft Word with more grace.

Tables

NOTE. The figure numbering begins within 1 in each part of the manual, there is no continuity of numbering of figures and tables across the two documents even though the chapter numbers flow from one document to the other. This will be updated in future versions of the manual when the editors figure out how to drive Microsoft Word with more grace.

Figures

15. FISHING MORTALITY

15.1 General description of the fishing mortality

Atlantis currently has three main ways to apply fishing mortality:

1) imposed catches forced through time series files, providing catch biomass that should be taken from a stock;
2) user set a fisheries induced mortality rate, defining a proportion of biomass to be harvested per day;
3) dynamic fishing based on an effort matrix (days of fishing per fishery) provided by the user and modified according to a range of management and economics options.

Only one of the three options can be applied for a given functional group - fishery combination. However, one species can be harvested through all three options, if they are operated by different fisheries.

The first option of imposed catches is executed via externally provided catch files (a form of forcing file). The content of these time series files are created from information on realised catches collected by fisheries scientists. The fishing mortality these catches represent is dynamically calculated in Atlantis based on the standing stock, fisheries parameters (and in some cases aspects of the life history of the groups involved) which define the distribution of catches across ages and boxes, and possible management actions (further details are provided in the following section). In principle fisheries discards or fishing based on dynamic effort can also be imposed using similar forcing files. In practice, however, imposed catches are usually used with historical catch time series.

The other two options for determining fishing mortality do not use external catch forcing files, but produce realised catch biomass as an output. This output can then be compared with the existing data during the model calibration stage. The main difference between the user set fishing mortality and dynamic fishing is in the final mortality level applied for a group. For option 2 the fishing mortality (e.g. 40% of biomass per year) is set by the user, and the actual catch biomass will depend on the species abundance and fishing parameters. In dynamic fishing the user does not set the fishing mortality but effort (in days) applied by each fishery in each box (or the economic rules that in turn dictate the realised effort applied). There are many ways to allocate this effort to fisheries, starting from simple prescribed effort per quarter to dynamic economically based effort allocations that simulate fisher behaviour. Once the effort per fishery is calculated the biomass of each species caught will depend on the parameters defining swept area by the fishing gear, fish catchability, selectivity of the gear, as well as vertical and horizontal overlap. These additional parameters (catchability, selectivity etc) are required in the effort model, but some of them can also be used in other fishing options too. For example, a selectivity curve could be used in conjunction with the user defined fishing mortality rate (option 2).

In addition, for aquaculture species (defined as isCultured is *functional_groups.csv* file) Atlantis runs specialised aquaculture harvesting routines, which are almost identical to the user defined fishing mortality option.

The routines calculating catch and discards are called from the *Water_Column_Box()* and *Epibenthic_Box()* routines before the execution of ecological processes (this means groups that live in the sediment layers deeper than the surface layer can not currently be directly harvested). **The harvest routines are called only if a species (functional group) is identified as isImpacted (=1)** in the *functional_groups.csv* file. The main routine that controls fisheries is *Harvest_Do_Fishing_And_ByCatch()* in **atHarvest.c**, which then calls two specific routines to get catch *Get_Catch()* in atHarvestCatch.c and discards *Get_Discards()* in atHarvestDiscards.c

NOTE!

What kind of fishing mortality to use?

When data on catches is available, modellers often use imposed catch to aid the model parameterisation. If the main question is to understand the ecosystem dynamics given the specified catch biomass that the fisheries want to take, then the imposed catch option may be the best way to go.

The choice between the user defined fishing mortality and dynamic fishing is determined by whether the user is mostly interested in the biological dynamics given a set fishing pressure, or whether fishery development and socio-economic aspects are of more interest. Of course, the actual fishery is never just a fixed annual pressure and biological dynamics will be determined by complex fisher behaviour and a range of socio-economic aspects that will determine the actual fishing pressure. Yet, all models are just a simplified version of reality and the level of simplification is determined by the questions that are to be addressed

Table 1. General fisheries parameters

Parameter	Description
flag_fisheries_on in run.prm	Flag indicating that the Harvest submodel should be loaded. Must be set to 1 when applying fisheries
isImpacted in functional_groups.csv	Flag indicated that the group is impacted by fisheries and other human activities. Must be set to 1 for any group that can be impacted through fishing, bycatch or incidental fishing mortality (e.g. where benthic habitat is crushed by fishing gear)
YYY_tStart	Day of the model run when a fishery YYY starts operating
YYY_tEnd	Day of the model run when a fishery YYY ends operations
flagYYYday	Period of activity in fisheries YYY. 0 – fishery only operates at night, 1 – fishery operates during the day, 2 – fishery operates all the time. Note, this has implications for scaling of imposed catch (see chapter 15.2.1). Species activity (set in flagXXXday in *biology.prm*) files does not affect its availability to fisheries – an inactive species will still be fished.
flagfishXXX	Flag indicating whether a species XXX is actively fished. This flag is similar to isImpacted in the .csv file, and is partly the legacy of earlier

	development. Currently flagfishXXX indicates that a species is directly affected by fishing, whereas isImpacted includes also other impacts, such as bycatch or incidental mortality. The flagfishXXX should be set to 1 for all fished species.
flagfinfish	A global flag indicating that fisheries are operating. Turning on fisheries through flag_fisheries_on in *run.prm* means that Atlantis will apply either direct fishing (flagfinfish=1) or incidental mortality (flagincidmort=1). So if flagfinfish =0, incidental mortality can still occur through bycatch if flagincidmort=1, meaning a species might still be affected by fishing activities.
flagincidmort	A flag indicating whether fisheries can cause incidental mortality through bycatch. The bycatch will be determined depending on how a fishery is set: if imposed catches are used or user defined fishing mortality is set species in the model then bycatch of other species can be generated with the FCcocatchXXX parameter; whereas in dynamic fishing bycatch is determined by a range of parameters.
flag_access_thru_wc_XXX	Whether a fishery has access to fish in the entire water column (e.g. do trawl doors remain open as the gear moves up/down through the water column there by interacting with species who do not live at the depth the trawl is most actively being towed)
flaghabitat_XXX	Vectors indicate which habitat patchiness equation to use when undertaking dynamic fishing. 0=standard % overlap, 1=Ellis and Pantus based subgrid scale model
YYY_flagdempelfishery	This sets the flag for each group and fishery to show whether it is benthic or pelagic. 0 = pelagic, 1 = demersal. This will influence the vertical distribution scalar in dynamic fishing when the box is in shallow water (so there are less than the maximum number of layers); in this case the vertical distribution is contracted with a bias to bottom water column layers if defined as demersal or to surface layers if defined as pelagic.
k_mismatch	Reduction in effectiveness of gear due to mismatch in water column distribution of gear and vertebrates. This positional scalar downscales the availability of target species if they are marked as demersal (pelagic) and the gear is marked as pelagic (demersal).
habitat_YYY	Array of values indicating if fishery is excluded from certain habitats (0)
YYY_mindepth	Minimum seafloor depths fishery will act over (so if won't fish shallower than 1500m put 1500 here)
YYY_maxdepth	Maximum seafloor depths fishery will act over (so if won't fish deeper than 1500m put 1500 here)

15.2. Setting up fishing mortality through imposed catch

The imposed catch is handled by the *Get_Imposed_Catch()* routine in **atHarvestImposedCatch.c**

The imposed catch requires externally provided catch time series (in TS files, see chapter 8) and is

activated when flagimposeglobal in *harvest.prm* is set to a value > 0 (value of 0 means no imposed catch) and flagimposecatch_XXX is > 0 for at least one species-fishery combination. The parameter flagimposecatch_XXX is entered as a vector per species (XXX) with as many entries as there are fisheries in the *fisheries.csv* file; the assumed order of fisheries in the vector matches the order defined in the *fisheries.csv* file. Any value > 0 indicates which fishery has imposed catch. To date, for simplicity, all models have imposed catch for one fishery (which represents the aggregate catch over all fishery sectors being imposed in this way), which means that only one catch forcing file is required. In theory, Atlantis can apply different imposed catches for different fisheries, in which case the user should provide separate catch forcing files for each box and each fishery. However, this has not been applied in practice yet (so please contact the model developers if you want to try and use this multi-file option).

For age-structured groups and age-structured biomass pools the imposed catch will be age-specific. The value in the forcing file represents the total imposed catch and then the proportion of this forced catch to be taken from each age group is given by the CatchTS_agedistribXXX parameter vector, which has as many values as there are age group in species XXX (these values must sum to 1).

It is possible to apply imposed catch only for a certain time period of the model run. For example, the user might apply imposed catch for the first 10 years and then use other fishing options from year 11 on. The time period for the imposed catch is specified in imposecatchstart_XXX and imposecatchend_XXX. These parameters are entered as vectors, which have as many values as there are fisheries, and indicate the day of the model run that the imposed catch starts/ends for each species-fishery combination.

The flagimposeglobal and flagimposecatch_XXX can have **four different options,** which specify how the catch will be imposed:

1. Only global catch is imposed (flagimposeglobal = 1 and flagimposecatch_XXX = 1)
This option means that imposed catch provided in the single TS forcing files is the total catch per day for the entire distributional area of a species; and the catch imposed in any specific box is proportional to the biomass of the harvested species in that box versus the entire model domain. Note that in this case the *force.prm* file should still indicate one specific box for the imposed catch to facilitate read-in (see chapter 8), however it will not simply be used in that box but will inform catches in every non-boundary model box.

2. Imposed catch is box-specific (flagimposeglobal = 2 and flagimposecatch_XXX = 2)
This option requires a catch time series be provided for every box in the model where catch is to be extracted (if you want to be particularly careful and make sure that there is a time series supplied for every box then supply a time series of zeroes for any box where there is no fishing). Imposed catch is taken only from the boxes specified in the *force.prm* file. If the biomass of the species in the box is insufficient in the specified box, Atlantis will attempt to get the missing catch from other age groups of that species in the box (see Note! below). Any catch not taken is rolled over to the next day, then it is added to the forced catch (as defined by the forcing file) to be extracted on that day. If insufficient is available on that day it is rolled over to the next and so on until years end.

3. Imposed catch is box-specific but is allowed to take missing catch from the same stock (flagimposeglobal = 3 and flagimposecatch_XXX = 3)

This option is applied in the same way as the option 2 above, but if the biomass of the targeted species age-groups is insufficient in the specified box, Atlantis will take catch from other boxes inhabited by the same stock. Note, that first Atlantis will attempt to get the required catch by sampling other age groups in the specified box and only then go to other boxes. The orders of the boxes to be harvested for the missing catch corresponds to the box ID order, e.g. if not enough biomass is available in box 4, Atlantis will look in box 5, then in box 6 and so on, as long as these boxes have the same stock of the species, indicated in the XXX_stock_struct parameter in the *biology.prm* file. This sequential use of boxes to be harvested when attempting to make up the difference means the actual box fished for the extra may differ at each time step – for example, if after one month insufficient biomass is left in box 5 to supply the shortfall, then box 6 will be harvested until it is depleted and so on (assuming imposed catch remains higher than the biomass of harvested species in the specified box).
If, after all boxes have been fished for the extra, there is still insufficient biomass to take the catch then any remaining is rolled over to the next day, as for option 2.

4. Imposed catch is box-specific but is allowed to take missing catch from adjacent boxes (flagimposeglobal = 4 and flagimposecatch_XXX = 4)

This is applied as the option 3 above, but the missing catch is taken only from the neighbouring boxes. The orders of the boxes to be harvested for the missing catch corresponds to the order defined in the *ibox* vector for the box in the BGM file, e.g.

```
box2.nconn    8
box2.iface    5 6 78 76 75 74 73 4
box2.ibox     187 3 26 25 1 301 299 300
```

In this case the 8 neighbours of box 2 would be checked in the order box 187, 3, 26, 25, 1, 301, 299 and 300.

If, after all the neighbouring boxes have been fished for the extra, there is still insufficient biomass to take the catch then any remaining is rolled over to the next day, as for option 2.

NOTE!

How Atlantis supplements missing imposed catch from different age groups of a species

The CatchTS_agedistribXXX parameter provides proportions of imposed catch to be taken from each age group (e.g. 0 0.1 0.2 0.3 0.4 for a species with five age groups). This means that 40% of imposed catch is taken from age group 5, 30% from age group 4 and so on).

If insufficient biomass is available in a box to get the required imposed catch using the age distribution as stated, Atlantis will first attempt to get the outstanding catch by increasing the proportion it takes from the oldest age groups. It will first start by increasing the proportion of the catch taken from the oldest age groups (by rescaling the value given by CatchTS_agedistribXXX) and, if sufficient biomass still cannot be caught, then incrementally move down the age groups to the youngest fished age group (if an age group is marked as zero it will never be harvested as the rescaled proportion will still be 0).

Regardless of the options (1-4) used for imposed catch, if, at the end of the year, Atlantis has outstanding catch it could not take (i.e. any accumulated rollovers), it will give a warning message in

the *log.txt* file about the size of the mismatch between the imposed catch time series for the year and what could actually be taken (i.e. the outstanding amount) and will the zero all the missing catch and begin again for the new year, i.e. **it will not carry over missing catch into the next year.**

The imposed catch given in the TS files can be **further modified** to account for **underreporting, adaptive management** actions, **marine protected areas** (MPAs) and **fisheries activity** (day, night, both).

1. The **underreporting** implies that the actual catch is higher than what is known by the fisheries and provided in the forced catch files. The scale for underreporting by different fisheries for different species is provided in the reportscale_XXX parameter. It must have as many entries as there are fisheries, giving scalar values of underreporting for each fishery. If no underreporting is assumed then the value for the fishery should be set to 1.

2. A simple **adaptive management** scalar is applied if a fishery YYY is identified as managed, by setting YYY_flagmanage=1. The management of the fishery YYY starts on the day given in the YYY_start_manage and ends on the day given in YYY_end_manage parameters. It is therefore possible to only start the management in the middle of the model run. For example, if imposed catch is used for the first 10 years and then dynamic fishery start in year 11, the user may only want to apply management actions from the year 11. In this way the forced catch time series data will not be modified by the management scalars.

The management of the fisheries can many options (total allowable catch, or fisheries reference points), which are described in chapter 16. Briefly, all of these options return a scalar (typically ≤1) that will be applied to the original effort or imposed catch to calculate the actual catch taken and landed. Many of these alternative management options require YYY_flagmanage to be set to a value >1.

3. **MPA** or spatial management applies to a fishery YYY, set with YYY_flagmpa >0 and a global flag flagmpa=1. There are eight different options to apply spatial management, described in chapter 16.4, but briefly all of them will return a scalar for each box depending on whether the presence of an MPA in the box affects the fishing activity. The base spatial management scalar for each fishery in each box is set in MPAYYY, but it can be modified depending on the MPA options chosen. The MPA scalar is typically ≤1, with the value representing the fractional area of the box open to fishing. The value can be set to >1 for cases where spatial management actually leads to increased fishing activity (e.g. around the edge of an MPA). Infringement of spatial management is also possible - see 15.3.3.

4. Finally the imposed catch will be scaled by 0.5 if the fishery operates all the time rather than only during day or night (see box below).

NOTE!

Scaling of imposed catch due to fisheries diurnal activities

The imposed catch values assume that the fishery only operates for half of the day, i.e. during the day or night. This is set using the flagYYYday parameter (1if active by day or 0 if active by night). If the fishery has no preference, which means that it operates constantly then set flagYYYday=2 – in which case the imposed catch given in the times series file in mg s^{-1} will be halved.

If the scalars listed above are applied then it is likely that the actual catch taken from the box will be different from the imposed catch in the TS forcing files. The user should carefully think whether this is a desired outcome. If the user only aims to parameterise the model given the known catches in each box, or observe ecological dynamics given the imposed catch, then the management options should be turned off. On the other hand, by activating an MPA or management scalar it is possible to explore, for example, what the stock biomass might have been if certain management actions had been in place and modified the historic catches accordingly.
For the imposed catch the final catch H (mgN day^{-1}) to be taken by fishery Y from species CX age group i in a specific box j is calculated as

$$H_{CX,Y,i,j} = m_{CX,Y,j} \cdot p\_age_i \cdot repsc_{CX,Y} \cdot managesc_Y \cdot mpasc_{Y,j} \cdot activesc_Y$$

where $m_{CX,Y,j}$ is the total biomass (over all age groups) to be taken out from the box given in the forcing files (mgN day^{-1}); p_age_i is the distribution of forced catch over the age groups of species XXX (CatchTS_agedistribXXX); $repsc_{CX,Y}$ is the scalar for underreporting (reportscale_XXX) by fishery Y on species CX; $managsc_Y$ is the optional adaptive management scalar for the fishery Y (when YYY_flagmanage > 0, see chapter 16); $mpasc_{Y,j}$ is the optional scalar due to MPA presence in the box j (when YYY_flagmpa >0, see chapter 16.4); and $activesc_Y$ is the activity scalar (set to 0.5 if fishery YY is active during the day and night (flagYYYday=2)).

Example of the box-specific catch forcing TS file for two species

```
# Historical catch time series file until 2000 for box 2
#
## COLUMNS 3
##
## COLUMN1.name Time
## COLUMN1.long_name Time
## COLUMN1.units days since 1910-01-01 00:00:00 +10
## COLUMN1.missing_value 0
##
## COLUMN2.name FPL
## COLUMN2.long_name FPL
## COLUMN2.units mg s-1
## COLUMN2.missing_value 0
##
## COLUMN3.name FPO
## COLUMN3.long_name FPO
## COLUMN3.units mg s-1
## COLUMN3.missing_value 0
##
0  0.000000e+000 0.000000e+000
1  0.000000e+000 0.000000e+000
2  0.000000e+000 0.000000e+000
3  0.000000e+000 0.000000e+000
4  0.000000e+000 0.000000e+000
5  0.000000e+000 0.000000e+000
6  0.000000e+000 0.000000e+000
```

... list days of the run and the catch imposed

14654 5.886416e-004 5.316763e-004
14655 5.886416e-004 5.316763e-004
14656 5.886416e-004 5.316763e-004
14657 5.886416e-004 5.316763e-004
14658 5.886416e-004 5.316763e-004
14659 7.443469e-004 5.316763e-004
14660 7.443469e-004 5.316763e-004
... and so on

Note that this file gives the **total catch per box in mg per secon**d over all age groups of a species. The age distribution of the imposed catch is given separately in the CatchTS_agedistribXXX parameter.
It is not necessary to give catch for each day. The user can give data for set time periods (every month or every year) and then indicate via the typeCatchts parameter in *force.prm* whether to use the last valid value (typeCatchts set to 1) or to interpolate between the two provided values (typeCatchts set to 0). See "Format of the time series (TS) forcing files" box in Chapter 8.

Table 2. Parameters required for imposed catch setup

Parameter	Description
flagimposeglobal	If >0 imposed catches are applied for at least one species-fishery combination and a forcing TS file must be supplied (or else Atlantis will quit). Values of 1 to 4 indicate which option of imposed catch to use
flagimposecatch_XXX	A vector with as many values as there are fisheries (with the order assumed to match the order in the *fisheries.csv* file). If >0 imposed catch is applied for species XXX for that fishery. Values of 1 to 4 indicate which option of imposed catch to use
CatchTS_agedistribXXX	Proportion of the imposed catch to take from the age group (if there is sufficient biomass over all fished age classes to cover the imposed catch). A vector should have as many values as there are age groups in the species XXX and should sum to 1.0.
imposecatchstart_XXX	A vector with as many value as there are fisheries Day of the model run that the imposed catch starts for a species XXX and the specified fishery
imposecatchend_XXX	A vector with as many value as there are fisheries Day of the model run that the imposed catch ends for a species XXX and the specified fishery
reportscale_XXX	A vector with as many value as there are fisheries A scalar for imposed catches to scale from reported catches to total catches when accounting for misreporting by each fishery. For example if there was an additional 25% of the catch that was not reported, so that total catch was equal to 125% of official reported catch, then the scalar should be set to 1.25 for that fishery
YYY_flagmanage	If >0, management options will be applied to the fishery YYY and imposed catch may be scaled by a management scalar.

YYY_flagmpa	If >0, the fishery YYY will be affected by MPAs and imposed catch will be scaled by the MPA scalar.

15.3. Setting up fishing mortality through user defined non-dynamic fishing mortality

An alternative way of providing fishing mortality values (and resulting catches) is to set fixed fishing mortalities (i.e. provide the rates as parameters). This is handled by the *Get_Fishing_Mortality()* routine in **atHarvestCatch.c**

This option will apply a fix mortality rate for each species identified with flagF_XXX and mFC_XXX parameters, with the user set options for distributing the mortality across age groups and/or sizes.

The general equation to calculate catch/harvest (mgN day^{-1}) of species CX age group *i* to be taken by fishery Y in a box *j* is then calculated as

$$H_{CX,Y,i,j} = mFC_{CX,Y} \cdot Biom_{CX,i,j} \cdot sel_{Y,i} \cdot managesc_{Y} \cdot mpasc_{Y,j} \cdot brok_{Y}$$

where $mFC_{CX,Y}$ is the fishing mortality rate (as a proportion of biomass day^{-1}) that fishery Y applies to species CX, $Biom_{CX,i,j}$ is the biomass of age group *i* in a box *j* (mgN); $sel_{Y,i}$ is the selectivity of fishery Y on age group *i* which can be defined as size based selectivity (where values will range from 0 to 1), or simply a minimum size that is caught (any individuals smaller than this will have $sel_{Y,i}$ set to 0 and any individuals larger then the minimum size will have $sel_{Y,i}$ set to 1); $managsc_{Y}$ is the TAC management scalar for the fishery Y, which takes a value of 1 if the relevant TAC has not been reached and 0 if the TAC has been reached (or exceeded) and flag_stop_F_tac =1; $mpasc_{Y,j}$ is the optional MPA scalar applied to fishery Y in a box *j* (when YYY_flagmpa >0, see above and chapter 15.6); and $brokst_{Y}$ is the optional broken stick management scalar for the fishery Y (see below).

15.3.1. Defining the fishing mortality level

To select the option for fishing mortality for species XXX set flagF_XXX to 1 for at least one fishery (the parameter vector must have as many values as there are fisheries) and give a daily mortality rate for species XXX for that fishery in the mFC_XXX parameter vector. The fishing mortality values given in the mFC_XXX apply to the whole species, which means that the same proportion of biomass will be harvested in each box where the species is found (subject to spatial management options noted below).

VERY IMPORTANT NOTE!

Thanks to Shane Richards for helping with the following equations

Setting up the fishing mortality mFC_XXX parameter

The mFC_XXX parameter in the *harvest.prm* file must provide the daily probability of being caught – in effect the proportion of biomass to be harvested in one day. To set these values correctly it is important to understand the difference between the **harvest rate** and the **probability of being caught**. In fisheries, these two options are referred to as **instantaneous fishing mortality rate F**

over the time period *t* (where t is usually one year) and the **discrete exploitation rate μ** (which is the proportion of population harvested over the time *t*).

The instantaneous rate F over the time period *t* can be >1 and is NOT the same as the proportion of the population harvested (μ). They relate to each other as

$$F = -(1/t)\ln(1-\mu) \quad \text{or} \quad \mu = 1-\exp^{-Ft}$$

The discrete rate μ can only range from 0 to 1. The corresponding annual harvest rate F ranges from 0 to a very large number. Over a single year (t=1), if μ=0.5 then F=0.7, if μ=0.99 then F=4.6.

In Atlantis, the mFC value in the parameter file is treated as a **daily probability of being caught** (daily μ). When parameterising this, many people look to annual F rates, which are often available from stock assessments and can be translated to a daily mFC value (to be entered in the mFC_XXX parameter) as

$mFC = 1-\exp^{-F/365}$, where F/365 refers to daily F value

So if the user wants to apply an annual fishing mortality F=1.5 $year^{-1}$, it means the daily value is
$mFC=1-\exp^{-1.5/365} = 0.004101$

Alternatively if the user has annual harvest probability values μ (which can only range from 0 to 1), the daily probability to be entered is
$mFC=1-(1-\mu)^{1/365}$

This is derived assuming that mFC is the daily mortality probability, which means that (1-mFC) is the daily survival probability, and $(1-mFC)^{365}$ is the annual survival probability. This leads to the annual catch (death) probability being $\mu = 1-(1-mFC)^{365}$

In the case of the example above, the value of F=1.5 $year^{-1}$ corresponds to μ=0.77687 $year^{-1}$ and the daily value to be provided to Atlantis is
$p=1-(1-0.77687)^{1/365} = 0.004101$

Finally, in Atlantis the **mFC_XXX value refers to the probability of the entire age group of a species being caught.** The actual proportion caught will be smaller or even zero once selectivity or minimum age restrictions are applied. This is different from the F value in stock assessments which typically indicate harvest rate of individuals that are already recruited to the fishery, or the harvest rate of fish after applying fisheries selectivity.

15.3.2. Age or size selectivity when using a specified fishing mortality

The distribution of catch **across age groups** can be controlled in **two exclusive ways** (only one of them can be chosen):

1) **Set** up the **minimum and maximum ages at which fishing pressure applies**. This can be done only for age-structured groups and age-structured biomass pools and is applied using the XXX_mFC_startage and XXX_mFC_endage parameters, which gives the youngest and oldest age group of species XXX for which fishing pressure applies. Once the minimum and maximum age group is set, the **same fishing mortality rate is the applied to all age groups that are older than this minimum age and younger than this maximum age.**

NOTE!

When setting parameters defining the first age group remember that Atlantis counts from 0

The value in the parameter file should reflect the fact that Atlantis (or C++) counts from 0. This means that if the user wants fishing to affect the third and older age groups, the correct setting for XXX_mFC_startage is 2. The same applies to other parameters, such as maturation age set in the *biology.prm*.

2) **Size based selectivity**, applied by setting flag_sel_with_mFC=1. The shape of the selectivity curve to be used by each fishery is set by the YYY_selcurve parameter - see chapter 15.8 for details on the different selectivity curve options. Depending on the selectivity curve chosen, the user will have to provide relevant parameters defining the shape of the curve (see chapter 15.8).

NOTE!

Selecting the correct form of size selectivity for different species

Only one selectivity curve can be applied for a given fishery (there is no way to currently allow for different selectivity curves per species captured by a fishery). This curve is indicated by the YYY_selcurve parameter. It is important to remember that if a fishery with a specified selectivity curve targets different species of different sizes (i.e. has non zero flagF_XXX parameters for multiple species), some of the species may not actually be caught if they are too small (i.e. if the selectivity curve for returns a selectivity of zero for fish of that size), for example. For cases where fishing mortalities are applied through the mFC_XXX parameter and the use of the size selectivity option (flag_sel_with_mFC=1) is leading to biomasses much less than intended (due to selectivity issues) then it may be a good idea to split this single fishery into multiple parts – effectively using different fisheries for harvesting species of different sizes and making sure that the selectivity function chosen is reasonable for those size fish.

15.3.3. Other optional factors that can affect user defined fishing mortality rates

The fishing mortality can be further modified (typically reduced) if **total allowable catch (TAC)** management is applied, **MPAs are present** or/and a **broken stick management scalar** is applied:

1) The **TAC and weekly catch limit management**

The initial values for the TAC per fishery for a species XXX are set in the TAC_XXX parameter (in **tonnes** of catch per year).

When flag_stop_F_tac is set to 1, the fishing completely stops once the TAC is exceeded OR weekly catch limit is reached – regardless of any management options in place! If the flag_stop_F_tac = 0, the fishing continues even after exceeding TAC and weekly catch limits, and management options determine what to do with the catch that exceeds these limits. If there no management in place or a fishery is not managed through TACs (YYY_flagmanage < 2) the catch above the TAC will be retained as normal catch. Alternatively, if management is in place and a fishery is managed through TACs (YYY_flagmanage > 1), all catch above TAC will be discarded. Likewise, flag_stop_F_tac = 0, all catch above the weekly catch limit will be discarded. If you do not want any TAC related restrictions make sure you set TAC_XXX to 10^{12}, TripLimit_XXX to 10^{12} and flag_stop_F_tac = 0. If management is in place and a fishery is managed through TACs (YYY_flagmanage > 1), then the TAC values will be updated dynamically depending on the management options (see chapter 16).

The values for the weekly catch limit are set in TripLimit_XXX parameter (**kilograms** of catch per week). These values are not updated dynamically through management.

If TAC_XXX or TripLimit_XXX for a given species-fishery combination are set to 10^{12} then Atlantis treats this species as no quota and no trip limit species for this fishery.

2) The **presence of MPAs** is applied in the same way as for imposed catch. First, set flags for appropriate fisheries YYY_flagmpa >0 and a global flag flagmpa=1. There are eight different options to apply spatial management, described in chapter 16.4, but briefly all of them will return a scalar for each box depending on how the presence of the MPA affects the fishing activity. Further the user can apply MPA infringement, set by the global flaginfringe=1 parameter and YYY_infringe parameters, defining the proportion of the total box area (not MPA area!) that will be fished due to MPA infringement.

NOTE!

Meaning of YYY_infringe parameter

It is important to understand the meaning of the MPA infringement parameter, as it has different meaning in different fisheries settings. For imposed catches or the case where a user defined fishing mortality is being used, if the YYY_infringe value is higher than the MPA scalar given in the box-specific MPAYYY parameter, the MPA scalar for that box is replaced with the YYY_infringe value (the infringe values are not box specific!). This means that the YYY_infringe scalar for the fishing mortality option does NOT represent the proportion of MPA to be infringed, but the proportion of the total box area to be affected by fishing.

So for example, the MPAYYY for a box is set to 0.5, meaning that half of the box is closed to fishing. If no infringement is applied, the fishing mortality in that box is scaled by 0.5. If infringement is allowed,

however, and YYY_infringe is set to 0.6, for example, then the fishing mortality in this box is instead scaled by 0.6.

3) The **broken stick scalar** is applied if broken stick management action option is used. The broken stick scalar applies Australian-style harvest control rules where fishing mortality is scaled down or completely closed if a stock goes below a reference limit point; this option is applied only if management is applied to a fishery that operates under the user defined fishing mortality option (YYY_flagmanage=1) and is further explained in chapter 16.3.

15.3.4. Temporal changes in the user defined fishing mortality

The user defined fishing mortality set through mFC parameter can **change during the simulation**. This is triggered by setting the global parameter flagchangeF=1 and setting flagFchange_XXX to 1 for the species and fishery combinations where changes in the F values are desired.

If changes in F are required for a specific species and fishery combination, then the number of changes for that species-fishery combination is given in XXX_mFC_changes. Note that while XXX_mFC_changes are at present specified per species-fishery combination, the actual specifics of each change (timing and magnitude) are only specified at the species level and so all the changes are applied to ALL fisheries that catch species XXX using the fishing mortality option and have XXX_mFC_changes > 1. If you require the changes to be specific to both the species and fishery please contact the Atlantis developers.

The day(s) of the model run at which the change(s) in F start is set in mFCchange_start_XXX parameter. This parameter should have as many entries as there are number of changes given in the XXX_mFC_changes parameter. The time period over which the change will occur is set using the mFCchange_period_XXX parameter, which again must have as many values as there are changes. This means that the change will not be instant but will be applied slowly (linearly interpolating from the original value to the new value) through the number of days indicated in mFCchange_period_XXX. Once the change period is finished the scalar on F does not return to the original value but is applied for the rest of the simulation unless another change is applied. The multiplier by which the ORIGINAL fishing mortality should be multiplied for each of the implemented changes is given in the mFCchange_mult_XXX parameter. This multiplier determines the factor that the original mFC value will be multiplied by to give the new (changed) F. If you are making multiple changes remember that these values multiply fishing by the original mFC value, not by whatever the fishing pressure was after the previous change.

Table 3. Parameters required to setup a user defined fishing mortality rate.

Parameter	Description
flagF_XXX	Vector indicating which fishery (set as 1) should apply a fishing mortality for species XXX
mFC_XXX	Fishing mortality rate for species XXX (per day). See note above on how to set correct rates
XXX_mFC_startage	First age class (counting from 0) of species XXX for which the fishing mortality should be applied
XXX_mFC_endage	Age class of species XXX where mFC is set back to zero.
flag_sel_with_mFC	Whether the fishing mortality should use size based selectivity (if yes set to 1). If the size based selectivity option is selected here, it will apply to **ALL** fisheries-species combinations operating through a fishing mortality (i.e. all fisheries-species combinations with flagF > 0 and mFC > 0 will have selectivity applied)
YYY_selcurve	Which size based selectivity curve the fishery YYY should use (see chapter 15.7)
flag_stop_F_tac	If set to 1, the fishing will stop if the TAC is reached. If set to 0, all catch above the TAC is discarded. Note, that for the fishing mortality case, if flag_stop_F_tac is set to 1 then the TAC and weekly Trip Limits will be applied irrespective of the YYY_flagmanage settings! If no quota limits are wanted, then both TAC and TripLimit parameters should be set to 10^{12} (see chapter 15.9.1).
flaginfringe	If set to 1, some fisheries will infringe MPAs. This flag is only relevant if MPAs have been setup through flagmpa and the YYY_flagmpa parameter, see chapter 15.6
YYY_infringe	The level of MPA infringement by fishery YYY (sets the minimum proportion of the box that is accessible, if an MPA is more restrictive than that then infringement occurs)
flagchangeF	Flag indicating whether mFC values will change for at least one fishery during the simulation
flagFchange_XXX	Which species and fishery combination for which F values change
XXX_mFC_changes	Number of changes for a species-fishery combination
mFCchange_start_XXX	Vector indicating when each change in fishing mortality begins in days of the model run. There should be as many entries as there are number of changes given in the XXX_mFC_changes
mFCchange_period_XXX	Period in days over which the change in fishing mortality will occur. There should be as many entries as there are number of changes given in the XXX_mFC_changes
mFCchange_mult_XXX	The multiplier by which the ORIGINAL fishing mortality is rescaled to obtain the new (changed) mortality rate. There should be as many entries as there are number of changes given in the XXX_mFC_changes See further details on the wiki.

15.4. General introduction to dynamic fishing

For simple questions on fishing-induced ecosystem changes, imposed catches or user defined fishing mortality serves as a good representation of the fishing pressure. However, many modellers want to explore the interaction between fishing and the ecosystem dynamically. In real life fishing is not constant through time, but changes in response to species abundance, distribution, fisher behaviour and fish price. Some of these options are available through dynamic fishing options in Atlantis. The dynamic fishing is based around the concept of fishing effort, how time and resources a given fishery is going to spend fishing in a given box. This imitates real fisher behaviour, where they cannot target specific species or exact fishing mortality, but only have control over the place of fishing and time spent fishing.

The fishing effort can be defined in a number of ways, ranging from simple constant (prescribed) effort levels to complex economically driven effort dynamics.

The definition of **fishing effort** and factors defining it **are different** from the **management rules** and are implemented independently. The goal of the fishing effort routines is to determine the amount of time each fishery (and its sub-fleets, if a fishery has several sub-fleets) will spend in each box of the model. The actual catch will then depend on the gear used by the fishery and its selectivity, fish abundance, distribution and catchability. Management in turn can limit this effort if some triggers, such as stock biomass or total allowable catch, have been reached. However, application of management is optional and does not affect the original definition of fishing effort.

The key aspect of dynamic fishing is in setting up the effort applied by different fisheries. There are many ways to define the effort, but they all return a matrix of days that each fishery will operate in each box. Once this matrix is available, the actual catch will be determined by the gear parameters (area swept and selectivity), vertical and horizontal overlap between the species and the fishery, and species parameters (catchability and escapement).

A general equation for the catch (mgN day^{-1} harvest) of a species CX caught by the fishery YY in a box *j* using the dynamic fishing option is:

$$H_{CX,Y,i,j} = \left(Eff_{Y,j} \cdot gear_Y \cdot sel_{CX,Y,i}\right) \cdot a_{Y,j} \cdot \left(\delta^Y_{depth,CX,i} \cdot \delta^Y_{pos,CX,i}\right) \cdot \left(\delta^Y_{habitat,CX,i} \middle| q_{CX,Y}\right) \cdot \left(Biom_{CX,i.j} \cdot \left(1 - p\_esc_{CX,Y}\right)\right)$$

where $mEff_{YY,j}$ is the effort applied by the fishery YYY in the box j (days day^{-1}) (with 14 different options available to set it up, see chapter 15.5); $gear_{YY}$ is the swept area of by gear of fishery YYY calculated as proportion of total volume of the cell (YYY_sweptarea) (day^{-1}) the fishery can "sweep" (for static gear that just sits on the bottom, such as lobster pots, this does not represent active movement by the gear but animals moving to and envountering the gear); sel_i is the selectivity of fishery YYY on age group *i* of species CX (ranging from 0 to 1, see chapter 15.8); $a_{YY,j}$ is the optional management scalar on fishery YYY in box *j* that can apply temporary or permanent reductions in effort (see chaper 16); $\delta_{depth.CX,i}$ is the depth (vertical) overlap between the species CX age group *i* and fishery YY; $\delta_{pos,CX,i}$ is the postitional availability scalar that accounts for an optional mismatch between the pelagic and demersal distribution of species and the fishery (see below); $\delta_{habitat.CX,i}$ is the habitat overlap between the species CX age group *i* and fishery YY (see below); $Biom_{CX,i,j}$ is the biomass of the species

CX age group *i* in a box *j* (mgN); $q_{CX,YY}$ is the catchability of species CX by the fishery YYY (0 to 1) conditional on the habitat refuge scalar used (see chapter XYX); and $p\_esc_{,CX,YY}$ is the proportion of biomass of species CX in the swept area that escapes the fishery YY (0 to 1, see below).

The first three terms in the equation describe fisheries parameters, the $a_{YY,j}$ is the management term, the further three terms describe the spatial overlap, and the last term describes species parameters.

The **proportion of biomass escaping the fishery** set in flagescapement_XXX for each fishery can be either zero (flagescapement_XXX=0), constant (flagescapement_XXX =1 and proportions given in p_escape_XXX), or size-based (flagescapement_XXX =2). Constant escapement just gives a proportion of escapement applied uniformly for each age class. The size based escapement is determined by the length of the age group or a species (see NOTE! below on how length is calculated in vertebrates and non-vertebrates) and parameters Ka_escape_XXX and Kb_escape_XXX (giving values for each fishery, so different escapement options can be set up for each fishery). In the size based escapement option the proportion of individuals in an age group escaping a fishery is calculated as

$$p\_esc_{CX,Y} = Ka_{esc,CX,Y} \cdot Length_{i,CX} + Kb_{esc.CX,Y}$$

Where $Ka_{esc,CX,Y}$ and $Kb_{esc,CX,Y}$ are Ka_escape_XXX and Kb_escape_XXX values for each species and fishery combination.

NOTE!

How is catch calculated in dynamic fishing?

It is recommended users look at the *Get_Dynamic_Catch()* routine in the **atHarvestCatch.c** to familiarise themselves with the dynamic catch algorithm. The text below gives the general overview of it.

When dynamic fishing is used, the amount of biomass that will be caught at each timestep is determined by the volume of a cell (one layer in a box) swept by the fishery gear and the biomass of the species in this cell that can be caught by the gear, given its selectivity.

If the fishery does not cause incidental mortality (flagincidmort=0), i.e. it does not take bycatch or cause habitat damage and only catches target species, then only the target species of a fishery (set in target_YYY) will be caught from the swept volume of the cell. If incidental mortality occurs, then the fishery will catch both target species and bycatch species available to that fishery (where the vector incidmort_XXX indicates which fisheries will catch species XXX as bycatch).

The volume of the cell swept depends on the fishing effort in the cell (calculated in one of the 14 options and measured in days of effort per day), depth (vertical) distribution of the effort (Effort_vdistribYYY), habitats that the fishery can access (proportion of the cell available to the fishery), positional availability and swept area of the gear (m^3 of water swept per one day of effort). So, for example, if 50% of the cell volume is available and being swept in each time step, then 50% of the biomass of all the target and bycatch species that happen to be in that cell can be caught. This proportion will be modified by the catchability (q_XXX), escapement (flagescapement_XXX) and gear

selectivity (sel_XXX and YYY_selcurve).

The **catchability** sets a proportion of the biomass (constant across all age groups, but potentially different between genetic stocks) of a species XXX that can be caught by each fishery. As the catchability is included in the habitat scalar calculations, if the "Ellis and Pantus" habitat scalar sub-models is used (flagbabitat_XXX=1) then it is also dependent on the other patchiness parameters used in the habitat scalar (see chapter 15.7.3). In case of a standard habitat scalar (flaghabitat_XXX=0) the catchability is a simple multiplier. In the example above, if the catchability of the species is 0.5 then the fishery will only have access to half of the available biomass. So instead of 50% of its cell selected biomass, it will only catch 25% of it. The catchability parameter aims to account for fish behaviour and other factors that help fish to avoid fishing gear. It is NOT size or age dependent and works as a simple scalar on the biomass available to the fishery.

The proportion of caught biomass can be further modified by the **escapement**. All fish that escape the gear are assumed to survive in perfect health, they are not damaged by the gear in any way. In contrast to catchability, the escapement can be age or length specific. This allows the user to give different escapement probabilities and, for example, improve the survival of the largest individuals. In the example above, if the escapement of a selected age group is set to 50%, it means that only 12.5% of that age group will be caught in the cell.

Finally the caught biomass will also be modified by the **selectivity**. The selectivity can be constant or length dependent. The selectivity will act as a further scalar reducing the biomass caught. In the example above, constant selectivity of 0.5 would mean that the final catch is further scaled and only 6.25% of the biomass or age group (depending on how selectivity is set) is caught in the cell.

It might be easier to track the fishing outcomes if catchability or escapement are modified one at a time. So, for example, the catchability can be set to 1, but escapement can be set to a different age or length specific proportion. Alternatively, the user might want no escapement and modify the available biomass through catchability only. Finally, the simplest option would include no escapement and 100% catchability; then the available biomass of the target and bycatch species of fishery YYY, in the swept volume of the cell, would only be limited by the fishery's gear selectivity.

15.5. Defining effort in dynamic fishing

There are 14 different ways to set the amount of effort each fishery will allocate to each box. The effort is measures in days of effort that the entire fishery will fish per day (but for recreational fishery it is set as days per day by one fisher). Some options set effort as a constant through time, others allow the effort spatial distribution to change dynamically through time in response to catch per unit effort (CPUE), while one option sets effort on the basis of economic and market parameters, calculated in a separate Economics submodel. The effort options are selected with the parameter YYY_effortmodel and can be set differently for different fisheries. Note that the sequence number of the options does not necessarily reflect their complexity, but the order in which they were implemented in Atlantis. Therefore below they are not necessarily listed in the increasing order.

The fishing effort term $Eff_{YY,j}$ is calculated in *Allocate_Immediate_Effort()* in **atManage.c.** This routines

returns the effort per cell, which can then be scaled down in response to different management options (endangered species, MPAs, and so on, see chapter 16).

15.5.1. Constant effort per season (quarter) in each box (efformodel=0)

This option is set with YYY_effortmodel=0. It is based on two parameters that give the total annual effort across the entire model domain for each fishery (YYY_Effort, given in days per day per entire fishery), and the quarterly scalar of this average effort defining the seasonal (quarterly) changes (mEff_YYY, no units). This total effort is evenly distributed across boxes without **accounting for the area of the box.**

In this option the effort by the fishery Y in a cell j ($Eff_{Y,j}$ in days day^{-1}) in a given timestep is calculated as

$$Eff_{Y,j} = \frac{TotEff_Y \cdot \left(\varepsilon \cdot \left(Eff_{Q+1,Y} - Eff_{Q,Y}\right) + Eff_{Q,Y}\right)}{Nbox}$$

where ε is the proportion of the quarter that has passed; $Eff_{Q,Y}$ and $Eff_{Q+1,Y}$ are the seasonal scalars of the total effort of the fishery Y (mEff_YYY, unitless); $TotEff_Y$ is the total average daily effort (days per day) of the fishery Y (YYY_Effort, days per day); and $Nbox$ is the number of dynamic (non-boundary) boxes in the model domain (both boundary and dynamic boxes!). Note that if Q is the last quarter of the year, then Q+1 is the first quarter of the next year.

15.5.2. Constant effort per season adjusted by the relative box area (efformodel=1)

This option is set with YYY_effortmodel=1. It is calculated as above, but the distribution of total effort for each fishery across boxes (in days per day) is adjusted based on the relative area of the box.

$$Eff_{Y,j} = \frac{TotEff_Y \cdot \left(\varepsilon \cdot \left(Eff_{Q+1,Y} - Eff_{Q,Y}\right) + Eff_{Q,Y}\right) \cdot Area_j}{\sum_j Area_j}$$

Here $Area_j$ is the area of a box j and the denominator is the total area of the model domain (not counting boundary or land boxes).

15.5.3. Constant effort given by prescribed spatial effort matrices (effortmodel=3)

This option is set with YYY_effortmodel=3 and the user must prescribe both temporal (quarterly) and spatial (per box) relative effort distribution (i.e. proportion of the daily effort allocated to that box in that quarter).

The effort by the fishery Y in a cell j ($Eff_{Y,j}$ in days day^{-1}) in a given timestep is calculated as

$$Eff_{Y,j} = TotEff_Y \cdot \left(\varepsilon \cdot \left(hEff_{Q+1,Y,j} - hEff_{Q,Y,j}\right) + hEff_{Q,Y,j}\right) \cdot \left(\varepsilon \cdot \left(Eff_{Q+1,Y,j} - Eff_{Q,Y,j}\right) + Eff_{Q,Y,j}\right)$$

where ε is the proportion of the quarter of the year that has passed; $hEff_{Q,Y}$ and $hEff_{Q+1,Y}$ are the box-specific season-specific proportional effort of the fishery Y in quarter Q and Q+1 provided by the user (Effort_hdistribYYY); $Eff_{Q,Y}$ and $Eff_{Q+1,Y}$ are the seasonal scalars of the total effort of the fishery Y (mEff_YYY, unitless); and $TotEff_Y$ is the total average daily effort (days day^{-1}) of the fishery Y (YYY_Effort, days per day). The box-specific effort proportions must sum to 1 for each quarter of the year. The proportional effort is the season-specific scalar on the total yearly effort. The quarterly distributions and scalars are interpolated linearly through the year, in the same way as it is done in the prescribed vertebrate spatial distributions (see chapter 11.1).

15.5.4. Human population-based recreational fishing (efformodel=6, effortmodel=12)

This option is set with YYY_effortmodel=6 or 12 and is designed to capture **recreational fishing by people living in ports.** The recreational fisheries are setup in the same way as commercial fisheries and all recreational fishing should be pooled into one fishery. The recreational fishing option requires prescribed box specific season specific proportional effort matrices (Effort_hdistribYYY), like the effortmodel=3 case described in chapter 15.5.3; but it also requires the total average daily effort (YYY_Effort, days day^{-1}) of the fishery.

However, in contrast to the effortmodel=3 option, the YYY_Effort parameter **sets the average effort not by the entire fishery, but by one recreational fisher!** This is very important, because the final effort is multiplied by the number of people who fish in each location (port).

The YYY_effortmodel=6 will use port populations provided by the user (including any applied port population changes) and total effort $TotEff_Y$ from the *harvest.prm* file (YYY_Effort). The YYY_effortmodel=12 will get both the port population and total effort from the Economics submodel, where population and effort can change in response to a range of economics drivers (see chapter 17 on Economics). This means that if the Economics submodel is applied and a recreational fishery is used, it is best to apply YYY_effortmodel =12 to describe it.

The effort by a recreational fishery Y in a cell j ($Eff_{Y,j}$ in days day^{-1}) in a given timestep is calculated as

$$Eff_{Y,j} = \sum_Z \frac{FPop_Z \cdot Speed_Y \cdot dt}{distance_{Z,j}} \cdot TotEff_Y \cdot \left(\varepsilon \cdot \left(hEff_{Q+1,Y,j} - hEff_{Q,Y,j}\right) + hEff_{Q,Y,j}\right)$$

where ε is the proportion of the quarter of the year that has passed; $hEff_{Q,Y}$ and $hEff_{Q+1,Y}$ are the quarterly box-specific proportion of total effort of the fishery Y allocated to the box in quarter Q and Q+1 as provided by the user (Effort_hdistribYYY); $TotEff_Y$ is the total average daily effort (days per day) by **one fisher** (YYY_Effort, days day^{-1}); $FPop_Z$ is the number of people in port Z that fish recreationally, calculated as a multiplier of population in each port (ports_pop giving values for each port) and proportion of the population that fish recreationally (k_proprecfish, global parameter applied to all ports); $Speed_Y$ is the boat speed of recreational fishers (Speed_recboat); dt is the timestep of the model in hours; and $distance_Z$ is the distance from port Z to the box j.

The population of ports can change during the model simulation. The ports for which population

changes are given in the ports_popchange vector (where 1 indicates that population of a port changes) and the number of changes is given in the ports_popnumchange vector (both parameters must have as many entries as there are ports). The start of each change in each port is given in the PortPopChange_start_PortZ vector, which must have as many entries as there are changes in port Z. The period over which the port population change is given in the PortPopChange _period_PortZ parameter, which again must have as many entries as there are changes in port Z. Finally the multiplier for the final port population (to be reached at the end of the change period) is given in the PortPopChange_mult_PortZ. It must have as many entries as there are changes in port Z.

15.5.5. Constant or dynamic effort with spatial distribution based on the relative CPUE in the box (efformodel=2, effortmodel=10)

This option is calculated as in the option YYY_effortmodel=1 described above using mEff_YYY and YYY_Effort parameters, but the effort for each box is not weighted by the relative area of the box but by the relative catch per unit effort (CPUE) in that box over the "**memory" period**. This aims to imitate the situation where fishers return to areas where earlier catch has been good.

The "memory" period over which the CPUE is calculated is given in the K_num_catchqueue parameter in the *run.prm* file. It is often set to one day, which means that the CPUE effort matrices are updated every day, but it can be set for longer periods.

NOTE!

How is CPUE calculated in Atlantis?

Catch per unit effort (CPUE) is calculated based on the catch and effort over the "memory" period. For a fishery YYY the CPUE can be calculated either based on all catch (set as YYY_flaguseall=1) or only on catch of target species (YYY_flaguseall=0). The calculation of CPUE based on target species only assumes that non target species are of no value and fishers don't base their future effort allocation decisions on their catch. The target species of a fishery YYY are listed in the target_YYY parameter (where 1 indicates a target species and 0 indicates a non-target species).

Recreational fishing is not handled in the same way as commercial fishing, which means that CPUE for recreational fishing will be very overstated, as catch is lumped for invertebrates, as well as pelagic and demersal species, but is counted in the CPUE for each component species. This was done for ease of computation and is acceptable as long as dynamic effort allocation is **NOT** used for the recreational fishing. A warning to this effect has been added to the model initialisation code.

It is also possible to calculate shot by shot CPUE, which is used when trying to consider how to correct for changes in targeting and fishing behaviour through time. See the description here

There are two ways to apply the CPUE scaling. The first one applies constant effort through time and is set through YYY_effortmodel=2. In this option the relative scaling of effort per box is multiplied by the user provided total effort $TotEff_Y$ (YYY_Effort) to get the actual effort per box:

$$Eff_{Y,j} = TotEff_Y \cdot \left(\varepsilon \cdot \left(Eff_{Q+1,Y} - Eff_{Q,Y}\right) + Eff_{Q,Y}\right) \cdot \frac{cpue_{Y,j}}{\sum_j cpue_{Y,j}}$$

$TotEff_Y$ can be altered through the course of the run via scenarios of changing effort. This option is selected through YYY_effortmodel=10 is quite similar, but allows for a more dynamic calculation of the total effort, which can change throughout the simulation in response to potential management restrictions not just enforced effort changes. This means that the relative scaling of effort per box is multiplied not by the user provided constant effort but by the total effort in the previous time step (which includes all the applied effort scalars).

$$Eff_{Y,j} = TotEff_{Y,t-1} \cdot \left(\varepsilon \cdot \left(Eff_{Q+1,Y} - Eff_{Q,Y}\right) + Eff_{Q,Y}\right) \cdot \frac{cpue_{Y,j}}{\sum_j cpue_{Y,j}}$$

Here $TotEff_{Y-1}$ is calculated dynamically; $cpue_{Y,j}$ is the CPUE of fishery Y in box *j* during the last "memory" period (mg day^{-1}); and $\sum cpue_{Y,j}$ is the total CPUE of the fishery Y over the entire model domain during the same time period.

Note that for the burn-in period of the simulation (set using the tburn parameter in the *run.prm* file), the previous day's CPUE is not known and the effort still needs to be set with the effort spatial distribution parameters (Effort_hdistribYYY and YYY_Effort), as described in the chapter 15.5.3.

NOTE!

Exploratory fishing is important when effort is scaled by CPUE

This scaling of effort based on the previous CPUE means that spatial distribution of the fishery can change a lot through time. In particular, once no catch has been obtained in a specific box, the relative CPUE of that box will be 0, and the **fishery will not come to this box again**, as all subsequent scaling of effort for that box will be multiplied by 0. This means that a fishery will contract to a smaller area of the model. Such spatial contraction of the fishing effort is corrected once per year, if exploratory fishing is allowed, set as YYY_explore = 1. Exploratory fishing means that some effort is allocated to every box of the model domain, and the CPUE matrix is reset.

Exploratory fishing will only be done in boxes where the minimum and maximum depth are within the depth that a fishery can operate (YYY_maxdepth and YYY_mindepth parameters). The amount of fishing effort allocated in each box during exploratory fishing is set in the YYY_mEff_testfish parameter (days day^{-1}), which sets the same amount of fishing effort for each box. However, this effort is only applied if a fishery is open and allowed to operate (due to management rules). In a fishery is closed due to quota limits (for example) or other management factors (see chapter 16), on the reopening of the fishery, the exploratory fishing in a box where the fishery has not operated is set at a default level of 1 effort day per day.

Exploratory fishing is only carried out if no normal fishing is allocated to that box.

NOTE!

The role of the target species parameter target_YYY

To set the dynamic fishing option the users first have to identify target species for each fishery, set in the target_YYY vector.

The target_YYY and flagincidmort parameters determine which species in the cell are available to a fishery (see NOTE! in chapter 15.4).

The target_YYY will also determine whether a fishery will be closed when exceeding the total allowable catch – if only one species is targeted, then it will be closed as soon as its TAC has been reached. However, if a fishery has several target species, it will be closed only when it has reached TAC for the number of target species set in the YYY_num_max_sp parameter.

The catch and prices of the species identified as target species in target_YYY will also be used to **calculate CPUE** and in deciding where to concentrate the effort in dynamic effort allocations (prices are used only in the effort models that apply economics parameters).

The relative box biomass of target species is also used when setting up relative spatial effort in the case for YYY_efformodel=9.

15.5.6. Constant effort based on ideal knowledge of target fish distributions (effortmodel=9)

This is a simple option which assumes that the fishers know the fish distribution and can allocate the effort proportionally to the relative biomass of target species (identified in the target_YYY parameter) available in each box.

In this option **the spatial distribution of relative effort for the box j at each time step is set anew and is equal to the average relative biomass of all target species in the box j**.

$$Eff_{Y,j} = TotEff_Y \cdot \sum_n \frac{Biom_{CX,j}}{\sum_j Biom_{CX,j}} \cdot \frac{1}{n}$$

The total effort is given by the user in the YYY_Effort parameter (days per day); and n is the number of target species for the fishery YYY (target_YYY).

For the burn-in period of the simulation (set using the tburn parameter in the *run.prm* file) the effort will be set according to the prescribed effort spatial distribution parameters (Effort_hdistribYYY and YYY_Effort), as described in the chapter 15.5.3. This is done to ensure first, that the initial allocation of effort is not dependent on the relative biomasses in the initial conditions file, and second, to alert users that these two parameters must be set to reasonable values, because they will be used if all target species are fished out. If, for whatever reason, no target species are available in a model domain, the fishery will not close but the fishing effort will continue according to these prescribed paramaters (Effort_hdistribYYY and YYY_Effort). This assumes that even if all target species have been fished out the fishers will continue fishing and switch to other species.

While exploratory fishing is allowed for with this option (as in other cases it is only done if YYY_explore = 1), it does not currently expand the spatial distribution of the fishery through time (as that is only ever dictated by the target species or the prescribed distributions if no target species persist). It does allow for a small amount of fishing beyond the standard grounds, but little else. This could be modified in the future, on request, so exploratory fishing could update the Effort_hdistribYYY parameters dynamically. This would, for instance, be required in cases where fisheries operate based on effort quota rather than catch quota.

15.5.7. Effort read from forced time series files (effortmodel=11)

In this option the fishing effort per timestep per box per fishery is read from the forcing files. See Table 12 in chapter 8 on how to setup forcing files. Remember that if effort forcing files are provided, it is important to set the YYY_effortmodel to 11.

This can now include effort displacement and MPA considerations (including infringements) by setting the associated flags – flaginfringe (see section 15.3.3), YYY_flagmpa (see section 16.4), flagdisplace (see section 15.6.1).

15.5.8. Dynamic effort exponentially related to previous CPUE and boat speed (efformodel=4)

This option is set with YYY_effortmodel=4 and sets the effort dynamically based on the previous memory period's CPUE. The calculation of the new effort per box is not based on a simple scaling of the relative earlier CPUE as in the effort models described above, but is done in two steps. First, Atlantis calculates the 'ideal' effort per box, as an exponential function of the CPUE of the previous memory period, with a steepness of effort increase at high CPUE areas and maximum the maximum effort per box set by the user. Second, the rate at which new effort can be achieved is calculated based on the speed of the boat.

The new ideal effort by the fishery Y in a box j ($idEff_{Y,j}$ in days day^{-1}) in a given timestep is calculated as

$$idEff_{Y,j} = \frac{Effmax_Y}{1 + e^{(-Effa_Y \cdot cpue_{Y,j})}} - Effoffs_Y$$

where $Effmax_Y$ is the maximum potential effort per box allowed per fishery Y (YYY_mEff_max, days day^{-1}); $Effa_Y$ is the steepness of the effort increase to the maximum limit when CPUE is high (YYY_mEff_a, unitless and **typically in the range of 0.01 to 0.1**); and $Effoffs_Y$ is the offset of the effort (YYY_mEff_offset, days day^{-1}) to bring down effort in the lowest producibity boxes. This representation was originally developed for a location where fishing continued in locations even after CPUE had dropped to very low levels. In this approach if $Effoffs_Y$ is set to zero then the lowest possible effort levels is half of $Effmax_Y$. If you do not want any effort in the boxes with very low CPUE levels (e.g. zero) then you will need to set $Effmax_Y$ to twice the desired maximum effort level and $Effoffs_Y$ to half of that (i.e. the desired effort level). This approach may seem artificial, but it produces a range of fishing

pressure with the desired spatial heterogeneity and magnitude. Figure 1 shows the ideal new effort given the previous day's CPUE and the *Effa* parameter setting the steepness of the effort increase.

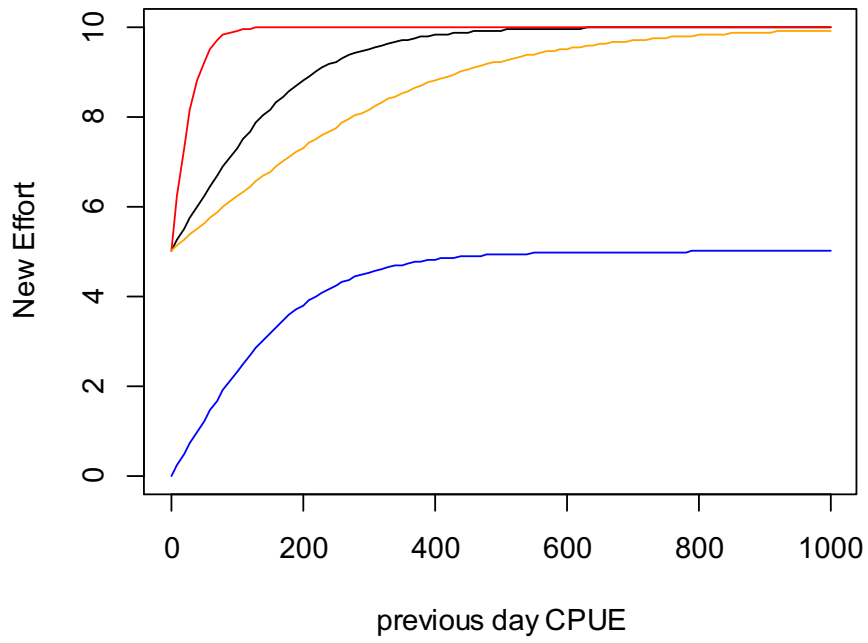


Figure 1. Ideal effort allocation in YYY_effortmodel=4, where previous day's CPUE (tons day-1 per box) varies from 0 to 1000.

mEff_max = 10 days day^{-1}

Black: mEff_a=0.01, offset=0
Red: mEff_a=0.05, offset=0
Orange: mEff_a=0.005, offset=0
Blue: mEff_a=0.01, offset=5 days day^{-1}

In the second step, the ideal effort is linearly interpolated across the model domain according to the speed of the fishery boats to obtain the realised new effort. This means that a fishery may not be able to switch to a new box immediately, if the boxes is large. The realised new effort is calculated as

$$newEff_{Y,j} = \max\left(1, \frac{Speed_Y \cdot dt}{width}\right) \cdot \left(idEff_{Y,j} - oldEff_{Y,j}\right) + oldEff_{Y,j}$$

where $Speed_Y$ is the speed of commercial fishing (Speed_boat, m hour^{-1}); *dt* is the number of hours in the main Atlantis timestep (usually 12, but 6 or 24 are also common); *width* is the width of the entire model domain along its the longest dimension (north–south or east-west); and $oldEff_{Y,j}$ is the effort by the fishery Y in the box *j* in the previous timestep. The (*Speed*dt/width*) term simply calculates what distance a fishery Y can travel in one time step compared to the entire model domain (e.g. 0.1 of the maximum possible distance in the model). It shows that if the fishery can cross the entire model domain in one timestep and therefore find itself in any box of the model domain in one timestep, then the scalar will be 1 and the new effort will be the same as the ideal effort. If the scalar is smaller than one, then new effort will be a combination of the old effort and the ideal effort determined above.

NOTE!

When the exponential CPUE to new ideal effort equation is used, **the amount of ideal effort in each box and in the entire model domain can change a lot through time.** If this ideal effort is translated quickly into the realised new effort (which happens if the boat speed is high enough, see below) then the amount of box specific and total fishing effort can also fluctuate A LOT through time. Users who apply this approach are advised to determine the ideal effort by carefully exploring the outputs of the effort in each box and throughout the model domain to ensure they produce

reasonable values. Such consideration is important for all models, but particularly so when using this option.

As with the majority of the other effortmodel cases, exploratory fishing can be used with this exponentially weighted effort distribution.

15.5.9. Dynamic effort exponentially related to previous CPUE and distance to ports **(efformodel=5)**

This option is set with YYY_effortmodel=5 and is a modification of the effortmodel=4 option described above. The ideal new effort per box is calculated in the same way using YYY_mEff_max, YYY_mEff_a and YYY_mEff_offset parameters and previous CPUE.

However, the new ideal effort is converted into the actual new effort not based on the speed of the boat, but based on the distance to ports that each fishery is associated with. This is used to simulate (more realistic!) cases where fisheries return to ports and may be unwilling to go to distant boxes even if the CPUE in those boxes is high.

This new scaling used to interpolate new ideal effort and old effort uses distances to ports and a specific user defined scalar $mFCscale_Y$. The new realised effort by fishery Y in box j is calculated as

$$newEff_{Y,j} = \sum_Z \frac{mFCscale_Y}{Z \cdot distance_{z,j}} \cdot \left(idEff_{Y,j} - oldEff_{Y,j}\right) + oldEff_{Y,j}$$

where $mFCscale_Y$ (YYY_mFCscale, unitless) is a fishery specific scalar which can heighten or lessen the effects of distance and in this way capture social and economic forces that may encourage fishers to go far into the sea or stay close to home (e.g. different fisheries may have different preferences); $distance_Z$ is a distance to port z; and Z is the total number of **ports active** for fishery Y (i.e. the fishery operates from that port and the port is open, see below). Figure 2 shows the effect of mFCscale parameter on the relative weighing of ideal effort. When mFCscale is high the ideal effort can be achieved quickly, whereas when the mFCscale is smaller the switch between the old and new ideal effort distributions is slow. The user entered mFCscale should be from 0.001 to 10. The lower the mFCscale the more homogeneous the distribution across the boxes (and closer to previous day's effort – i.e. the more stable the effort). The **higher the mFCscale the more the change in the effort will overshoot the new ideal effort** and the fishery will be effectively fishing only in the closest boxes. These patterns are achieved because the user entered mFCscale **is rescaled on read in** by the order of magnitude of the average distance of boxes to port.

For this option the user must give the number of ports (K_num_ports) in the *run.prm* file. The association of each fishery YYY with each port is given in the parameter ports_YYY which has as many entries as there are ports. The coordinates of the ports are given in the ports_x and ports_y paramters, which simply gives the xy coordinates (in metres) within the model space (these coordinates must be in the same projection as the BGM file and thus in metres). The easiest way of setting these coordinates is by using a GIS tool and converting the latitude and longitude coordinates using the projection string given in the top of the BGM file (see chapter 3). The time of port activity is given in ports_start and ports_end parameters, which allows for different ports to be active at different times of the simulation (perhaps reflecting historical settlement patterns).

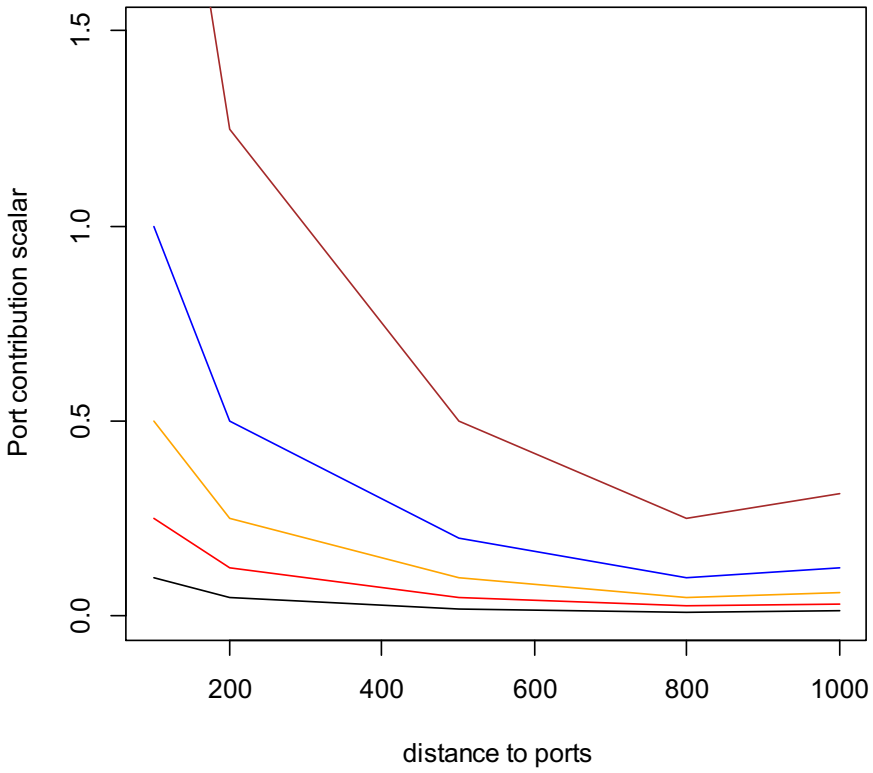


Figure 2. The port contribution scalar (mFC/distanceZ) for five boxes with a distance to ports set at 100, 200, 500, 800 and 1000 km (i.e. the mFCscale would be rescaled by the average distance of 520 on read-in).

Black: mFCscale=0.004
Red: mFCscale=0.01
Orange: mFCscale=0.02
Blue: mFCscale=0.04
Brown: mFCscale=0.1

When mFCscale values are very high it is possible that the effort will overshoot dramatically and will concentrate in one box with best CPUE of the previous day. This can be prevented by not setting a very high mFCscale value in the first place (or exploring how that might affect the effort), but also by capping the effort per cell. The effort cap is set by first identifying the fisheries where the effort per cell will be capped (YYY_flagcap_peak=1) and then setting the maximum effort per box using the YYY_mFCpeak parameter. If the effort is capped, then the effort in the most attractive cells will be reduced to the cap level given by YYY_mFCpeak and any additional effort instead redistributed and normalised across other attractive cells (which are below YYY_mFCpeak).

In addition, this effortmodel option allows for **exploratory fishing** once per year, as described in the chapter 15.5.5. This exploratory fishing will reset the CPUE matrices and will allow new effort in boxes where previous CPUE was zero.

15.5.10. Dynamic effort based on relative CPUE and distance to ports (effortmodel=7)

This option is activated with YYY_effortmodel=7 and is similar to effortmodel=5 and effortmodel=4 in that it calculates new ideal effort based on the previous CPUE (as for effortmodel=4) and interpolates the distribution of ideal effort to new realised effort based on the distance to ports and YYY_mFCscale parameter (as for effortmodel=5). However, the new ideal effort is not exponentially related to the previous CPUE (and therefore changing through time), but instead a simple scalar of the relative box CPUE on the old effort is used. This means that the new ideal effort represents a simple spatial redistribution of the old effort, rather than exponential *de novo* calculation of the amount of new effort in each box. The new ideal effort is thus calculated as

$$idEff_{Y,j} = \frac{cpue_{Y,j}}{\sum_j cpue_{Y,j}} \cdot TotoldEff_Y$$

Changes in the distribution of the ideal fishing effort in response to the previous period CPUE can be illustrated with Table 1, showing a hypothetical example of fishing in five boxes. The initial effort in each box is 20 days and the first day's CPUE varies from 1 to 50 tons/day. The new ideal effort on the next day will be heavily skewed towards boxes with highest CPUE, but the total effort will not change.

Table 4. Hypothetical example of changes in spatial ideal effort distribution in five boxes when effort is weighed by the previous day's (or other "memory" period) CPUE. Each box's distance to a port (only one port assumed) is indicated.

	Box 1 100km	**Box 2** 200km	**Box 3** 500km	**Box 4** 800km	**Box 5** 1000km	**Total**
Effort day t (days day^{-1})	20	20	20	20	20	**100**
CPUE day t (tons day^{-1})	5	10	50	20	1	**86**
relCPUE t	0.058	0.116	0.581	0.233	0.012	
Ideal effort on day t+1 (days day^{-1})	5.8	11.6	58.1	23.3	1.2	**100**
mFCscale/distance (mFCscale = 0.004; rescaled on read in to 2)	0.02	0.01	0.004	0.0025	0.002	
New effort day t+1	19.72	19.92	20.15	20.01	19.96	**99.76**
mFCscale/distance (mFCscale = 0.1; rescaled on read in to 50)	0.5	0.25	0.1	0.063	0.05	
New effort day t+1	12.91	17.91	23.81	20.20	19.06	**93.89**

The new actual effort is then calculated as in YYY_effortmodel=5, where the speed by which the fishery can change the effort depends on the distance of the box to the ports and the mFCscale parameter

$$newEff_{Y,j} = \sum_Z \frac{mFCscale_Y}{Z \cdot distance_{Z,j}} \cdot \left(idEff_{Y,j} - oldEff_{Y,j}\right) + oldEff_{Y,j}$$

This new effort then can be capped, as described in chapter 15.5.9. Note that even though in the first step the total amount of effort does not change but is only redistributed spatially, the introduction of the mFCscalar and distance to port scalar can lead to the changes in the total amount of realised effort (see Table 3).

If set with YYY_explore = 1, once per year the fisheries will do **exploratory fishing** to update the CPUE matrices. This is especially relevant for boxes where no fishing has occurred in previous timesteps. It is recommended that exploratory fishing be included in all effort models that use a CPUE-based approach (see NOTE! on this topic above).

Note that for the burn-in period of the simulation (set using the tburn parameter in the *run.prm* file), the previous day's CPUE is not used (as initially not known) and so the initial effort still needs to be set

with the effort spatial distribution parameters (Effort_hdistribYYY and YYY_Effort), as described in the chapter 15.5.3.

One additional step that occurs for the YYY_effortmodel=7 option is the potential for an annual check on total effort with modifications made based on the annual aggregate CPUE. If the CPUE is above a specified threshold CPUEthresh then the effort applied in the fishery is scaled up by CPUEscale, whereas if the annual CPUE is less than CPUEthresh then the effort is reduced by a scalar set to (1.0 – 0.5 * CPUEscale). A check on the divergence on effort is included however, and if the decline has been more rapid than a power law then it is reset to a power law.

15.5.11. Dynamic effort based on relative CPUE, relative distance to ports and boat speed with explicit effort (de)growth (effortmodel=8)

This option uses both relative CPUE of the box (like in effortmodel=7) and relative distance to the box from the active ports to calculate the effort. Further, this option allows the total effort to grow or contract in response to the CPUE and user set threshold parameters.

For the burn-in period of the simulation (set using the tburn parameter in the *run.prm* file) the initial effort still needs to be set with the effort spatial distribution parameters (Effort_hdistribYYY and YYY_Effort), as described in the chapter 15.5.3. At each other step the weight given to each box *j* for each fishery Y ($w_{Y,j}$) depends on the relative cost or distance of reaching that box compared to all other boxes and the relative CPUE in that box compared to all other boxes. This is calculated as

$$w_{Y,j} = \frac{\sum_Z \frac{mFCscale_Y}{Z \cdot distance_{Z,j}}}{\sum_j \sum_Z \frac{mFCscale_Y}{Z \cdot distance_{Z,j}}} \cdot \frac{cpue_{Y,j}}{\sum_j cpue_{Y,j}}$$

where the first term represents the relative distance of the box *j* and the second term is the relative CPUE in the box *j* by fishery Y. The mFCscale (YYY_mFCscale, unitless) is a fishery specific scalar which can heighten or lessen the effects of distance, see Figure 2.

The **key difference between this effortmodel=8 and the effortmodel=5 and 7 above**, is that here the distance to the box is used to assign the relative weight to each box when finding the ideal effort, whereas in the previous options the distance was only used to assess how quickly the new ideal effort can be achieved (or it may be never achieved if the new ideal effort changes faster than the fishery can react to this).

The ideal new fishing pressure in each cell is then calculated as the ratio of relative weight of box *j* to the sum of all box relative weights, multiplied by the total old effort by the fishery in the entire model domain (days day^{-1})

$$idEff_{Y,j} = \frac{w_{Y,j}}{\sum_j w_{Y,j}} \cdot TotoldEff_Y$$

The new actual effort is converted into the realised effort similarly to the option YYY_effortmodel=4, except that the speed of the fishery boats is not divided by the total width of the model domain, but by

the (absolute) distance between the box of the maximum effort j_{maxEff} and the box of maximum CPUE $j_{maxCPUE}$.

$$newEff_{Y,j} = \max\left(1, \frac{Speed_Y \cdot dt}{\left|distance(j_{maxEff} - j_{maxCPUE})\right|}\right) \cdot \left(idEff_{Y,j} - oldEff_{Y,j}\right) + oldEff_{Y,j}$$

The denominator of distance between j_{maxEff} and $j_{maxCPUE}$ aims to reflect the likely average distance the fishery will need to travel to achieve the ideal new effort. The ideal new effort $idEff_Y$ will be concentrated in the box of highest CPUE, and the largest number of boats will be in the box of maximum effort. So this equation assumes that the smaller the distance between the **two** model boxes with highest CPUE and highest effort, the faster the ideal new effort across the entire model domain can be achieved. If the fishery boats can travel the distance between j_{maxEff} and $j_{maxCPUE}$ in one time step, the new ideal effort **across the entire model domain** will be achieved in one time step as well.

The effortmodel=8 options includes **explicit growth or reduction of the fishing effort**, when CPUE is high or low. Once per week the fishery Y assesses whether the effort should be increased or decreased given the CPUE of the last memory period. If the CPUE is higher than the threshold CPUE provided in the YYY_mEff_thresh_top (kg day^{-1}) parameter, then the effort will grow by the YYY_mEff_shift/52 proportion, where 52 accounts for the number of weeks in the year. So the parameter YYY_mEff_shift gives the annual change in the effort.
Similarly, the effort will be reduced if the CPUE of the last memory period is smaller than the YYY_mEff_thresh parameter. The effort will be reduced by the YYY_mEff_shift/52 amount. This means that the upper and lower thresholds for growing and reducing the effort might be set at different values to reflect the willingness of the fishers to invest more or reduce the effort, but the amount the weekly effort will change in response to both triggers is the same.

This can be written in an equation showing the final *newEff** after increase or decrease in the fishing effort.

$$newEff^*_{Y,j} = \begin{cases} newEff_{Y,j} + \dfrac{EffShift}{52}, & cpue_t > mEff_{thresh}\_top \\ newEff_{Y,j}, & mEff_{thresh} > cpue_t > mEff_{thresh}\_top \\ newEff_{Y,j} - \dfrac{EffShift}{52}, & cpue_t < mEff_{thresh} \end{cases}$$

Note that the spatial allocation of effort does not change, so the increased effort is not all allocated to the best box but is distributed proportionally to all boxes according to the new realised effort.

Finally, if set with YYY_explore = 1, once per year the fisheries will do **exploratory fishing** to update the CPUE matrices. Remember that this option is recommended in all effort models that use CPUE approach (see NOTE! above).

NOTE!

"Overloaded" parameters

Occasionally in Atlantis the same parameter is used in different ways under different options, or has a double role (called overloading in programmer speak). YYY_mEff_thresh is one such parameter. Not only does it dictate the lower band of CPUE, below which effort may leave the fishery under the YYY_effortmodel=8 option, but it is also the level CPUE below which effort may be displaced (either by an MPA or poor stock status).

15.5.12. Dynamic effort time series as calculated in the economics model

In this option the effort per fishery per box is calculated dynamically in a separate Economics submodel. It is set through YYY_effortmodel=13 and will be described separately in the Economics chapter.

15.5.13. Prescribed changes in fishing effort

As with some other fishing parameters, users can set up temporal changes in the fishing effort, which means that the effort can be forced to increase or decrease over a set period of time. Several changes in the effort can be applied. The prescribed change in fishing effort is activated with a global flag flagchangeeffort and with fishery specific parameters – YYY_num_changes indicating number of changes for each fishery, EffortChange_start_YYY giving the start date of each effort change, EffortChange_period_YYY giving the period over which each effort changes to the new desired level, Effort_mult_YYY giving the multiplier of the each change in the effort (>1 if effort should increase and <1 if it should decrease) and flagpulseYYY parameter to indicate whether each change in the effort is a pulse change (flagpulseYYY=1) or not. The pulse change means that after the change period the effort will return to the original value, while non-pulse change means that the changed effort will apply for the rest of the simulation or until the next change in the effort.

For the effort models that use prescribed, i.e. constant total effort given in the YYY_Effort parameter (effortmodel 0, 1, 2, 3, 6, 9) the multiplier will be applied to this YYY_Effort parameter. This means that every new change in the effort will apply to the original effort value and NOT to the latest effort value. In the dynamic effort models, where total effort is determined dynamically based on CPUE, distance to ports or economic parameter, the effort change multiplier will be applied to the latest total effort calculated dynamically. This means that in addition to the dynamic change in the effort, the user can setup forced changes that will scale up or down the total effort over the set period. This could, for example, be used to simulate cases where some external factors not included in the model (fishery buy-outs, subsidies to fishers) might have affect the total effort. Remember that pulse change in the effort (flagpulseYYY=1) can be applied to both prescribed and dynamic effort models. If effort change is pulsed, for prescribed effort cases the effort will return to the values given in the parameter file (YYY_Effort) after the change period. For dynamic pulse effort, the total effort values after the change period will be divided by the change multiplier on the first time step after the pulse ends.

Table 5. Parameters used to setup different effort options

Parameter	Description
YYY_effortmodel	Flag indicating which effort model determines the total effort by fishery YYY and its spatial distribution (options 0 to 13 available).
Effort_vdistribYYY	A vector giving relative distribution of effort for fishery YYY across the vertical layers in each box (applied in the same way as the vectors determining vertical distribution of functional groups)
Effort_hdistribYYY1 Effort_hdistribYYY2 Effort_hdistribYYY3 Effort_hdistribYYY4	A vector setting relative effort per box by the fishery YYY for each quarter of the year (1 to 4). It is used in combination with YYY_Effort to get the actual final effort per box in days per day
YYY_sweptarea	The swept volume of water by gear of fishery YYY, m^3 day^{-1}. This is a volume, but called area to keep in line with the fisheries science usage.
flagescapement_XXX	Flag setting which escapement formulation to use. 0 – no escapement, 1- constant escapement per age class, 2 – size based escapement.
p_escape_XXX	A vector giving proportions of each age class that will escape the fishing gear
Ka_escape_XXX	The constant in length-based escapement formulation for species XXX (when flagescapementXXX=2)
Kb_escape_XXX	The second constant in length-based escapement formulation for species XXX (when flagescapementXXX=2)
YYY_maxdepth	The maximum depth that a fishery can operate at, in meters. This will constrain the area available to a fishery, which means that a fishery will not operate in a cell that is outside the depth limits, regarless of the effort model applied
YYY_mindepth	The minimum depth that a fishery can operate at, in metres. This will constrain the area available to a fishery, which means that a fishery will not operate in a cell that is outside the depth limits, regardless of the effort model applied
YYY_Effort	Average effort by the fishery YYY across the entire model domain throughout the year (days per day). When applied to recreational fisheries this sets the average effort by one fisher and not by the entire fishery (so the value should be much smaller)
mEff_YYY	A vector of four values setting relative quarterly effort (used in combination with YYY_Effort to give the final effort in days per day)
ports_pop	A vector giving the population of each port (should have the same number of values as there ports indicated in K_num_ports parameter in the *run.prm* file)
k_proprecfish	Proportion of population in ALL ports that go recreational fishing. This parameter applies the same proportion to all ports
ports_popchange	A vector of values indicating whether a port population changes during the simulation (0 – no, 1 – yes). It must have as many values as there are ports.
ports_popnumchange	A vector of values indicating how many population changes occur in each port. It must have as many values as there are ports
PortPopChange_start_PortZ	The day of the simulation when population change(s) in port Z start.

Parameter	Description
	There must be as many entries as there are changes in port Z.
PortPopChange _period_PortZ	Period of time (in days) over which the population of port Z changes. There must be as many entries as there are changes in port Z.
PortPopChange_mult_PortZ	The multiplier for the final port population, to be reached at the end of the change period. There must be as many entries as there are changes in port Z.
Speed_recboat	Speed of recreational boats (m hour^{-1})
YYY_mEff_max	The maximum effort per box allowed per fishery YYY in the exponential CPUE related effort allocation option (days day^{-1}), used in YYY_efformodel =4 and 5
YYY_mEff_a	Steepness of the effort increase in response to CPUE in the exponential CPUE related effort allocation option (unitless), used in YYY_efformodel =4 and 5
YYY_mEff_offset	The effort offset (days day^{-1}) to reduce the effort per box, used in YYY_efformodel=4 and 5
K_num_catchqueue in *run.prm*	The length of the "memory" period to calculate the latest CPUE, used in all effort models that require CPUE estimates
target_YYY	A Boolean vector indicating which species are targeted by the fishery YYY. The vector should have as many entries as there are values in the *functional_groups.csv* file. 1- species is targeted, 0 – species is not targeted (see NOTE! above on what this parameter means)
YYY_flaguseall	A flag indicating whether to use all species caught to define CPUE of a fishery YYY for CPUE based effort models (1=yes) or target species only (0).
YYY_mFCscale	A fishery specific distance scalar used in determining the effort distribution; it heightens or lessens the effects of distance between the optimal box and the ports (unitless)
ports _x and ports_y	Vectors giving x and y coordinates of each port in metres in the model space projection.
ports_start and ports_end	Vectors giving the period of port activity during the simulation. The ports can be active from the first timestep or only for a set period of time. If a port is not active, then no fishery will work from that port.
ports_YYY	A vector giving association of each fishery YYY with each port. It has as many entries as there ports, with a value of 1 indicating that the fishery works from that port and a 0 meaning that fishery does not work from the port.
YYY_flagcap_peak	A flag indicating whether the effort of fishery YYY in any one box is capped (0 – no, 1 – yes)
YYY_mFCpeak	If the effort of a fishery YYY is capped (YYY_flagcap_peak=1), this sets the maximum potential effort in any one box (days day^{-1})
Speed_boat	Speed of commercial fisheries boats (m hour^{-1}). This is used across for all fisheries to estimate how quickly the new ideal effort can be realised. Applied in YYY_efformodel=4 and 8
YYY_explore	Flag indicating whether once per year a fishery YYY will do exploratory fishing in all accessible boxes (within YYY_maxdepth and YYY_mindepth depth ranges). 1 – yes, 0 – no. Exploratory fishing is

Parameter	Description
	recommended with all CPUE based effort models
YYY_mEff_testfish	The exploratory fishing effort (days day^{-1}) to be applied to each box if YYY_exolore is set to 1. Note, this effort is applied in addition to normal fishing
YYY_mEff_thresh	The CPUE threshold (kg day^{-1}) below which the fishery YYY will decrease the effort in YYY_effortmodel=8
YYY_mEff_thresh_top	The CPUE threshold (kg day^{-1}) above which the fishery YYY will increase the effort in YYY_effortmodel =8
YYY_mEff_shift	The yearly proportional increase/decrease in effort if CPUE is above or below the threshold levels in YYY_effortmodel =8. The increase/decrease in threshold happens once per week and is decreased by YYY_mEff_shift/52

15.6. Effort displacement, caps and effort drop

15.6.1. Effort displacement

Once the spatial allocation of effort is done according to options describe above, it is possible to allow for **some effort displacement to adjacent** boxes that can offer better catch opportunities. Better catch opportunities (higher CPUE) will be found in boxes with either higher biomasses of target species and/or lower proportion of MPAs.

The effort displacement is an optional feature activated with global flag flagdisplace=1. If effort displacement is activated then it will occur when

1) CPUE in a box falls below the YYY_mEff_thresh level (mgN of catch per m^3 per day), a fishery specific threshold level (note that this YYY_mEff_thresh is the same parameter used in YYY_effortmodel=8 discussed above – see the above Note! on this topic).
2) MPAs are imposed

If CPUE falls below the threshold level Atlantis then assesses which if any of the adjacent boxes have a higher biomass of the species targeted by the fishery. Once the best box is found some of the effort is then moved to the best adjacent box. The proportion of the displaced effort is calculated as the relative proportion of biomasses in the current versus adjacent box, as

$$prop_{disp} = \frac{Biom_{targ,i}}{Biom_{targ,i} + Biom_{targ,opt}}$$

where $Biom_{targ,i}$ is the biomass of all target species in box *i* (which has CPUE lower than the threshold); and $Biom_{targ,opt}$ is the biomass of all target species in an adjacent box with highest biomasses. The new effort in the box i is then calculated as ($Effort_i$ – $prop_{displ}$).

If CPUE did not fall below the threshold, but MPAs are imposed, the effort displacement will still occur (if flagdisplace=1). The amount of effort displaced is the difference between the original effort in the box and the effort left after the MPA restrictions are imposed (the MPA scalar).

15.6.2. Effort caps

The next routine applied is *Check_Cap()* which allows for the implementation of effort caps (effort quotas) or can be used to more evenly smear effort through time (effectively imposing effort-based trip limits earlier in a season). The users can set a maximum annual effort applied by a fishery. This is done by YYY_flagcap=1. If effort caps are applied then the user should provide a cap of effort for each fishery in YYY_cap parameter (in total number of days fished per year). If after all calculations and adjustments, the cumulative effort applied is above the cap then fishing will cease.

The effort caps can change through time, and it is set in the same way as changes in other fisheries parameters (see chapter 15.10). Setting changes in effort caps requires turning on a global flag flagchangecap=1, indicating the number of changes in effort caps in YYY_cap_changes, giving start dates of each cap change in CAPchange_start_YYY, period over which the change takes effect (CAPchange_period_YYY) and the multiplier to be reached at the end of the change period (CAPchange_mult_YYY).

Note effort caps can also be used to force more even temporal distributions of effort. In this case for all effort models except the forced time series and economics based effort models the annual effort level extrapolated from the current daily effort level (i.e. the current effort level * 365) is compared against the cap. If the extrapolated annual effort is greater than the cap then the current effort is reset to the capped effort level divided by 365. This levels out some of the strong spikes in effort that can occur on a day-to-day basis and can see effort pile up in the early days of a season and then dwindle later on. Such patchy effort is not unrealistic, however, so using the cap in this way must be done with caution.

NOTE!

Capping of effort should be avoided unless well justified

The capping of effort provides an ad hoc approach to limit effort in a box (or through time), and reduces the potential understanding of the underlying processes which are leading to effort in one box increasing to high levels. **Effort caps are different from effort displacement, and they simply reduce the dynamically calculated effort**. This means that some **effort is lost from the system** without the explicit factors (MPA, low CPUE, TAC reached, etc.) accounting for this effort decrease.

Applying caps means that the user either assumes that (i) there are some external limitations on effort (such as port extent, maintenance requirements, physical space or specific regulations) that limit effort concentration (thereby justifying the use of a cap) or (ii) due to inevitable model limitations Atlantis cannot reproduce the effort dynamics sufficiently well and the model is producing effort levels that are higher than they should be. In the later case, in particular, it is important to try and understand why the effort increases too much and explore whether a more reasonable parameterisation or formulation is possible. It is useful to trace the sources of effort increase in one box to check on whether the assumptions are justified, e.g. whether

1) the increase is due to high CPUE and if so, should the effort be capped? It might be better to adjust the CPUE threshold parameters and in this way allow the effort to change dynamically

2) the increase is due to displacement due to MPAs? If so, is it reasonable to cap it? What would happen in the real world if large MPAs are imposed? Perhaps it would be better to include some MPA infringement instead?

15.6.3. Effort drop

Some parameterisations of the CPUE and distance based models can lead to precipitous and unobserved drops in effort once fish stocks become depleted. This can be overcome (if well justified, see the Note! above) by setting flagbuffereffort to 1. This will use the weighting inherent to those effortmodel formulations to distribute effort, but will constrain any abnormally large increases or reductions in effort to those dictated by the management decisions in place (e.g. MPAS, quotas and the like) rather than any highly variable CPUE distributions.

Table 6. Parameters used to setup other fishery relevant different options

Parameter	Description
flagdisplace	If set to 1, effort in dynamic fishery will be displaced into adjacent cells when CPUE falls below a certain threshold or when MPAs are imposed
YYY_mEff_thresh level	Level of CPUE below which effort displacement occurs
YYY_flagcap	If set to 1, the effort will be capped at the maximum level set in YYY_cap
YYY_cap	Effort cap of a fishery (maximum number of days per year)
flagchangecap	If set to 1, effort caps can change through time. See chapter 15.10 on how temporal changes are implemented
flagbuffereffort	If set to 1, sudden decreases in dynamic effort will be buffered through time (and hence not allowed). This should be avoided unless well justified (see Note! in chapter 15.6.2)

15.7. Vertical, positional and habitat scalars of the fishing effort

15.7.1. Vertical distribution of fishing effort

The calculation of effort described above is done at a box level. However, this must be translated to the effort per cell or each layer of a box, as all processes in Atlantis are repeated in each cell. The vertical scaling of effort distribution in each box is done using simple vertical scaling Effort_vdistribYYY, in the same way as for the vertical distribution of the biological functional groups. The vertical distribution stays stable over time. However, each fishery has a minimum and maximum depth it can access, so the final effort will be rescaled to fit into the accessible layers.

Once distributed vertically, the effort of a fishery at each layer is compared to the vertical distribution of a species (VERTnight_XXX and VERTday_XXX in the *biology.prm* file) given the time of the day that the fishery operates, to get the biomass available per box (i.e. the $\delta_{depth.CX,I}$ scalar). If a fishery only operates at night, the fishery's vertical distribution is compared against the species vertical

distribution at night. So it can be important that there is not temporal/vertical mismatch between the fishery and its target species (i.e. indicating it fishes at night and then having diel vertical migration of the target species such that there is no overlap with the fishery).

15.7.2. Accounting for pelagic/demersal mismatch - positional availability

Atlantis can account for the **pelagic/demersal mismatch** between the species distribution and fishery. Each species and each fishery can be set as either pelagic or demersal (parameters YYY_flagdempelfishery and flagdemXXX respectively). If a species is pelagic and fishery is demersal (or vice versa), and a fishery does not access to the entire water column (flag_access_thru_wc_YYY=0) then the availability of a species will be scaled by the mismatch scalar, set through a global parameter k_mismatch. If you don't want the mismatch factor to apply, set flag_access_thru_wc_YYY=1. This scaling is included in the $\delta_{depth.CX,I}$ scalar and applies to all water column layers the fishery is marked as accessing in its vertical distribution Effort_vdistribYYY.

15.7.3. Habitat scalar for fisheries availability

Atlantis will determine a species, box and fisheries specific habitat scalar $\delta_{habitat.CX,i,Y}$ assessed depending on the proportion of species biomass available given the habitat restrictions of a fishery and a species distribution across different habitats. This assumes that habitats can provide a certain level of cover from fishing, and that this cover is provided throughout the entire water column, as the habitat scalar applies to all water column layers. The habitats that a fishery can operate in are given in the habitat_YYY parameter, which must have as many values as there are biogenic and physical habitats in the model. The biogenic habitats are "created" by the functional groups identified with isCover parameter in the *functional_groups.csv*. Physical habitats include "reef, flat, soft, canyon" and are always placed at the end of the habitat_YYY vector. If the value of habitat_YYY is set to 1 for a habitat then the fishery can act in and over that habitat type (and thereby interact with any species which also lives in or over that habitat type). If the value of habitat_YYY is set to 0 then the fishery cannot access that habitat type (and is blocked from fishing the proportion of the cell made up of that habitat type).

Three different options are used to calculate the overall habitat refuge scalar, selected using the flaghabitat_XXX parameter in the *harvest.prm* file (the parameter must have as many values as there are fisheries, so a species can have different habitat scaling for different fisheries):

1) **flaghabitat_XXX = 0**. Simple proportional habitat overlap of fishery and species

In this option, the biomass available to the fishery is simply the proportional overlap between the species' and fishery's available habitats. If both a species and a fishery can operate in all habitats, the scalar is 1. Alternatively, if only 25 or 50% of the used habitat area is shared between a species and fishery then the habitat scalar will be set at 0.25 or 0.5 respectively.

If the spatial coverage of the fishery is **set to change through time**, as set in the flagchangep and YYY_changeP parameter, then the habitat scalar calculated above will be further scaled by the scalar defining spatial coverage change. See chapter 15.10 on how forced parameter changes are set.

Note, this fishery-species habitat overlap calculation is different from the habitat overlap between a prey and predator used in the Ecology submodel in two ways:

a) In Ecology the habitat dependency of the prey and the state of that habitat dictates how difficult it is for the predator to access the prey, while a predators habitat preferences can dictate its spatial movement/distribution. There is no explicit direct comparison of overlap as with fisheries.
b) In Harvest submodel the habitat overlap is applied to all species (as all fisheries are assumed to have potential habitat dependency in their ability to access specific geological seabed types etc.), whereas in Ecology it is applied only to species that are identified as habitat dependent.

2) **flaghabitat_XXX = 1**. In this option the habitat refuge from fisheries for species XXX is calculated according to the Ellis and Pantus (2001) approach, which aims to account for the patchiness of habitats and fishing processes using a Poisson or uniform distribution. This option requires three additional parameters (k_pattern, k_patches, k_coverYYY) and here the $\delta_{habitat.CX,i}$ scalar is calculated as

$$\delta^{Y}_{habitat,CX,i} = 1 - \min\left(1, scale^{Y}_{CX,i}\right)$$

and

$$scale^{Y}_{CX,i} = \left(prop_{fish} \cdot \exp\left(-\frac{k_{cover}}{prop_{fish}} \cdot \frac{log\left(1 + q_{CX,Y} \cdot k_{pattern}\right)}{k_{pattern}}\right) + 1 - prop_{fish}\right)^{k_{patches}}$$

where $prop_{fish}$ is the proportion of box area accessible to the fishery according to the habitats they can access, which can also be seen as a probability that the fishing gear has access to the individuals of species in this box (note, this is NOT a proportion of area fished based on effort allocations, but just the habitat overlap)
k_cover is a fishery specific parameter indicating the level of cover perceived by the fishery in the patch (this will be lower if using gear such as rock-hoppers etc)
k_pattern is a global parameter setting the distribution of fishing, where 0 is Poisson, -1 is uniform and a value > 0 indicates an aggregated negative binomial distribution
k_patches it the total number of patches of different fishing intensity per box, typically set to 2 (fished and not fished)

If the spatial coverage of the fishery is **set to change through time**, as set in the flagchangep and YYY_changeP parameter, the $prop_{fish}$ will be scaled by the scalar defining spatial coverage change. See chapter 15.10 on how forced parameter changes are set. Note, under this formulation if *k_pattern* is set to 0 then catchability has no effect as a parameter as the spatial model is assumed to implicit subsume catchability.

3) **flaghabitat_XXX = 2**. This is a newer two-stage variant of the Ellis and Pantus approach, which fully described in Ellis et al (2014) *Can. J. Fish. Aquat. Sci.* 71: 733–746. It limits the lower bound of the patchiness and allows for a more explicit influence of catchability and further refines the patchiness distributions.

People interested in either **flaghabitat_XXX=1 or 2 should read Ellis and Pantus (2001, 2014)** so they have a full understanding of what this option does. It is a powerful representation, but requires some thought in application and parameterisation (preferably done with spatially resolved logbook data).

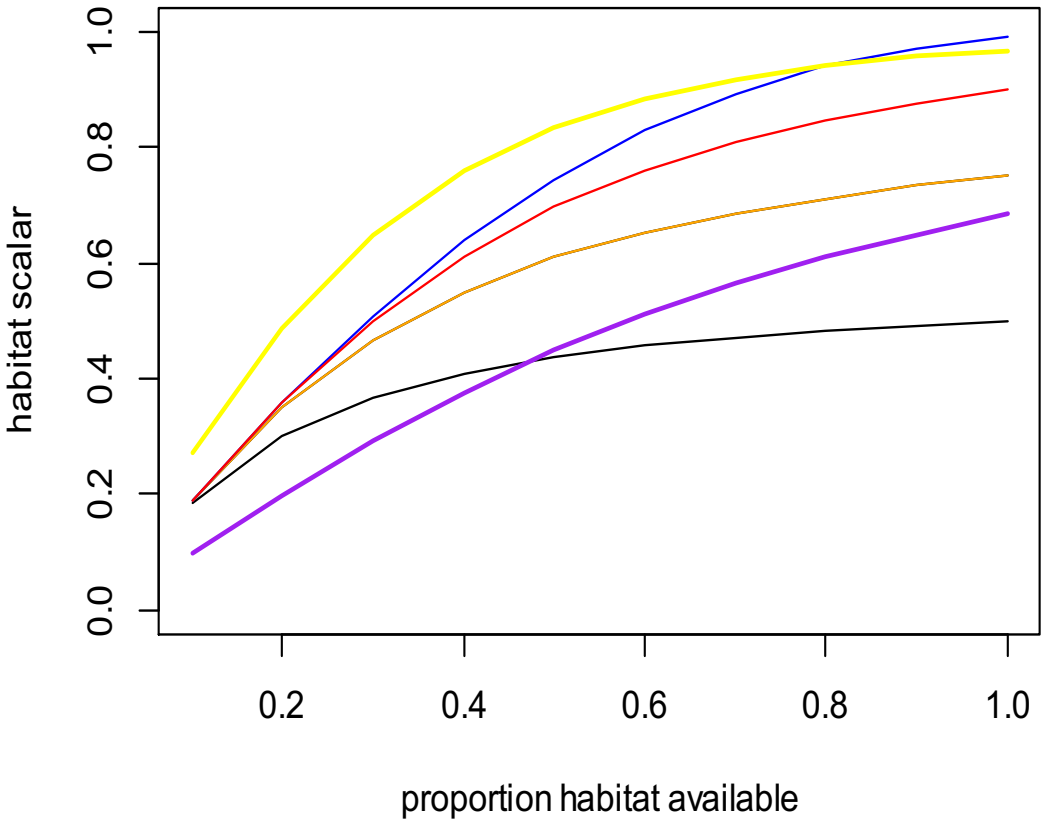


Figure 3. Shape of the habitat scalar as a function of habitat overlap between a species and a fishery, when k_pattern=-1

k_patches=2
Black: q=0.5, k_cover=0.5
Orange: q=0.5, k_cover=1
Red: q=0.9, k_cover=0.5
Blue: q=0.9, k_cover=1

Yellow: same as blue, but k_patches=3
Purple: same as blue, but k_patches=1

Table 7. Parameters used to setup habitat scalar options

Parameter	Description
flag_access_thru_wc_YYY	If set to 1, fishery has access to both demersal and pelagic species. Otherwise the effectiveness of the fisheries gear (which is pelagic or demersal) will be reduced by k_mismatch when targeting demersal or pelagic species respectively
k_mismatch	See above, the reduction in fisheries gear effectiveness
flagchangep	If set to 1, a spatial coverage of the fishery changes through time (see chapter 15.10)
k_pattern k_patches k_coverYYY	Parameters used to set up Ellis and Pantus (2000) habitat refuge from fisheries (see chapter 15.7.3)

15.8. Fisheries gear selectivity, swept area and conflict

Atlantis has several options for setting up selectivity of the fishing gear. Each fishery can have only one gear selectivity option, although the selectivity can change through time. The selectivity will determine how much of the encountered fish will be caught (see Note! in chapter 15.4 on how catchability and escapement will also determine this). Nine selectivity options are currently available in Atlantis, selected through YYY_selcurve parameter. The simplest version of the selectivity is set as a constant

proportion applied to all age classes, but it can also be size based and calculated based on a normal, logistic, lognormal, gamma, knife-edged, bimodal and binormal distributions.

15.8.1. Setting up changes in fisheries gear selectivity

Atlantis allows for temporal changes in gear selectivity and even changes between different selectivity types. So a fishery can change from constant to logistic selectivity, for example. The selectivity changes are first indicated with the global flag flagchangesel, which when set to 1 tells Atlantis that at least one fishery will undergo selectivity changes. This is then refined with the flag YYY_changeSEL, which sets (if YYY_changeSEL =1) which specific fishery has changes in selectivity. The number of changes for each fishery is given in YYY_sel_changes and the start (in days of the model run) of each change is given in SELchange_start_YYY. The new type of the selectivity is given in SELchange_curve_YYY parameter, which must have as many entries as there are changes. If the type of selectivity stays the same, make sure parameters YYY_selcurve and SELchange_curve_YYY have the same value. Finally, the change in the selectivity curve is given with SELchange_addlsm_YYY and SELchange_addsigma_YYY (see below how these are used for different selectivity options).

15.8.2. Constant selectivity (selcurve=0)

The constant selectivity is set with YYY_selcurve=0 and it assumes that a constant proportion of each age class is caught by the gear. This proportion is set for each species XXX with the parameter sel_XXX, which must have as many entries as there are fisheries (so different fisheries can have different selectivity).

The proportion of age group *i* selected by the gear is $psel_i$ and it is calculated as

$$psel_i = sel$$

where *sel* is given in sel_XXX.

If **selectivity changes** through time, but remains in the form of a constant selectivity (fisheries can switch selectivity types altogether) then the new selectivity is calculated as

$$psel_i = sel + lsm$$

where *lsm* is the SELchange_addlsm_YYY parameter for fishery YYY, it represents the adjustment in the constant selectivity on species XXX, so that there is a change in the proportion the fishery selects. After the change in selectivity takes place, the new constant function will use the sel_XXX parameter from the *harvest.prm* file with SELchange_addlsm_YYY **added to it.**

Note that the addlsm parameter is used to represent changes in selectivity, whether the selectivity curve changes from one constant to another constant proportion, or whether it changes from, say, logistic to a constant one.

15.8.3. Constant selectivity for juveniles and adults (selcurve=1)

This option is similar to the selcurve=0 option, but allows different selectivity for juveniles (sel_XXXjuv) and adults (sel_XXXad). The first age group that is considered adult is determined by the XXX_age_mat parameter in the *biology.prm* file. **Remember that Atlantis counts from zero**, so XXX_age_mat=4 means that the adult stage starts from the fifth age group.

Similar to the option above, if there is a selectivity change, the new selected proportion for juveniles and adults will be calculated as

sel_XXXjuv + SELchange_addlsm_YYY
and
sel_XXXad + SELchange_addlsm_YYY

15.8.4. Calculation of length for length-based selectivity

The remaining gear selectivity options use length of an average individual to calculate gear selectivity. Since length is dynamic and will change in response to feeding condition, the realised gear selectivity will also be dynamic through time. See NOTE! in chapter 10.2.5 on how length of different functional groups is calculated.

15.8.5. Logistic length-based selectivity curve (selcurve=2)

The logistic selectivity curve is one of the most commonly applied length-based selectivity options in fisheries (most often used for trawl nets or other gear where the likelihood of capture grows to plateau with larger sizes).

It is defined by two parameters – the length at 50% selectivity, also defined as the inflection point *lsm* (YYYsel_lsm, given in cm) and the spread or steepness of the selectivity curve *selb* (YYYsel_b). See Fig. 4 for how the parameter *selb* affects the shape of the selectivity curve. The selectivity of age group *i* is calculated as

$$psel_i = \frac{1}{1 + \exp\left(-selb \cdot (len_i - lsm)\right)}$$

where len_i is the length of the age group i.

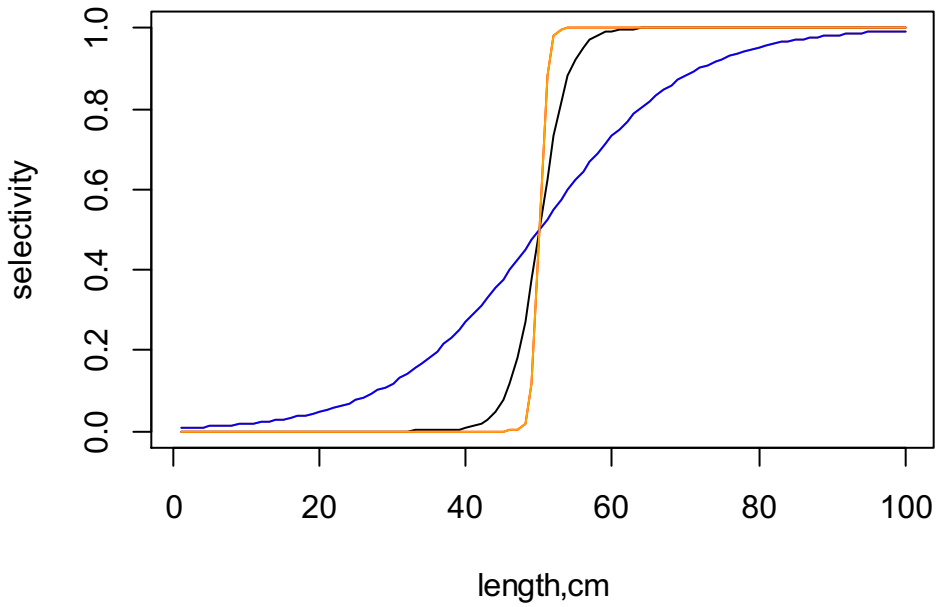


Figure 4. Shape of the logistic gear selectivity curve. The inflection point (sel_lsm) is set at 50cm.

Black: sel_b=0.5
Orange: sel_b=2
Blue: sel_b=0.1

If a change in selectivity is included in the simulation (using flagchangesel, YYY_changeSEL and other parameters, see above), then the *lsm* and *selb* parameters after the change takes place is set at

lsm + SELchange_addlsm_YYY
selb + SELchange_addsigma_YYY

15.8.6. Normal length-based selectivity curve (selcurve=3)

This option applies a normal distribution selectivity curve and is defined by two parameters – the length at highest selectivity *lsm* (YYY_normlsm, given in cm) and spread of the normal selectivity curve *sigma* (YYY_normsigma). The selectivity of age group *i* is calculated as

$$psel_i = \exp\left(\frac{-(len_i - lsm)^2}{2 \cdot sigma^2}\right)$$

where len_i is the length of the age group i. The figure 5 shows the shape of the normal selectivity curve given three values of the *sigma* parameter.

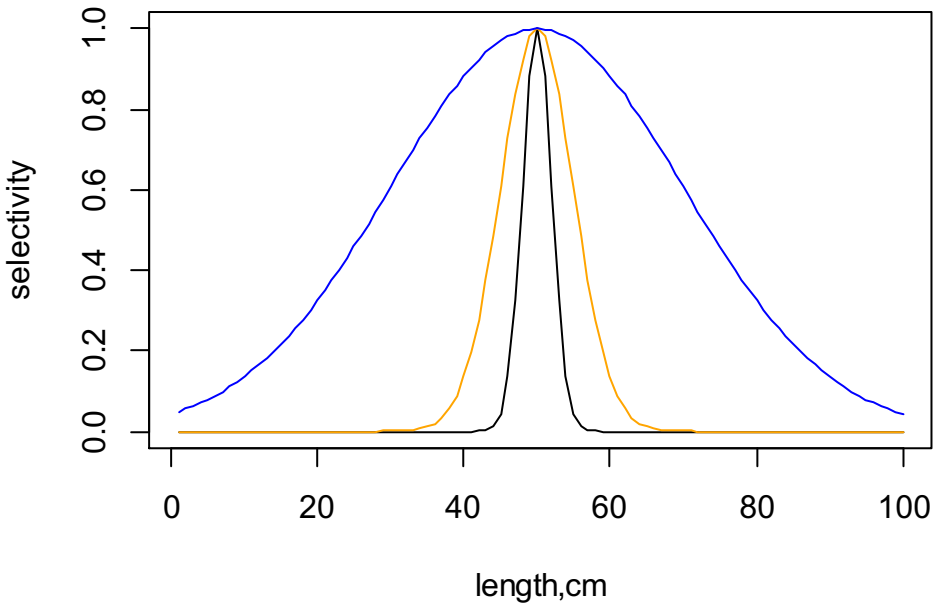


Figure 5. Shape of the normal gear selectivity curve for an organism of length 1-100cm. The highest selectivity point (*lsm*) is set at 50cm.

Black: sigma=2
Orange: sigma=5
Blue: sigma=20

If a change in selectivity is included in the simulation (using flagchangesel, YYY_changeSEL and other parameters, see above), then the *lsm* and *sigma* parameters after the change are calculated as

lsm + SELchange_addlsm_YYY
sigma + SELchange_addsigma_YYY

15.8.7. Lognormal length-based selectivity curve (selcurve=4)

This option applies lognormal distribution selectivity curve and is defined by two parameters – the parameter that sets length at highest selectivity *lsm* (YYY_lognormlsm) where the length of maximum selectivity (in cm) is approximately $e^{YYY\_lognormlsm}$ and the spread of the selectivity curve *sigma* (YYY_lognormsigma). The higher the sigma term the flatter the curve (approaching a close to uniform, though low selectivity when *sigma* ~ *lsm*). The selectivity of age group *i* is calculated as

$$psel_i = \frac{\exp\left(\frac{-(log(len_i) - lsm)^2}{2 \cdot sigma^2}\right)}{sigma \cdot \sqrt{2\pi}}$$

The figure 6 shows the shape of the lognormal selectivity curve given two values of *lsm* and *sigma* parameters (note that the curve is very sensitive to the parameter values).

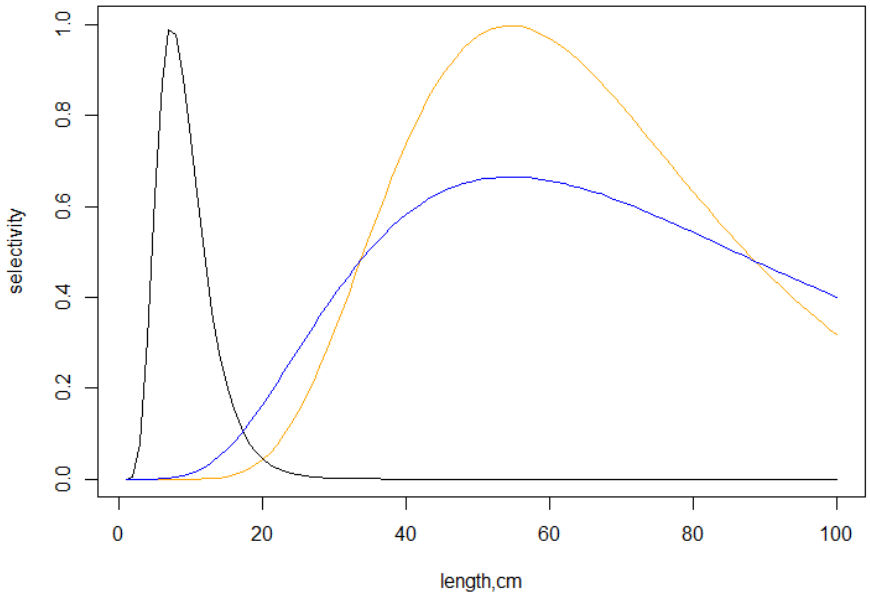


Figure 6. Shape of the lognormal gear selectivity curve for an organism of length 1-100cm.

Black: lsm=2, sigma=0.4
Orange: lsm=4, sigma=0.4
Blue: lsm=4, sigma=0.6

If a change in selectivity is included in the simulation (using flagchangesel, YYY_changeSEL and other parameters, see above), then the *lsm* and *sigma* parameters after the change are set at

lsm + SELchange_addlsm_YYY
sigma + SELchange_addsigma_YYY

15.8.8. Gamma length-based selectivity curve (selcurve=5)

This option applies a gamma selectivity curve and is defined by two parameters – the length at highest selectivity *lsm* (YYY_gammalsm, given in cm) and spread of the selectivity curve *sigma* (YYY_gammasigma). The selectivity of age group *i* is calculated as

$$psel_i = \left(\frac{len_i}{lsm}\right)^{\frac{lsm}{\beta}} \cdot \exp\left(\frac{lsm - len_i}{\beta}\right)$$

where

$$\beta = \frac{\sqrt{lsm^2 + 4 \cdot sigma^2} - lsm}{2}$$

where len_i is the length of the age group i. The figure 7 shows the shape of the gamma selectivity curve given three values of the *sigma* parameter.

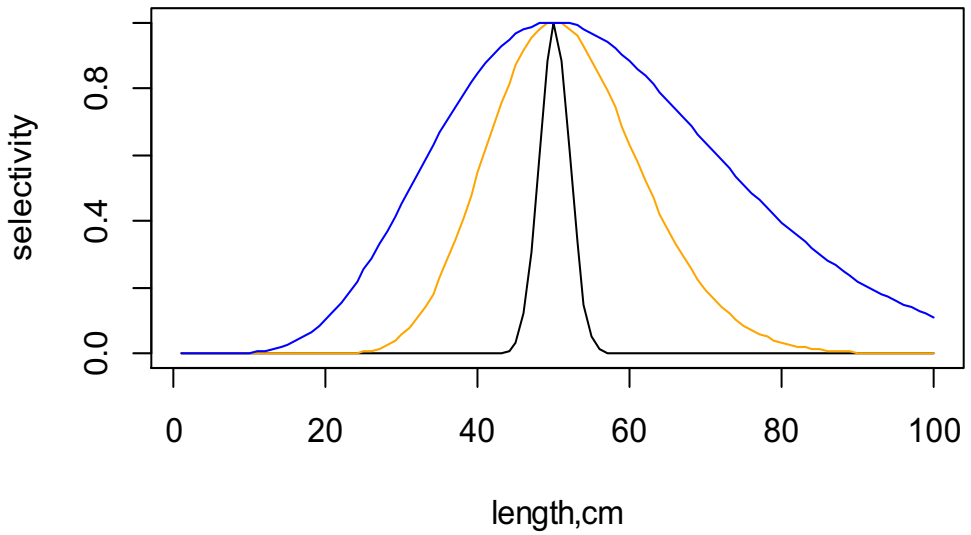


Figure 7. Shape of the gamma gear selectivity curve for an organism of length 1-100cm. The highest selectivity point (*lsm*) is set at 50cm.

Black: sigma=2
Orange: sigma=10
Blue: sigma=20

If a change in selectivity is included in the simulation (using flagchangesel, YYY_changeSEL and other parameters, see above), then the *lsm* and *sigma* parameters after the change are set at

lsm + SELchange_addlsm_YYY
sigma + SELchange_addsigma_YYY

15.8.9. Knife-edge length-based selectivity curve (selcurve=6)

The knife-edge selectivity means that selectivity is 0 below a critical species specific length and 1 above this length. To avoid adding even more parameters the sel_XXXjuv and sel_XXXad are re-used. In this option the sel_XXXjuv sets the threshold length for the species, below which the selectivity is zero. The sel_XXXad gives the selectivity value (from 0 to 1) for individual longer than sel_XXXjuv.

If there is a selectivity change, the new selected proportion (i.e. the selectivity on any individuals greater than the threshold length) will be calculated as

sel_XXXad + SELchange_addlsm_YYY

15.8.10. Bimodal length-based selectivity curve (selcurve=7)

This option applies a bimodal selectivity curve and is defined by three parameters – the right side length at highest selectivity *lsm1* (YYY_bilsm1, given in cm), the – the left side length at highest selectivity *lsm2* (YYY_bilsm1) and spread of the selectivity curve *sigma* (YYY_bisigma). The selectivity is calculated as the co-joined set of two normal selectivity curves, each with its own length of peak selectivity – at any given length, the curve giving the higher selectivity value is used.

The selectivity of age group *i* is calculated as

$$psel_i = \max\left(\exp\left(\frac{-(len_i - lsm1)^2}{2 \cdot sigma^2}\right), \exp\left(\frac{-(len_i - lsm2)^2}{2 \cdot sigma^2}\right)\right)$$

where len_i is the length of the age group *i*. The figure 8 shows the shape of the selectivity curve given three values of the *sigma* parameter. Note that same *sigma* (spread) parameter is applied to both *lsm1* and *lsm2*.

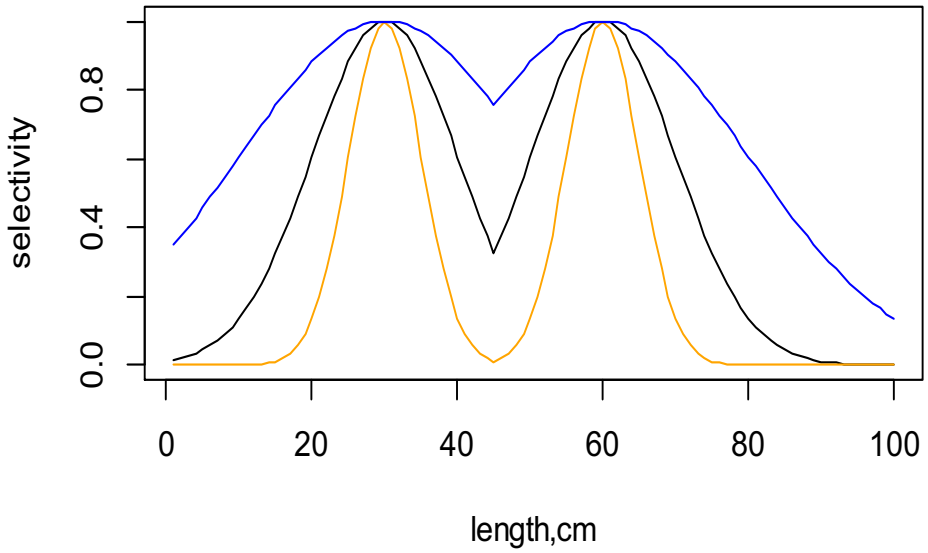


Figure 8. Shape of the bimodal gear selectivity curve for an organism of length 1-100cm. The highest right and left side selectivity point (*lsm1* and *lsm2*) are set at 30cm and 60cm

Black: sigma=10
Orange: sigma=5
Blue: sigma=20

If a change in selectivity is included in the simulation (using flagchangesel, YYY_changeSEL and other parameters, see above), then both *lsm1* and *lsm2* are changed by adding the same SELchange_addlsm_YYY so that after the change takes place the parameters are

lsm1 + SELchange_addlsm_YYY
lsm2 + SELchange_addlsm_YYY
sigma + SELchange_addsigma_YYY

15.8.11. Binormal length-based selectivity curve (selcurve=8)

This is the most complex length-selectivity option. It allows for a curve with two peaks (YYY_bilsm1 and YYY_bilsm2), each with it's own spread (YYY_bisigma and YYY_bisigma2), and with stronger peak selectivity at one of the peaks set using an amplification scalar *ampli* (YYY_ampli).

The selectivity of age group *i* is calculated as

$$psel_i = \max\left(\exp\left(\frac{-(len_i - lsm1)^2}{2 \cdot sigma^2}\right), \left(ampli \cdot \exp\left(\frac{-(len_i - lsm2)^2}{2 \cdot sigma2^2}\right)\right)\right)$$

where len_i is the length of the age group *i*. The figure 9 shows the shape of the selectivity curve.

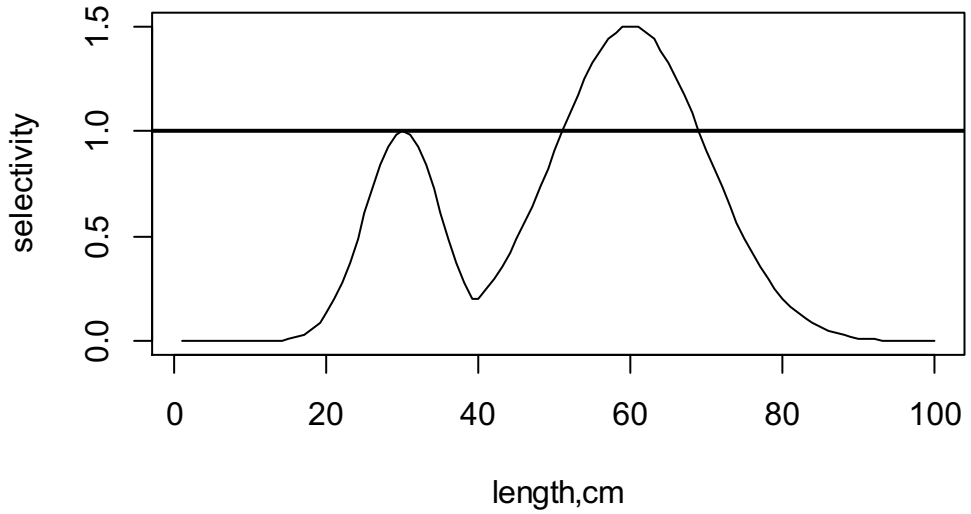


Figure 9. Shape of the binormal gear selectivity curve for an organism of length 1-100cm.

The highest right and left side selectivity point (*lsm1* and *lsm2*) are set at 30cm and 60cm, sigma1=5, sigma2=10 and the amplification = 1.5. The horizontal line shows that the selectivity is always capped at 1.

If a change in selectivity is included in the simulation (using flagchangesel, YYY_changeSEL and other parameters, see above), then both *lsm1* and *lsm2* are changed by adding the same SELchange_addlsm_YYY and both *sigma* and *sigma2* are changed by adding the same SELchange_addsigma_YYY so that the final parameters used are

lsm1 + SELchange_addlsm_YYY
lsm2 + SELchange_addlsm_YYY
sigma + SELchange_addsigma_YYY
sigma2 + SELchange_addsigma_YYY

15.8.12 Volume swept by the fishing gear

The volume of water that is subjected to fishing at one effort unit (day per day) is determined by the gear swept volume. This is set by the YYY_sweptarea parameter, which gives the volume of water (in m^3) swept by one effort unit of the entire fishery (note that even though the parameter name says "area" it actually gives the volume and not the area swept). For recreational fishing the effort is given not per fishery but **by the fisher**, so the swept volume is given in m^3 per day per fisher. Because effort is set in days per day, the YYY_sweptarea parameter gives the volume swept in one day in one box by the entire fishery (or a fisher of the recreational fishing).

To convert the swept volume per unit effort of fishing pressure to fishing pressure per day per m^3 Atlantis divides the total swept volume (in m^3 per day per box) by the box volume available to the fishery based on its access to habitats, i.e. the $prop_{fish}$ value. This is the same proportion of accessible box volume that is used in the calculation of the habitat scalar, described in chapter 15.7.3.

$$F_{m3} = \frac{volume_{swept}}{volume_{box} \cdot prop_{fish}}$$

This means that the parameter YYY_sweptarea gives the volume of water swept by the gear in the available habitats.

NOTE!

Ensuring meaningful representation of fisheries habitat access and vertical effort distribution

Note, that the habitat dependent $prop_{fish}$ scalar is applied to each layer of the box and not to the bottom layer only. This means that the users have to make sure that the vertical distribution of fishing effort and habitat access by fisheries makes sense. The habitats probably should have no effect on fisheries that operate in pelagic areas only. In this case the habitat_YYY should be set to 1 for all habitats to ensure that the fishery has access to the entire volume of the cell. The habitat_YYY vector makes sense for demersal fisheries only and the users should make sure that the vertical distribution of the fishing effort is also concentrated (or occurs entirely) in the bottom water column layer.

The concept of "swept volume" is used broadly here and applied to all kinds of gear. It can be seen as the area affected by fishing (see NOTE! in the chapter 15.4 on how catch is calculated in dynamic fishing). The fishing gear can also represent hooks or traps, but the total volume affected per day may require careful thought as it actually represents the volume of water that mobile animals move through to encounter the gear during a day's fishing (or its soak time during a day's fishing).

15.8.13. Changes in the YYY_sweptarea parameter

Like other fisheries parameters, it is possible to include temporal changes (scenarios) in the volume swept by the fishing gear (see chapter 15.10 for further explanations around changing harvest related parameters).

Changes in swept volume are done by first turning on a global flag flagchangeswept=1 and then indicating which fishery has changes via the YYY_changeSWEPT parameters. The number of changes for each fishery are given in the YYY_swept_changes parameter, the start day of each change for each fishery is given in the SWEPTchange_start_YYY vector. The periods over which each change occurs is given in the SWEPTchange_period_YYY vector and the multiplier used to get the final value at the end of the change period is given by the SWEPTchange_mult_YYY vector.

15.8.14. Fishing gear conflict

Fisheries gear conflict value is calculated in Atlantis for reporting purposes, but has no effect on the model dynamics. The gear conflict between a pair of fisheries is calculated as a proportion of the box area in REEF, SOFT and FLAT physical habitats where **both** fisheries fish (identified with habitat_YYY), multiplied by the gear conflict parameter gear_conflict_YYY (which gives conflicts between fishery YYY and each of other fisheries) and multiplied further by the pelagic-demersal overlap (which is 1 when both fisheries are pelagic or demersal and 0 in other cases).

Table 8. Parameters used to setup fishing gear selectivity and swept area.

Parameter	Description
YYY_selcurve	Shape of the gear selectivity curve for fishery YYY
flagchangesel	If set to 1, fisheries gear selectivity changes through time (see chapter 15.10 and 15.8.1 for further explanations)
sel_XXX sel_XXXjuv sel_XXXad YYYsel_lsm YYYsel_b YYY_lognormlsm YYY_lognormsigma YYY_gammalsm YYY_gammasigma YYY_bilsm1 YYY_bilsm2 YYY_bisigma YYY_bisigma2 YYY_ampli	Parameters used to set fisheries gear selectivity. They can have different meanings depending on the YYY_selcurve option. Check chapter 15.8.1-15.8.11 for details
YYY_sweptarea	Volume of water that is subjected to fishing at one effort unit (in m^3)
flagchangeswept	If set to 1, volume swept by a fishing gear can change through time. See chapter 15.8.13 and 15.10 for further details
gear_conflict_YYY	A parameter used only for reporting purposes, and does not affect Atlantis calculations (see chapter 15.8.14).

15.9 Fisheries discarding

15.9.1 Key factors determining discarding

In most cases, each fishery will have some level of catch discarding. In Atlantis the catch per group per cell is calculated first and then some proportion of this catch can be discarded depending on the discarding parameters. Calculations of discarding are done in the *Get_Discards()* routine in **atHarvestDiscards.c**

Discards are affected by four factors:

1) **Whether a species is under quota, i.e. total allowable catch (TAC) or not**. The initial TAC is set in the TAC_XXX parameter vector (kilograms per year), but this value can be modified depending on the management options chosen (see Chapter 16). If no TAC is wanted the TAC_XXX value should be set to 10^{12} – this means a species for a given fishery is a no quota species.

   When management options are in place and a fishery is managed through TACs (YYY_flagmanage>1), AND a species is set as isTAC in the *functional_groups.csv* file then all

catch above the TAC will be discarded. This applies both for dynamic fishing and the user defined fishing mortality case, but NOT for the imposed catch option. However, if **a fishery is required to land all the take of TAC species, then no discarding will be allowed** (YYY_landallTAC_sp=1), and all catch above TAC will be included in the reported catch. Note that when YYY_flagmanage>1 the management itself will reduce or close the fishing once the TAC has been reached; although, depending on the management option chosen, this may not come into effect immediately (see chapter 16).

2) **Cumulative weekly catch limit**, called Trip Limit in Atlantis, set in the TripLimit_XXX parameter for each fishery (kilograms per week). This value sets the weekly allowed landing – if the catch exceeds this value, ALL catch above the weekly limit will be discarded, **regardless of other management and discarding options**. To turn the weekly trip limit off, set it to 10^{12}. However, the weekly trip limit will not apply if a fishery is required to land all the take of any quota species (YYY_landallTAC_sp=1) and its quota for that species is $<10^{12}$ (i.e. meaning that a fishery is operating under quota for that species). This means that YYY_landallTAC_sp=1 applies to both TAC and weekly trip limit landings.

3) One of the five available **discarding formulations** chosen (see below). They are set with the flagdiscard_XXX parameter vector, which must have as many values as there are fisheries. These five options are exclusive, so the user can only apply one option for one species-fishery combination. Note that the TAC and weekly catch limit options described above take precedence over the the discard calculated in these five options, so if the TAC and weekly limits are applied and they result in more discards than the discard option chosen for the fishery then the higher value will over-ride the discard option result.

4) Whether the fishery will also do **optional highgrading and market based discarding** (see below). These occur in addition to other forms of discarding (remember that if the TAC and weekly limits would see more discarding they take precedence). There is no high grading or market based discarding if YYY_landallTAC_sp=1.

Atlantis allows for survival of discarded catch. This is important, because if a user allows the survival of discards, the discarding will affect mortality due to fishing. The survival of discard is set through incidmort_XXX parameter, which gives the **proportion of discards that die** after discarding (note, this is incorrectly described as proportion of discard surviving in some harvest parameter files). The discards that are dead are sent to the Carrion (DC) pool.

Forced changes in discarding parameters, described in chapter 15.9.2, are not applied to highgrading and market based discarding, but they can change dynamically depending on whether TACs (for highgrading) and sale prices (for market based discarding) are allowed to change.

15.9.2. Temporal changes in discarding practices

As with other fishing parameters, a user can set up temporal changes in discarding practices. This is done using a global flag flagchangediscrd=1, setting up species specific parameters indicating for which species and fishery combination discarding changes (flagchangeDISCRD_XXX), how many changes will occur (XXX_discard_changes), the start date of each change (DISCRDchange_start_XXX). Three additional specific parameters for changes in the discarding are also required - DISCRDchange_discardmult_XXX indicating changes in discarding proportions (see below),

DISCRDchange_retainmult_XXX indicating changes in the multiplier for illegal retention of undersized animals (see below), and DISCRDchange_threshmult_XXX indicating a multiplier for the discarding threshold in size based discarding formulations. See below for information on how these multipliers are applied in each of the five discarding options.

Note, that temporal changes in discarding are based on species and not fisheries. Also, it is very important to remember that **changes in the discarding formulation do NOT have the period over which the changes are applied**. This means that the multiplier takes the full effect (no gradual change) from the first day of the change and also that the implemented **change remains for the rest of the simulation**. If a user wants to return to the pre-change discarding values, then two changes must be set up, where the second change multiplier is set as (1/first change multiplier).

15.9.3. Discarding waste

Regardless of the form of discarding used (options listed below), a user can also set up a proportion of the legal sized catch that is discarded as waste (such as guts, skin). This proportion is given in k_waste_XXX and gives values for each fishery; the proportion is the same for all age groups.

The proportion of biomass that is discarded as waste can change through time in a simulation. This is set through the DISCRDchange_wastemult_XXX. This multiplier will apply to the original waste discarding proportion.

15.9.4. Constant discarding (flagdiscard=0)

This is the simplest discarding formulation. It is based on a constant discarding proportion set with the FFCDR_XXX parameter (which must have as many values as there are fisheries). In this option the same discarding proportion applies to each age class, and it is equal to FFCDR.
If discarding changes are applied (set in discard change, see below) then, once the changes take place, the FFCDR will be multiplied by the DISCRDchange_discardmult_XXX parameter.

15.9.5. Constant proportion of an age group discarded (flagdiscard=1)

This option is similar to the option above, except that proportions of discarding in each age class can be set. Since it would be too difficult to set age-specific discarding proportions for each species for each fishery (e.g. 10x50x30 values), Atlantis allows two alternative formulations for age-specific discarding. They are set up FFCDR_XXXchrt and FFCDR_XXXchrtB and each of them must have as many entries as there are age groups in the species XXX.

So, for example, a user might decide that for a species XXX a fishery can either discard mostly young individuals or mostly old individuals. In this case the

FFCDR_XXXchrt for a five age class group might look like
0.5 0.3 0.2 0.1 0.0
whereas
FFCDR_XXXchrtB might look like
0.0 0.1 0.2 0.3 0.4

The parameter FC_caseXXX then selects which of these two discarding formulations is used for each fishery (0 - FFCDR_XXXchrt is used, 1 - FFCDR_XXXchrtB is used).

Like in the option above, if discarding changes are applied then the proportion to be discarded is multiplied by the DISCRDchange_discardmult_XXX parameter.

15.9.6. Length-based discarding (flagdiscard=2)

This option allows for a more realistic length-based discarding option to be used. Here catch below a threshold minimum length will be discarded. Because length is a dynamic property, changing through time, it means that the amount discarded will also change dynamically through time. The minimum size of catch is set by the FCthreshli_XXX parameter, giving the minimum length of species XXX that each fishery can catch. All catch below this length will be discarded.

It is possible for fisheries to retain some of the undersized catch illegally. This is set in the k_retain_XXX parameter giving a fishery-specific proportion of the undersized catch that will not be discarded (so if it is set to 1, then no undersized catch is discarded at all).

In summary, in length-based discarding, Atlantis will make these steps:

1) *Is the caught age-group below the size limit?* If yes, go to step 2. If no, go to step 3.
2) *Is there illegal retention of undersized animals?* If yes, discard only the proportion of the undersized group (1-k_retain_XXX). If no, discard all.
3) *Is there any discarding of legally sized animals as waste*? If yes, discard the k_waste_XXX proportion of legally sized biomass. If no, keep all.

If changes in discarding parameters are applied then two options can change:

1) The minimum size threshold can change and it is set through DISCRDchange_threshmult_XXX. This **multiplier will apply to the original size** given in the harvest.prm file, so if several changes are applied, make sure the multiplier refers to the original size.
2) The proportion of undersized fish that is retained illegally can also change and is set through DISCRDchange_retainmult_XXX. This **multiplier will apply to the original retain** proportion given in the harvest.prm file.

15.9.7. Discarding imposed through time series (flagdiscard=3)

This option requires the user to provide time series of imposed discards. See Chapter 8 on how to setup forcing time series files. The TS file gives a time series of discards in terms of total biomass (mg s^{-1}) discarded for a specified species for a single box. The age distribution of discards for age-structured groups and age-structured biomass pools is given in the DiscardTS_agedistribXXX parameter (which must have as many values as there are age group in species XXX).

Since discard biomass is given per box, the final value of discards of species CX by fishery Y in $mgNm^{-3}$ and per age group *i* is calculated as

$$Disc_{Y,i,CX} = (Disc^{*}_{Y,CX,j} \cdot \delta_{depth,Y} \cdot activesc_{Y} \cdot p\_age_{Y})/volume_{z}$$

where *Disc** is the imposed discard time series per box j species CX and fishery Y; $\delta_{depth,Y}$ is the vertical distribution of fisheries activity Effort_vdistribYYY needed to convert the discards per box to discards per layer; $activesc_Y$ is the scalar of 0.5 applied if the fishery YY is active during both day and night (indicated by flagYYYday=2), otherwise the scalar is 1, as the default assumption is that a fishery is only active at day or at night; p_age_Y is the age specific distribution of discards DiscardTS_agedistribXXX; and $volume_Z$ is the volume of layer Z in box j.

15.9.8. Density dependent discarding (flagdiscard=4)

This discarding formulation includes two different options, both set through the FCcocatch_XXX parameter (which must have as many values as there species in the *functional_groups.csv* file).

1) When catches are imposed through time series (through flagimposecatch), flagdiscard_YYY=4 allows the co-catch and discards of species typically caught alongside species for which catch is imposed (i.e. for each n tonnes of species A caught then o tonnes of species B is also caught/discarded). Atlantis assumes here that **all** cocatch of species is discarded. In this case the FCcocatch_XXX indicates how much of species XXX is **caught and discarded** for the unit of imposed catch of each species listed in FCcocatch_XXX. So, the vector of values in FCcocatch_XXX indicates the amount of XXX caught and discarded due to the imposed catch of each other modelled species.
2) When catches are not imposed, but are modelled through a user defined fishing mortality (mFC) or dynamic fishing then when flagdiscard_YYY=4 the FCcocatch_XXX will indicate how much biomass of each other species is caught and discarded per unit of species XXX biomass caught. This means that in contrast to the option above, here species XXX will not be discarded as co-catch but all other species given in the FCcocatch_XXX parameter will.

15.9.9. Highgrading

Highgrading means that when a fishery is approaching a quota limit some catch will be discarded. This option is turned on using a global flag flaghighgrading=1. If this is turned on, highgrading will start when a proportion set with a global prop_within parameter of quota has been reached (the same parameter applies to all fisheries).

The highgrading will apply to the **youngest cohorts** that make up a proportion of the catch smaller than set in the global highgrade_thresh parameter. So, for example, if highgrade_thresh = 0.2, and the age group 1 makes up 5% of total catch, age group 2 – 10%, age group 3 – 15%, age group 4 - 55% and age group 5 – 15%, then highgrading (complete discarding) will apply to age groups 1, 2 and 3. Note that even though age group 5 makes up less than 20% of the total catch, it is not discarded, as highgrading applies to youngest age groups, up to the threshold limit.

Highgrading **is not** applied for imposed catch options. Highgrading is also **not** applied if fishery must keep all of its catch (YYY_landallTAC_sp=1).

15.9.10. Market based discarding

Market based discarding is optional and is done in addition to all other discarding described above. It is turned on with a global flag flagmarketdiscard=1.

Market based discarding **is not** applied for imposed catch options and if fisheries must land all catch (YYY_landallTAC_sp=1).

Market based discarding should be done only when the Economics submodel is active and initial sale prices of species for each fishery are provided in the *economics.prm* file (XXXsaleprice parameter for each species and fishery) and are determined dynamically during the course of the run. If market based discarding is turned on without the Economics submodel, Atlantis will not find the sale price parameters and will quit.

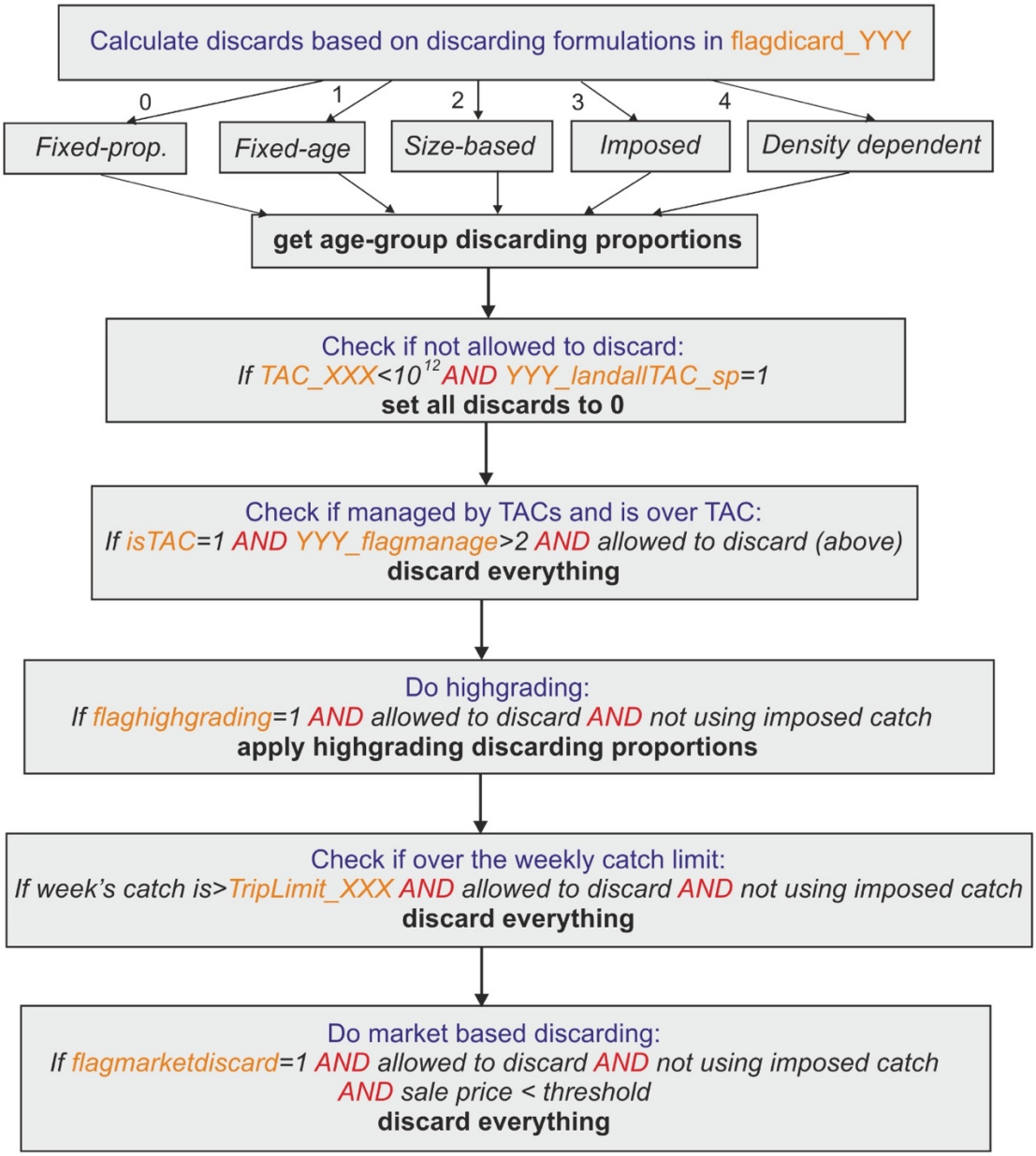
In using the market based discarding, a species is discarded if its sale price falls below the threshold of the maximum sale price. The threshold is a global parameter salethresh in the *harvest.prm* file and the maximum sale price is specific to each fishery YYY_maxsaleprice. The maximum set price can be either fixed or updated dynamically (UpdateMaxSalePrice=1) in the course of the run.

Table 9. Parameters used to setup fisheries discarding.

Parameter	Description
flagdiscard_XXX	Species specific flag setting up discarding options for each species/fishery combination.
incidmort_XXX	Proportion of species XXX discards that **die** after discarding (so if set to 1, then all discards die)
flagchangediscrd	If set to 1, discarding practices change through time. See chapter 15.9.2 and 15.10 on how to set up temporal changes.
k_waste_XXX	Proportion of legal catch that is discarded as waste (guts, skin). This proportion cannot survive but is sent to DC pool.
FFCDR_XXX	Proportion of species XXX catch that is discarded, in constant discarding formulation (when flagdiscard_XXX = 0). Same discarding is applied to all age classes.
FFCDR_XXXchrt FFCDR_XXXchrtB	Vectors giving two possible age specific discarding proportions when flagdiscard_XXX =1. Must have as many values as there are age groups in species XXX. The FC_caseXXX vector sets which of the two alternative formulations are used by each fishery (must have as many values are there are fisheries)
FCthreshli_XXX	Minimum length of species XXX that can be retained when length based discarding is used (flagdiscard_XXX = 2)
k_retain_XXX	A vector of values giving fishery-specific proportion of the undersized catch that will not be discarded
DiscardTS_agedistribXXX	Proportional distribution of forced discarding over age groups (when forced discarding is used, flagdiscard_XXX =3)
FCcocatch_XXX	Density dependent discarding parameter, that can have different meanings depending on whether imposed or set as user defined or dynamic mortality (see chapter 15.9.8)

flaghighgrading	If set to 1, fisheries will start discarding the youngest cohorts of the catch when they are approaching the TAC. The proportion of TAC fulfilled before highgrading starts is set in the global prop_within parameter
highgrade_thresh	The proportion of catch an age group must make up, before it is retained during highgrading (see chapter 15.9.9)
highgrade_thresh	If set to 1, discarding will be done according to the saleprice of fish species (set in XXXsaleprice). Should only be done when the Economics submodel is active, or else Atlantis will quit.

Figure 10. The main algorithm in the *Get_Discards()* routine explaining what determines the final discard proportion of a given species and age group by a given fishery.



15.10 Temporal changes in harvest parameters

Atlantis allows users to set up different fishing scenarios through temporal changes in fishing parameters.

Historically, during Atlantis development, there have been changes in the way these temporal changes are implemented. In some cases changes take place within one time step and in other cases they happen gradually over a "change_period" number of days.

In general, Atlantis implements changes in fishing characteristics through 4 parameters for any given characteristic:

1) number of changes for the parameter in each fishery-species combination (e.g. XXX_q_changes shows number of catchability changes for species XXX in an array of all fisheries),
2) day of the model run that the change starts, e.g. Qchange_start_XXX. This parameter must have as many entries as there are changes for species XXX. Note that even though XXX_q_changes gives the number of changes for each species and fishery combination, in reality only one fishery per species can have those changes. This is because the parameter Qchange_start_XXX only has as many values as there are changes (rather than allowing different values for different fisheries); entering multiple changes for multiple fisheries would result in a two-dimensional array and become too complicated to enter easily. If such a change is required please contact the model developers.
3) number of days over which each change is gradually (linearly) implemented (Qchange_period_XXX) – the change will take full effect at the end of this change period. The change period is not available for all parameter changes – some of them (e.g. selectivity) change in one time step, set by the date of change (this is because they represent a policy or technological change which typically occur more suddenly than behavioural changes).
4) the multiplier or a constant that applies to each change in the parameter by the end of the change period (Qchange_mult_XXX). This multiplier applies to the ORIGINAL values of the parameter given in the parameter files and NOT to the value from the previous changes. After the end of the change period, the implemented change is applied for the rest of the simulation, or until new changes apply. **This means that the parameter value does NOT revert to the original value after the end of the change period!** However, changes in the fishing effort can be pulsed (see chapter 15.5.13), which means that, if pulse change is requested, after the change period the effort will return to the original pre-change values.

This typical approach is implemented in the *Util_Get_Change_Scale()* routine in **atUtil.c**, but not all changes are applied through this routine.

The various changes have been described in separate chapters, and this section only provides a general summary on how fisheries options can be changed.

Table 10. Parameters and description of main fisheries features that can change through time.

What can change?	Global flag	Fisheries-specific flags	Comments
Fisheries gear selectivity	flagchangesel	Change occurs in one time step and remains for the rest of the simulation or until the next change	Chapter 15.8
Swept volume by the fishing gear	flagchangeswept	Change occurs over a set period and remains in place after the change period – typical 4 parameter description	Chapter 15.8.13
Fishing mortality set through flagF	flagchangeF	Change occurs over a set period and remains in place after the change period – typical 4 parameter description	Chapter 15.3.4
Total effort in dynamic fishing	flagchangeeffort	Change occurs over a set period, but pulsed changes are allowed. If pulse changes are applied the effort after the change period reverts to the original value	Chapter 15.5.13
Discarding	flagchangediscrd	Change occurs in one time step and remains for the rest of the simulation or until the next change	Chapter 15.9.2
Catchability	flagchangeq	Change occurs over a set period and remains in place after the change period – typical 4 parameter description	Described in the example above
Effort caps – whether effort per cell is capped	flagchangecap	Change occurs over a set period and remains in place after the change period – typical 4 parameter description	Chapter 15.6.2
Populations of ports – relevant to recreational fishing models and Economics submodel	flagchangepop	Change occurs over a set period and remains in place after the change period – typical 4 parameter description	Chapter 15.5.4
Fisheries spatial coverage	flagchangep and YYY_changeP	Changes in spatial coverage are applied through the habitat overlap	Chapter 15.7.3
Length of the fishing season	YYY_changeseason	Fishing season can be reduced if biomass of fishing stocks or endangered species fall below a threshold level	Chapter 16.5
TAC of a fishery	YYY_TACchange 0 – no change 1 – dynamic change 2 – forced change	Forced TAC change is implemented in the same way as other parameter change – typical 4 parameter description Dynamic change in TAC is implemented when some form of stock assessment is applied (see box in chapter 16.1)	Chapter 16.2.10

16. MANAGEMENT

16.1. General introduction to management

Fisheries management is an optional feature in Atlantis. Its role is to regulate the fishing effort according to a set list of rules and options, such as ensuring that stocks don't fall below a certain threshold, implementing total allowable catch (TAC) rules, or protecting endangered species. Management comprises various measures of effort reduction through limitation on fishing and spatial closures (temporal or permanent). The limitations on fishing typically start applying once a set trigger has been reached, for example total allowable catch (TAC) has been reached, stock is below a threshold biomass, a certain amount of engendered species biomass has been caught, and so on (see full list below). Annual management actions such as setting TACs or scaling the effort if stocks decline below a set level occur on the first timestep of each year and are typically in place for the entire year. In some cases (stock adaptive management), the re-evaluation of the fishing restriction can be done earlier than the start of the next year.

The management options available to regulate fishing mortality depend on whether fishing is done through imposed catch, user defined fishing mortality or dynamic fishing.

For harvesting set through **imposed catch** the management options include

1) ***Stock adaptive management*** (set with YYY_flagmanage=1). In this option the actual catch taken will be decreased if stocks fall below a set threshold (see below). Note that even though YYY_flagmanage has many options, only YYY_flagmanage=1 can be applied to the imposed catch. If YYY_flagmanage > 1, no management scalar will be applied to the imposed catch
2) ***Spatial management*** through MPAs (set if flagmpa=1)

For harvesting set as a user defined **fishing mortality** the management options include

1) ***Yearly and weekly catch limits*** set in TAC_XXX and TripLimit_XXX parameters and the flag_stop_F_tac parameter, which determines whether fishing is closed when limits are reached (see chapter 15.9.1).
2) ***Spatial management*** through MPAs (set if flagmpa=1)
3) Harvest control rules set as "***broken stick" scalar*** on the mFC value (see chapter 16.2.8)
4) ***Frame based management***, where fishing is allowed once biomass has passed a threshold level (fishing level is set by TAC_XXX and the biomass threshold is set as a scalar on the initial biomass using FC_high_thresh)

For **dynamic fishing** the management includes

1) ***Stock adaptive management*** or ***TAC management*** (set with YYY_flagmanage)
2) ***Spatial management*** through MPAs (set if flagmpa=1)
3) Seasonal closures (set in YYY_flagseasonal)
4) Control rules based on the bycatch of endangered species
5) Extra options that come through the economic incentives if the Economics submodel is used

The adaptive or TAC based management, and management based on endangered species controls the

overall effort or fishing intensity by the fishery, i.e. it is not spatial. In contrast, MPA applies spatial management, by closing a proportion of a box to fishing. Finally, seasonal closures are temporal management, which means that fishing effort is stopped for a set period of time within each calendar year.

Note!

Management, stock assessments and dynamic TACs
A lot of management decisions are based on TACs – when a fishery exceeds its TAC (counted in one of many different ways) it will be closed. It is important to understand that the TAC can be either set in the *harvest.prm* file or calculated dynamically during the simulation process. The dynamic calculation aims to simulate fisheries stock assessments and the TAC allocation process made in real life situations. Dynamic TACs require that one of three available stock assessment options are turned on:

1) **Pseudo assessments** (done if pseudo_assess=1 in *harvest.prm*) are the simplest version of stock assessments. It uses Atlantis biomasses and a random error added to them as inputs into the fishing mortality estimator and harvest control rule calculations.
2) **Full integrated assessment**, done in the **Assessment** submodel. This is done only if the *run.bat* file has a set –a parameter with *assessment.prm* parameter file. A number of assessment options are possible: Schafer production model, ADAPT , MSVPA Integrated assessment model (CAB) and R-based assessment calls (please see the wiki).
3) **Recommended biological catch based assessment** using tiered harvest control rules (applied if useRBCTiers=1 in *harvest.prm*). The tier based assessment follows the harvest control rule hierarchy used in Australia and is specified in the *CallTierAssessment()* routine. It includes a (i) full quantitative population model based (e.g. SS3), (ii) catch curves, (iii) CPUE, (iv) surplus production, (v), average length, (vi) trigger points, and (vii) catch composition. The parameters for these options are specified in *assessment.prm*.

The stock assessment options are not further described in the manual. For further details on the application of different assessment options see Fulton et al 2007 or Dichmont et al 2017.

16.2. Stock adaptive and TAC-based management

Stock adaptive and total allowable catch (TAC) based management is set through flagmanage flags. These settings limit, or completely stop, the fishing effort if stocks fall below a set threshold (stock adaptive management) or if the TAC has been reached. To allow for this management for each fishery select the specific management option with the YYY_flagmanage parameter.

16.2.1. No management on effort or catch applied

When YYY_flagmanage = 0, the fishery will not be affected by stock size related management actions or TACs. The effort of the fishery is set based on effortmodel options only (or imposed catch or user defined fishing mortality) and will stay in place regardless of the stock abundance or quotas.

16.2.2. Adaptive management based on reference limits of stock biomass

When YYY_flagmanage = 1, fishing effort is controlled by the stock abundance. The effort reduction is triggered when the stock falls below the level given by the FC_threshold_XXX – a proportion of the "virgin" biomass. This virgin biomass can be either the biomass at the start of the run or after a burn-in period. To setup the burn-in period set the global flagreinitpop=1 and give the day (in the days of simulation run) which should be used as the point in the simulation to define virgin biomass in the reinit_pop_day parameter. The burn-in period might be useful if biomasses fluctuate a lot during the early years of the simulation and are not useful as a stock management reference point.

If the biomass of a species XXX falls below the FC_threshold_XXX proportion, the effort will be reduced by the FC_restrict_XXX scalar and reductions will take place over the XXX_FC_period number of days. Remember, that the catch will be reduced via the mechanism of reducing effort (if using the effortmodel options) or the imposed catch. The assessment of the stock level is **done once per year** and the effort scalar is set and applied for the rest of the year. If more frequent assessments are desired, a user can set the second period after which the stock health will be assessed (XXX_FC_period2). If, after the time set in XXX_FC_period2, the biomass of a species XXX is above the FC_high_threshold_XXX proportion of the virgin biomass, the reduction in effort is lifted and the effort reverts to the original value before the restrictions. Note, the FC_high_threshold_XXX is often larger than the FC_threshold_XXX, which means that after a fishery has had effort restrictions imposed, a stock must recover to higher levels to allow for a return to the original effort.

16.2.3. Overview of the TAC based management

All other options of effort management involves some form of TAC or other catch constraint and apply complete closures of relevant fishery(ies) when it(they) exceed the allocated TAC. The management options are set through YYY_flagmanage = 2 to 8 and only apply when fishing mortality is set through dynamic fishing. Unlike the stock adaptive management, where stock status is evaluated once per year, assessment of whether a fishery has reached its TAC is done **at every time step** of the simulation. The original TACs for all relevant species and fishery combination are set in the TAC_XXX parameter. However, these values may be modified if TACs for a fishery are based on several species, such as in the case of companion species or basket species quota. Also, TACs can be updated dynamically if stock assessment options are included in the model (see NOTE! below).

The management response to reaching the quota will depend on whether the TACs are set on a global level or for different fisheries regions. The options for the allocation of a TAC to single or multiple species, and at global or regional levels, defines seven options for the YYY_flagmanage parameter, shown in Table 11.

The cumulative catch compared against the TAC can be calculated either based on landed catch only (flagTACincludeDiscard = 0) or based on both landed catch and discards (flagTACincludeDiscard = 1). Only the catch (and discards) accumulated in the current calendar year are used in the TAC comparison (i.e. the cumulative catch is zeroed at the start of each new calendar year in the simulation).

All groups for which TACs are to be applied should have a value 1 in the isTAC parameter in the *functional_groups.csv* file, or else the TAC routine for the species is skipped. Also, all functional groups which are to be managed with TACs should be identified as isFished=1 in the functional_groups.csv

file; otherwise Atlantis overwrites their TACs with a very large number (10^{12}) and considers them as non-quota species.

The TAC calculations are done by the *TAC_Check()* routine in **atManage.c**.

NOTE!

How are TACs set and how can they change during the simulation?

The TACs for set for each species and fishery combination. They are initially set in the TACXXX parameter. In the simplest version the TACs are static and remain the same throughout the simulation. However, it is also possible to have forced or dynamic changes in the TAC.

Forced changes are implemented in the same way as other forced changes in Atlantis parameters, where users have to set the dates and magnitude of the change. This is described in chapter 16.2.9.

Dynamic changes are applied if users have at least one of the three available stock assessment versions applied. This means that Atlantis will try to simulate the real-world situation where stocks are assessed at set time intervals (typically annually) and the TACs are updated dynamically based on the stock status, set stock biomass targets and harvest control rules (see Note! in chapter 16.1)

Table 11. Options for the YYY_flagmanage parameter

	Species level TAC allocation			**Spatial TAC allocation**	
YYY_flagmanage value and its "name" in the code	*TAC: one species*	*TAC: companion species*	*TAC: basket quota*	*Global TAC*	*Regional TAC*
2 TAC_mgmt	X			X	
3 basketTAC_mgmt			X	X	
4 regionalTAC_mgmt	X				X
5 coTAC_mgmt		X		X	
6 coBTAC_mgmt		X	X	X	
7 RbasketTAC_mgmt			X		X
8 coBRTAC_mgmt		X	X		X

The YYY_flagmanage=10 sets the USA-style management that was applied in the early 2000s (see Kaplan et al. 2013 for an example study). It involved cumulative trip limits based off TACs (often regional), and once the TAC was exhausted then spatial management actions were triggered. This system was deemed to be complicated and was replaced with transferable quotas in 2011. The option is not described here in further detail as it is specific to the historical fisheries management of the West Coast of the US. If its use is desired for historical hindcasts consult Kaplan et al. 2013 for explanations.

16.2.4. Species level TAC allocation

The TAC for each species XXX and fishery is given in the TAC_XXX array, which must have as many values as there are fisheries. However, the TAC_XXX parameter can be used differently depending on whether a species is managed through a single or multiple species TAC.

1) **Single species based TAC (YYY_flagmanage =2 or 4)**

The TAC is set for one species and its TAC limits are assessed only against its cumulative catch (or only based on landings, if flagTACincludeDiscard = 0) for that year. If a fishery targets only one species XXX (set in target_YYY parameter), it will be closed when the TAC for the species XXX is reached. If a fishery targets several species, it will be closed when it exceeds the TAC for as many species as specified by the YYY_num_max_sp parameter (see other options below on how the TAC can still be modified); in this way multispecies fleet management can be mimicked.

How to setup simple TAC management?

1) Set isTAC to 1 in the *functional_groups.csv* file for the TAC managed species XXX
2) Set a fishery specific flag YYY_flagmanage=2 (or to another value, see Table 11)
3) Give TACs in the TAC_XXX parameter for the fisheries that catch species XXX and should be limited by its TAC (tons per year for the entire model domain)
4) Set the number of species for which the TAC must be meet/exceeded before a multispecies fishery is closed (YYY_max_num_sp)

Remember, the **management parameters apply to a fishery and not to a species**! If the user wants to apply a TAC for a given species and ensure that no fishing is conducted once the TAC for that species is exceeded, the species should only be targeted by one fishery and the YYY_max_num_sp for that fishery should be set to 1.

2) **Companion species based TAC (YYY_flagmanage =5, 6 or 8)**

This approach is used in situations where catch of one species result in bycatch of other species under management. In this case the original TACs, given in the TAC_XXX parameter (and presumably based on single species stock assessments) are modified based on the ratio of the companion species bycatch. For example, it is estimated that for each 100 tons of ling a fishery Y catches 20 tons of redfish, which is the ratio of 0.2. Let's say that based on single species assessments the TAC for ling is 200 tons, but for redfish it is only 10 tons. These values are then set in TAC_LIN as 200 and TAC_RED as 10. In the companion species TAC approach, Atlantis will recalculate the TAC for each species based on the ratio of bycatch and based on whether the TACs are marked as being based on the strongest or weakest (most vulnerable) species (set coType_XXX parameter, where =0 weakest, =1 strongest). So if coType_LIN = 0, it means that the TAC of ling will be decreased to 50 tons, since the co-catch ratio will lead to 10 tons of redfish catch, which is the TAC for redfish. If coType_LIN = 1, the TAC_RED will be increased from 10 to 40, because the fishery in fulfilling its 200 ton quota of ling and will also take 40 tons of redfish bycatch. See the box below for a step by step explanation on how to set the companion species management.

How to setup companion TAC management?

1) Set isTAC to 1 in the *functional_groups.csv* file for both target and companion species
2) Set a fishery specific flag YYY_flagmanage=5 (or other value, see Table 11)
3) Give TACs in the TAC_XXX parameter for each fishery for target and companion species
4) Set the number of species for which the TAC must be meet/exceeded before a multispecies fishery is closed (YYY_max_num_sp)
5) Set the maximum number of companion species (i.e. highest number of companion species across all species in the model). This is a global parameter and applies to all fisheries (K_max_co_sp), it is often set to 2. This basically tells Atlantis whether to bother reading in any of the other parameters and the size of the arrays to initialise before read-in.
6) For each species set their individual number of companion species max_co_sp_XXX. This needs to be less than or equal to K_max_co_sp. This controls the loops for the individual species so as to maximise speed of execution.
7) For each target species give the ID numbers (in the *functional_groups.csv* order) of the companion species in the co_sp_XXX vector. This parameter can be seen as a representation of which species usually school or occur together and hence will be caught together (fisheries data is sometimes available on this, either from catch composition records or observers). These parameters are vectors and must have as many values as the number given by the max_co_sp_XXX parameter (it will actually tell you that it needs K_max_co_sp entries, but it only uses the first max_co_sp_XXX. If you want to play it safe enter K_max_co_sp and if max_co_sp_XXX is less than K_max_co_sp fill up the spare entries either with -1 or a number larger than the ID of the companion species with the largest ID. So for example, if you only had one companion species occurs for species XXX and K_max_co_sp is set to 2, then the second value in the parameter co_sp_XXX should be set to -1, e.g.

   ```
   co_sp_XXX   2
   2  -1
   ```

   shows that only species ID 2 is caught together with species XXX. Alternatively, if across all the species the highest ID of any companion species was (say) 46. Then you could put

   ```
   co_sp_XXX   2
   2  48
   ```

   Either of these case will be treated in the same way – the second entry will never execute on the code, only the companion species with ID 2 will be processed.

8) Indicate whether the TAC is dictated by the weakest or strongest link among the companion TAC species in the coType_XXX parameter (=0 weakest, =1 strongest). See main text above for the explanation.
9) Give the assumed ratio of the catch of companion species accidentally caught by each fishery while targeting a species XXX in XXX_co_sp_catchZZ where ZZ is the number associated with the position of the companion species in the co_sp_XXX vector – so for the first species (ID 2 in the example given under (7) above) it would be XXX_co_sp_catch1, for the second species XXX_co_sp_catch2 etc. Due to the complexity involved in reading in multli-dimensional arrays K_max_co_sp vectors must be supplied for XXX_co_sp_catch – that is if K_max_co_sp is 3 then

XXX_co_sp_catch1, XXX_co_sp_catch2 and XXX_co_sp_catch3 must be given even if XXX has max_co_sp_XXX less than 3. Just fill the extra vectors with zeros. These vectors will never be used (as the code running the TAC setting only loops to max_co_sp_XXX) but it is too difficult to get the code to initially read only as many vectors out of the *harvest.prm* file as needed based on max_co_sp_XXX. Sorry for the inconvenience. Note that the values given fin the vector XXX_co_sp_catchZZ indicate how many kg of the companion species is taken for every 1 kg of the target species XXX taken. This vector must have as many values as there are fisheries. This means that different fisheries (gear, selectivity) will have a different ratio of companion species bycatch even when targeting the same species XXX.

10) The ratio of species cocatch given in the co_sp_catch_XXX parameter can stay fixed throughout the run or be updated dynamically according to the actual catches of the companion species by a fishery. If a dynamic value is wanted this is set with a global parameter flagdyn_coupdate = 1

3) Basket species TAC (YYY_flagmanage =3, 6, 7 or 8)

Basket species TACs are often issued for data poor species, where TACs are identified for different groups, such as demersal sharks or small pelagics. The user still provides a TAC_XXX for individual species and sets the routines as described in the box for individual species, but Atlantis will pool the TACs for all species listed as being in one basket into one large TAC and will only assess the overall catch of all species in the basket against this basket TAC. So for example if the basket TAC for three demersal sharks is 1000 tons, all three options below will give the same TAC (the example assumes only one fishery in the model)

Option 1:	**Option 2:**	**Option 3:**
TAC_sharkA 1	TAC_sharkA 1	TAC_sharkA 1
1000	0	0
TAC_sharkB 1	TAC_sharkB 1	TAC_sharkB 1
0	500	0
TAC_sharkC 1	TAC_sharkC 1	TAC_sharkC 1
0	500	1000

Irrespective of how the basket is made up, if the cumulative catch of shark (i.e. the sum of the catch of sharkA, sharkB and sharkC) exceeds the basket then the fishery is closed even if the cumulative catch is made up of a single species and the other species are largely untouched.

How to setup basket TAC management?

1) Set the size of a basket quota (the maximum number of groups in the quota) using K_num_basket in *run.prm*.
2) Set isTAC to 1 in the *functional_groups.csv* file for all the basket TAC species
3) Set a fishery specific flag YYY_flagmanage=3 (or 6, 7, or 8 if combined with other management options, see Table 11)
4) Give TACs in the TAC_XXX parameter for each fishery for species
5) Set the number of species for which the TAC must be meet/exceeded before a multispecies fishery is closed (YYY_max_num_sp)
6) Indicate which species are members of the basket TAC with basketSPXXX set to 1.

7) For each basket species XXX give the number of other species that are in the same basket using the basket_sizeXXX parameter.
8) Then provide ID numbers of the species that are in the basket with species XXX. This is given in the parameter basketTAC_XXX. If a species is not in a basket with any other species, put the ID number larger than the ID of the last species in the *functional_groups.csv* file. This means that if there are 41 functional groups and the last ID number is 40 (since Atlantis counts from zero!), then for a single quota species basketTAC_XXX should say 41 (or 42 or any other number greater than 40).
9) Make sure that the parameters above are consistent for all basket species, as currently Atlantis does not have internal checks for this. This means that if species A is in a basket with species B and C, the parameters for all three species should show that they are in the basket quota with each other.

16.2.5. Spatial TAC allocation: global or regional

TAC based management also allows users to either use a basic global TAC applied to the entire model domain (e.g. YYY_flagmanage=2, see Table 11), or use regional TAC management (e.g. YYY_flagmanage=4, see Table 11). The **regions can be defined based on fisheries stocks** identified in the biology.prm file **or based on management areas** (see NOTE! below). Parameter manage_reg identifies whether regional fisheries management is done based on stocks (manage_reg =0) or based on regions (manage_reg =1).

In global TAC cases, a species TAC applies to its total catch (or landing) over the entire model domain. In the case of the regional management TACs are assessed for separate stocks or regions independently (see NOTE! below on how to set up different stocks or regions). In this case the TAC for a species/fishery is still set as usual for the entire domain in TAC_XXX parameter.

If **regions are defined by stocks,** then allocation of the TAC among stocks is simply based on proportional biomass of each stock and is done once per year. If the regional management option is chosen (e.g. YYY_flagmanage=4) for the fishery YYY, then Atlantis will automatically apply stock specific TACs to all species that are caught by fishery YYY that also have several stocks identified (see NOTE! below). Note this means regional management is used regardless of the regionalSPXXX setting.

Alternatively, if TACs are based on **management areas**, then users must identify whether species XXX should be managed by regional quotas in the parameter regionalSPXXX (where a value of 1 means regional management is in place). Then the TACs (in tons per year) for each region are given in the RegTAC_XXX parameter, which must have as many value as there are regions or management areas. Note, even if some regions are managed through a TAC, it does not mean that ALL regions must participate in the TAC management. Such situations exist in real life, where for example, different regions belong to different jurisdictions (countries or states). To allow for this situation, regions that do not have a quota should have a TAC value of 10^{12} in the RegTAC_XXX parameter. For example, for a species that is managed through 4 regions the parameter may look like:

RegTAC_XXX
500 1000 5000 10E+12

When regional management is in place, the TAC limit is assessed for each region separately. If the TAC in a region is exceeded, then any catch of species XXX taken in that region is discarded (remember, the discards can survive, and this setting will affect the final fishing mortality). However, the fishery itself is not closed at this point. The fishery is only closed when the overall TAC is exceeded and a **fishery is considered as over TAC only when it exceeds TAC in ALL regions** for the number of species given in the YYY_max_num_sp parameter. If fishing effort is distributed proportionally to biomass and relative box biomass of the species does not change through time, then regional TACs will lead to the same catch as the global TAC. However, this is not usually the case, which means that applying regional TACs can lead to higher fishing mortality than applying a global TAC depending on species and fleet movements. For instance, if the fishery is required to land all the catch rather than discard anything (YYY_landallTAC_sp=1), the landings for some regions will usually exceed that regional TACs.

The regional TACs are calculated in *Apply_Annual_Fisheries_Mgmt()* routine in **atImplementation Annual.**c

NOTE!

What is a stock and how to define it?

Each functional group can have different stocks. These stocks are not spatially overlapping and therefore must occur in different boxes or at least in different vertical layers (e.g. deep water and shallow water stock). The number of stocks for each species is given in NumStocks in the *functional_groups.csv* and the spatial distribution of the stocks in the *biology.prm*, where XXX_stock_struct indicates which stock lives in each of the model boxes and XXX_vert_stock_struct shows which stock lives in each vertical layer (with the same vertical stock structure assumed to apply in all boxes).

Stocks can have different biological parameters, such as predation vulnerability scalar (pSTOCK_XXX) and stock-specific recruitment scalar (recSTOCK_XXX). Stocks can also have stock specific fisheries catchability qSTOCK_XXX.

Setting up different stocks can have its pros and cons. Using stocks can better represent real-world population structure with a minimum of additional parameters. However, if different "stocks" have different biological characteristics, it might be better to define them as different functional groups. Although, setting stocks as different species will increase computation time.

Remember that even if one defines separate stocks, they can still be managed as one large stock, if they are targeted by one fishery and the management of that fishery is set to YYY_flagmanage=2

NOTE!

What is a management region and how to define it?

The management regions are defined in the *biology.prm* in regID file where one box can be assigned only to one region. The management region can represent a separate TAC based region or other management unit. The biomass per group per region is output in the *BiomReg.txt* file.

16.2.6. Shared total allowable catch allocation across fisheries

The TAC_XXX parameter is given for each species and fishery combination, and usually each fishery is only affected by its own TAC. However, it is possible to set up a shared TAC for several (or all) fisheries using flagTACparticipate_YYY=1 parameter. In that case only one common TAC pool is allowed, so a fishery can either participate in this pool or have its own independent TAC (flagTACparticipate _YYY=0). The shared TAC is independent of the YYY_flagmanage options discussed above.

Table 12 gives a hypothetical example of three fisheries participating in a common TAC pool. The TACs for each targeted species by each fishery are given in the first three rows. The total TAC for a given species across all participating fisheries is then just a sum of their individual TACs. It is likely that different fisheries have different efficiencies at catching a species, so a fishery with a small TAC can benefit by "stealing" some of the TAC from other fisheries. This is because the fishery will not be closed when it exceeds its own TAC, but only when the total TAC has been exceeded. The last row shows the total catch by the three fisheries; for species A, C and F the catch has reached the total TAC. The maximum number of species for which each fishery can exceed its TAC is set to 2 (max_num_sp_ YYY=2) for all fisheries. The first two fisheries have reached the TAC for two species and will be closed, while the third fishery can still operate and catch both species C and D, even though the TAC for species C has been reached (because it will be closed only when it reaches its TAC for 2 species).

Table 12. An example of how shared TACs among three fisheries might affect catch and fishery closures. Grey columns show species for which TAC has been reached. The three fisheries target different species but all participate in a shared common pool TAC.

	SpecA	SpecB	SpecC	SpecD	SpecE	SpecF	SpecG	Species >TAC	Closed?
TAC of fishery I	100	200	-	-	200	50	-	2	Yes
TAC of fishery II	200	-	-	100	50	100	100	2	Yes
TAC of fishery III	-	-	1000	1000	-	-	-	1	No
Total TAC (sum)	**300**	**200**	**1000**	**1100**	**250**	**150**	**100**		
Catch	*320*	*180*	*1000*	*1000*	*200*	*160*	*20*		

16.2.7. Multi-year TACs

Typically TACs set a yearly catch limit and are re-assessed annually, but it is possible to also set multi-year TACs as well. The number of years for which the TACs are allocated are given in the tac_resetperiodYYY parameter. This value is most often set to 1, but could be any number of years.

The multi-year TAC can be applied in two cases:

1) Fisheries are allocated a total catch over a several year period, set with a global parameter bulkTAC =1. For example, a 3-year bulk TAC can be set to 600 tons, which makes an average yearly TAC of 200. The TAC_XXX parameter should still give yearly values, as Atlantis resets them based on the tac_resetperiodYYY parameter for that species and fishery. However, if a fishery catches 300 or even 550 tons in the first year it will still not be closed, until the 3-year TAC quota has been reached. If the fishery catches all its multi-year TAC in the first year, it will remain closed for the rest of the tac_resetperiod_YYY.
2) If some form of assessment model is used (see Note! in chapter 16.1) TACs are dynamic. In this case the multi year TAC option with bulkTAC=0 means that TACs are annual, as usual, but new assessments and new TACs are only done every few years, with the frequency given in the tac_resetperiodYYY parameter.

16.2.8. Trading closure after exceeding TAC with MPAs

When some form of assessment is used (see Note! in chapter 16.1) and TACs are dynamic, then a fishery might be given an option of imposed MPAs instead of reduction in TACs. This is based on the real-life situation where fishers prefer an imposition of MPAs instead of a reduction in TACs. To simulate this option in Atlantis, one must have dynamic TACs (and some form of stock assessment included) and set parameter flagTradeTACvsMPA=1. Further, to set up this option properly it is important to check that the correct options for YYY_flagmpa have been selected (see chapter 16.4).

16.2.9. Temporal changes in TAC

As with other harvest and management parameters, the user can allow for temporal changes in TACs. This can be done in a standard forced way (YYY_TACchange=2) or dynamically through the

Assessment or pseudoassessment submodel (YYY_TACchange=1). The dynamic TAC change through the Assessment submodel aims to simulate real world assessment and decision making processes, where TACs are updated based on the stock assessments. The Assessment submodel is currently not covered in this version of the manual. Users are currently advised to consult the wiki, the Atlantis code and Atlantis developers for further details – as assessment type, parameters, frequency of data collection and meta-rules around TAC setting (e.g. degree of political lobbying, buffers imposed at various decision steps) must also be set.

When changes in TAC are forced, the changes are implemented in the same way as with other temporal changes in a parameter. XXX_TAC_changes indicates the number of TAC changes for each fishery and species XXX. The start day of each change is given in TACchange_start_XXX, the period over which the change occurs is given in TACchange_period_XXX and the multiplier for the TAC change to be achieved by the end of the change period is given in TACchange_mult_XXX. **At the end of the change period, the new TAC scalar applies for the rest of the simulation (or until another change takes place).**

16.2.10. Frame based TAC

Some species are managed using biomass thresholds, if biomass is above a specific level then fishing is allowed. To represent that in Atlantis for fishery YYY:

1. Set the species to be a TAC managed in the *functional_groups.csv* file
2. Set YYY_flag_framebased to 1 for the fishery.
3. Set up the period for which this form of management should apply for the fishery - using the YYY_start_manage and YYY_end_manage parameters (in days of the simulation run).
4. Set the allowed level of fishing for when the fishery is active. To do this set the TAC_XXX parameter for each species in that fishery.
5. To set the biomass level at which fishing is allowed for the fishery set the value for FC_high_thresh_XXX for each species in the fishery. This scalar will be applied to the initial biomass to give the threshold biomass. Above this threshold fishing will be allowed.

Table 13. Parameters used in stock adaptive and TAC based management.

Parameter	Description
YYY_flagmanage	Fisheries management options (ranging from 0 to 8).
flagreinitpop	If set to 1, the virgin biomass for management and stock assessment will not be calculated from the initial conditions, but after a burn-in period given in reinit_pop_day parameter (in days of simulation run)
FC_thresh_XXX	Proportion of the population of species XXX when effort of ALL fisheries fishing of XXX is restricted. This is used in rule based stock management, set by YYY_flagmanage = 1
YYY_FC_restrict	Proportion of effort allowed once restrictions begin
YYY_FC_period	Period of time over which effort reduction occurs
YYY_FC_period2	Period of time before stock status is re-checked. If the stock has recovered at this time restrictions are removed. If the stock remains depleted the time counter is reset (i.e. the restrictions are triggered afresh).
FC_high_thresh_XXX	This parameter can be used in two ways.

	If frame-based fishing is **not** being used, this is the proportion of the population species XXX has to reach when existing fishing restrictions are in place due to low stock biomass (set by FC_thresh_XXX) are lifted and original TAC levels are reinstituted. If frame-based fishing is **allowed** then FC_high_thresh_XXX * initial_biomass gives the threshold biomass above which fishing is allowed to happen (set using TAC_XXX).
TAC_XXX	Total allowable catch (in tons per year) for species XXX by each fishery (must have as many values as there are fisheries). If set to 10^{12} then no TAC is applied for that species/fishery combination.
flagTACincludeDiscard	If set to 1, TAC calculations include all discarded biomass.
YYY_landallTAC_sp	If set to 1, fisheries YYY is not allowed to discard, but must land of catches.
pseudo_assess useRBCTiers	Flags determining which fisheries stock assessments are applied to calculated dynamic TACs (see Note! in chapter 16.1)
YYY_max_num_sp	A maximum number of species a multi-species fishery YYY can exceed it's TAC for, before it is closed (when simple TACs are applied, see chapter 16)
co_sp_XXX vector co_sp_catch_XXX	Vectors giving the ID numbers and ratios of companion species (species that are caught together) caught with each biomass unit of species XXX. Used when YYY_flagmanage=5. See chapter 16.2.4 for further details and related parameters.
basketSPXXX	A vector of 0 or 1 values indicating which species are included in the basket TACs together with species XXX. Used when YYY_flagmanage=3 (or other value, see Table 11). See chapter 16.2.4 for further details and related parameters.
manage_reg	If set to 1, TACs management will be based on regional rather than global biomasses, with regional TACs given in RegTAC_XXX vector. See chapter 16.2.5 for further details and related parameters.
flagTACparticipate_YYY	If set to 1, a fishery YYY shares its TAC with other fisheries (see chapter 16.2.6. for further details)
bulkTAC	If set to 1, some fisheries can be allocated multi-year TACs rather than TACs set for each (see chapter 16.2.7)
YYY_TACchange	If >0 TACs can change during the simulation run. If YYY_TACchange=2 then forced changes are applied as described in chapter 15.10 and 16.2.9. If YYY_TACchange=1, then dynamic changes in TACs are used, but this option must have some form of the stock assessment applied (see Note! in chapter 16.1).

16.3. Broken stick harvest control rules and broken stick scalar

The "broken stick" scalar management is similar to the stock adaptive management option described in chapter 16.2.2. However, while the stock adaptive management is applied for the dynamic fishing option, the broken stick management is only available for the user defined fishing mortality option described in chapter 15.3. The setup of broken stick management is not affected by the YYY_flagmanage options used for dynamic fishing options described in chapter 16.2.

Broken stick management is based on a broadly applied fisheries concepts of maximum sustainable yield (MSY), maximum economic yield (MEY) and limit reference points. Usually, in fisheries management the stock biomass and consequently the fishing mortality that would result in the target biomass is calculated in different versions of stock assessments. The biomass and fishing mortality values that lead to MEY, MSY and critically low stock status are referred to as reference point A, reference point B and limit reference points. A very broad rule of thumb is that stock biomass required for MEY is 48% of the virgin biomass, for MSY is about 40% and limit reference point is close to 20% of the virgin biomass. The general idea is that once a stock falls below the target reference point(s), some management action to reduce fishing mortality is required. However, there is a lot of variation and controversies about these reference points, so here they are only given for general illustration! Not all countries use three reference points, and in fact the most conservative reference point A is often ignored and does not lead to any action. Currently, in Atlantis the MEY (reference point A) is considered the primary "target reference point", but effort reductions do not come in to place until stock sizes slip below MSY (reference point B) as that was the form of management rule in place in Australia at the time the code was developed.

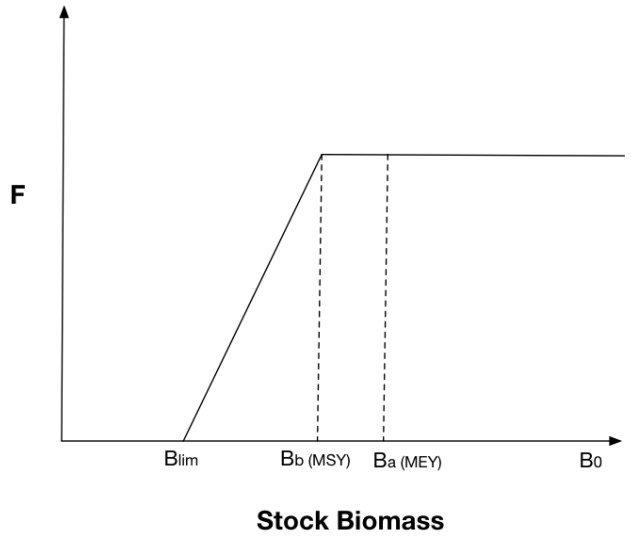


Figure 11. The shape of the broken stick harvest control rule with reference points marked on.

To set up the basic broken stick management in Atlantis, the user should provide parameters for the three target reference points (targ_refA, targ_refB and lim_ref). The user also should provide estimated fishing mortality that would lead to the biomass at the reference point A/B, and estimated virgin biomasses in estBo parameter. Note, that these estimated virgin biomasses are different from the way virgin biomasses are used in some harvest routines (through flagreinitpop=1 and reinit_pop_day parameters). The estBo virgin biomasses to be used in this option are provided in the *harvest.prm* file. In this way the users can explore what happens if estimated virgin biomasses used in assessments and decision making are very different from the actual virgin biomasses in the simulations. The parameters Frestart_scalar also needs to be set as this is the multiplier for mFC_XXX to use as the default starting F once the fishery is reopened.

How to set up broken stick management?

1) Set the species as isFished=1 in the *functional_groups.csv* file

2) Set fishing through the user defined fishing mortality as described in chapter 15.3
3) Set parameter tierXXX to a value larger than 0. The tierXXX parameter tells Atlantis which assessment option to use. However, in the case of broken stick management, no specific assessment is necessary (as a pseudo assessment can be used). Setting up tierXXX>0 just tells Atlantis that broken stick management option should be applied for the species if its fishing is done through a user defined fishing mortality.
4) Set the limit, A and B reference points in targ_refA, targ_refB and lim_ref parameters. Note, that these are NOT species specific, which means that same reference points will be applied for all species managed under the broken stick management option.
5) Give a list of virgin biomasses for the species in the estBo parameter. These biomasses can be the same as the initial biomasses in the simulation, or they can be completely different (if consequences in virgin biomass misspecification are to be explored)
6) It is possible to have two sets of reference points. This was implemented to simulate the situation where fishing of forage species required different reference points. To set up the second set of reference points, give their values in forage_refA, forage_refB and forage_lim_ref parameters. Also in parameter whichref identify which of the two reference points should be applied to each species (0 – forage reference points, 1 – target reference points)
7) Provide Atlantis with the annual fishing mortality rates Fref (F year^{-1}) that would result in the stock biomass equilibrating to either targ_refA or targ_refB (this choice is up to the user, based on the specifics of the harvest strategy in their location, remember that currently there is no reduction in fishing pressure until biomasses drop below the level specified by refB). These values are typically estimated in stock assessments, and are independent of the fishing mortality values set in the mFC_XXX parameter (see chapter 15.3). If using tier8 then it is also necessary to supply the fishing mortality value to use at high biomass levels (when biomass is greater than targ_refA).
8) The parameters Frestart_scalar also needs to be set as this is the multiplier for mFC_XXX to use as the default starting F once the fishery is reopened.
9) Set up the period for which the broken stick management of fishery YYY should apply in YYY_start_manage and YYY_end_manage parameters (in days of the simulation run).
10) You can also provide reference points for byproduct and bycatch groups using byproduct_refA, byproduct_refB, byproduct_refC, byproduct_refD, byproduct_lim_ref, bycatch_refA, bycatch_refB, bycatch_refC, bycatch_refD, bycatch_lim_ref – all of which have the same definition as for target and forage groups, but applied to byproduct and bycatch groups. You can also provide discardTAC, which is used once B < Blim for TAC managed groups (this is currently only used for tiered management options).

Note: If do_sumB_HCR is set to 0 then the Broken Stick is applied per species. If do_sumB_HCR is set to 1 then the harvest control rule will be applied at the guild level. The guild is defined by co_sp_XXX.

The broken stick management routine is run once per year and implemented in the routine *Check_F_Harvest_Control_Rules()* in **atManageAnnual.c**. It assesses the current stock biomass against the reference points. The routine **assumes perfect knowledge about the stock** and uses actual biomasses in the model domain as the current assessed biomass (use of stock assessment processes and assessment biomasses can be implemented upon request).

In both cases, to drive F to the desired levels the new $F_{current}$ (calculated as defined below) is found by rescaling the base F (mFC_XXX) appropriately, as noted below (as it is slightly different in each option).

Australian option – usually triggered using tierXXX set to 1 (tier1)

While biomasses are above reference point B the target fishing mortality F_{target} is used so that the currently implemented fishing mortality rate ($F_{current}$) is given by:

$$F_{current} = F_{ref} \qquad ,if\ B_{cur} > B_{refB}$$

where B_{cur} is the current stock biomass (assuming perfect knowledge); B_{refB} is the stock biomass at reference point B (estimated as targ_refB * virgin biomass); and F_{ref} is the fishing mortality that should lead to the biomass at the reference point B (Fref parameter).

If the stock biomass falls below the reference point B, management kicks-in and the fishing mortality provided in the mFC_XXX parameter is overwritten so that the current fishing mortality is given by:

$$F_{current} = F_{ref} \cdot \frac{B_{cur} - B_{lim}}{B_{refB} - B_{lim}} \qquad ,if\ B_{lim} \leq B_{cur} < B_{refB}$$

$$F_{current} = 0 \qquad ,if\ B_{cur} < B_{lim}$$

Here B_{lim} is the stock biomass at the limit reference point (estimated as targ_lim * virgin biomass). These equations show that if, for example, current stock biomass is half way between the target and limit reference points, the fishing mortality will be set to half of the reference fishing mortality *Fref* given in the parameter file (note that this reference fishing mortality is NOT the same as the user defined fishing mortality given in mFC_XXX parameter). If the stock biomass falls below the limit reference point, then fishing mortality is set to 0.

In the harvest code this is implemented as

$$F_{realised} = mFC \cdot \frac{F_{current}}{F_{max}}$$

US and Norwegian option – usually triggered using tierXXX set to 8 (tier8) or 9 (tier 9)

While biomasses are above reference point A the target fishing mortality F_{target_high} is used so that the currently implemented fishing mortality rate ($F_{current}$) is given by:

$$F_{current} = F_{ref\_high} \qquad ,if\ B_{cur} > B_{refA}$$

where B_{cur} is the current stock biomass (assuming perfect knowledge); B_{refA} is the stock biomass at reference point A (estimated as targ_refA * virgin biomass); and F_{ref_high} is the fishing mortality that is applied at high biomass levels (Fref_high parameter).

When biomasses are below reference point A but above reference point B the target fishing mortality F_{target} is used so that $F_{current}$ is given by:

$$F_{current} = F_{ref} \qquad , if\ B_{ref\boldsymbol{B}} \leq B_{cur} < B_{ref\boldsymbol{A}}$$

where B_{refB} is the stock biomass at reference point B (estimated as targ_refB * virgin biomass); and F_{ref} is the fishing mortality that should lead to the biomass at the reference point B (Fref parameter).

If the stock biomass falls below the reference point B, management kicks-in and the fishing mortality provided in the mFC_XXX parameter is overwritten so that the current fishing mortality is given by:

$$F_{current} = F_{ref\_lim} \cdot \left(B_{ref\boldsymbol{B}} - B_{cur}\right) + F_{ref} \cdot \frac{B_{cur} - B_{lim}}{B_{ref\boldsymbol{B}} - B_{lim}} \qquad , if\ B_{lim} \leq B_{cur} < B_{ref\boldsymbol{B}}$$

$$F_{current} = F_{ref\_lim} \qquad , if\ B_{cur} < B_{lim}$$

Here B_{lim} is the stock biomass at the limit reference point (estimated as targ_lim * virgin biomass). These equations show that if, for example, current stock biomass is half way between the target and limit reference points, the fishing mortality will be set to half of the reference fishing mortality *Fref* given in the parameter file (note that this reference fishing mortality is NOT the same as the user defined fishing mortality given in mFC_XXX parameter!). If the stock biomass falls below the limit reference point, then fishing mortality is set to 0. Note in the *harvest.prm* the parameters *Fref and FrefLim* are entered as a vector with one value per group.

If tierXXX is set to 8 then in the harvest code this is implemented as

$$F_{realised} = mFC \cdot \frac{F_{current}}{F_{max}}$$

whereas if tierXXX is set to 9 then it is

$$F_{realised} = mFC \cdot \frac{F_{t-1}}{F_{max}} \cdot \frac{F_{current}}{F_{ref}}$$

Escapement option –triggered using tierXXX set to 13 (tier13)

This escapement based method comes from Iceland. It is calculated as:

$$F_{current} = 1 - \frac{B_{lim}}{B_{ref\boldsymbol{B}}} \qquad , if\ B_{lim} < B_{cur}$$

$$F_{current} = 0 \qquad , if\ B_{cur} \leq B_{lim}$$

In the harvest code this is also implemented as

$$F_{realised} = mFC \cdot \frac{F_{current}}{F_{max}}$$

Adding error

The broken stick case can be applied with perfect knowledge or with error (noise) added. The parameters for setting this up are added to *harvest.prm* file. There are three error options – set using estError (a vector with one entry per functional group): uniform (0), normal (1), lognormal (2). The specification of the error is set using estCV (the variance of the error as a proportion of the true value)

and estBias (also as a proportion of the true value). Both of these parameters are entered as a vector, with one value per functional group.

If the error is uniform then the biomass used in the broken stick calculations is given by:

$$B_{est} = B_{true} + \mathcal{B}_{error}$$

where $\mathcal{B}_{error}$ is a uniform random number drawn from $[-B_{estCV}, B_{estCV}]$. Whereas if the biomass is of the normal type then the biomass used in the broken stick calculations is given by:

$$B_{est} = \sqrt{-2 * log(\beta_1)} \cdot cos(2\pi \cdot \beta_2) \cdot \sqrt{estCV} + estBias \cdot B_{true}$$

where both the β are random numbers drawn form [0,1]. Lastly, if the lognormal form is chosen then the biomass is of the normal type then the biomass used in the broken stick calculations is given by:

$$B_{est} = B_{true} \cdot estBias \cdot e^{\sqrt{-2 \cdot log(\beta_1)} \cdot cos(2\pi \cdot \beta_2) \cdot \sqrt{estCV} - \frac{estCV}{2}}$$

16.4. Spatial management through MPAs

Marine protected areas (MPAs) allow permanent or temporary closures of some model spatial domains to fishing. They are set as a proportion of the box that is covered by the MPA, which is treated as a scalar on box-specific effort, fishing mortality or imposed catch. The MPA scalar is fishery specific and is given in the MPAYYY parameter, listing scalars for each box of the model domain – this scalar represents the proportion of the fishable habitat within the box that remains **open** to fishing.

Note, that even though MPA scalar is usually <1 (which means that MPAs reduces the fishing pressure in the box), it is possible to set it to values >1. This would mean that the presence of MPA in a box (or perhaps an adjacent box) actually attracts fishing to the areas just outside the MPA and the end result is that the presence of the MPA increases local fishing pressure. This is a very simplified representation of real-world fisher behaviour that can see concentration of effort around MPAs (attracted by "spill-over" from within the MPA). This simplest representation is used because the exact location of fish within a box is not tracked, rather functional groups are assumed to be uniformly distributed within all suitable habitat within a box and the MPA parameter simply acts as a scalar on the box biomass available to fishing. In reality, even if MPAs attract increased fishing effort to its perimeter, the actual fishing mortality will depend on the movement of a species. If movements are slow and a species stays mostly within the MPA, increased effort around the MPA perimeter will not result in the total increases in fishing mortality. As Atlantis does not explicitly represent these fine scale interactions they must be kept in mind by the modeller when setting the MPA parameter values.

To activate MPAs in your simulations, set the global flagmpa = 1 and then select the MPA option for each fishery in the YYY_flagmpa parameter. Currently there are 15 different ways to set up MPAs, which include different combinations of the following options for setting MPAs:

1) MPA scalar set in MPAYYY parameter, indicating the proportion of the box open to fishing or more specifically a scalar on the available species biomass. These MPAs can be set for the entire model run or the MPA implementation can start and end at different times during the

simulation run. If the latter option is wanted, set flagSimpleStartStopMPA=1, and give the year number (NOT day number!) for when MPAs start and end (All_MPAstartyr and All_MPAendyr). Note, this option only allows for the start and end years to be set for **ALL MPAs at the same time**.

2) For more complex MPAs scalars are provided through an external TS forcing file. This option is useful when MPAs change through time. To use this option the user needs to provide time series files for desired model boxes, indicating an MPA scalar value through time. To indicate that external MPA forcing is provided, set nMPAts to value >0 in the *force.prm* file (see Chapter 8 on the general format of forcing files) and the correct MPA option is set in the YYY_flagmpa parameter (see Table 14 for the available options).

3) MPAs are set up when the biomass of endangered species drops below a set level. These MPAs act in the same way as other MPAs, but their box-specific scalars are provided in a separate MPAendangeredYYY parameter. This means that users can specifically target boxes that are important for endangered species. Also, unlike fixed MPAs, the endangered species triggered MPAs will only be in place when biomasses of endangered species are below a set level. To identify which species will be used to trigger such species set the SP_Concern parameter to 1 for that species (with the order of the species assumed to match that in the *functional_groups.csv* file). See chapter 16.5 on how the endangered species are assessed.

4) The MPA scalars supplied through either MPAYYY or external TS forcing files, but applied only when stocks fall below a set level. This is done in case where YYY_flagmanage=2 and stock adaptive management is used and an appropriate selection for YYY_flagmpa is done (see Table 14 below).

5) Closure of fisheries during the spawning season – applied in any spawning area – and set using YYY_flagmpa = 13.

6) The partial or complete closure of spawning areas traded in exchange for reduced TAC. This is only applied in cases where dynamic TACs are used and set through some sort of stock assessment process. This is used to imitate a situation (indeed applied in some fisheries management cases), where instead of reductions in TAC when stock levels are low, fishing is closed during the spawning season. For further details on setting this option see chapter 16.2.8.

7) MPAs defined in the MPAoverfishedYYY parameter are triggered when trip limits are exceeded for target species and YYY_flagmpa is set to 11.

8) In any one year, once the total catch taken exceeds the value set in TripLimit_XXX that box is closed – this option is set using YYY_flagmpa = 14.

These seven options for setting up MPAs can be used separately or in combination in one of the 13 currently available options of the YYY_flagmpa parameter. Because the YYY_flagmpa parameter is fishery specific, different fisheries can be affected by different spatial management (MPA) routines.

Table 14. Options for the YYY_flagmpa parameter showing which of the five available MPA varieties are included.

YYY_flagmpa parameter option (and name of the option used in the Atlantis code)	Fixed MPA applied all the time	Time-series box-specific MPA read from forcing files	Special MPAs applied only if endanger- ed species fall below a threshold	Fixed or time-series MPA applied only if stocks fall below a threshold	Spawning closures; including trading TAC reductions for spawning closures	Closures once catch exceeds values set in the TripLimit parameter
0 no mpas						
1 fix_mpa	X					
2 cycle_mpa		X				
3 stock_mpa	X			X		
4 pet_mpa			X			
5 stock_pet_mpa	X		X	X		
6 cycle_stock_mpa		X		X		
7 cycle_pet_mpa		X	X			
8 cycle_stock_pet_mpa		X	X	X		
9 mix_fix_rolling_mpa	X	X				
10 mix_f_r-spawn_mpa	X	X			X	
11 depth_stock_mpa				X*		
12 council_stock_mpa				X*		
13 spawn_closure					X	
14 catch_mpa						X

* These options are currently only available if using historical west coast US management processes (i.e. generally not used, but can be made available if required).

The MPAs are setup in the routine *Setup_MPA_Lists()* in **atManageMPATS.c** which creates a calendar of any fixed or time specific closures that is refered to throughout the course of the run. The list of active mpas is updated at each time step in the *Check_for_active_MPA()* routine in **atManage.c**. The *Check_for_active_MPA()* routine is called at each time step from *Manage_Calculate_Total_Effort()* routine, which scales down the effort according to the different management scalars.

It is possible to **allow for infringement of MPAs**. See chapter 15.3.3. and the NOTE! in it on how to set the infringement up.

16.5. Management through seasonal closures

Another management leaver can be applied through seasonal closures. This management is independent of the stock adaptive or TAC management set with YYY_flagmanage parameter or spatial management through MPAs. Seasonal closures refer to a complete halt of fishing activity for a set period of time every calendar year. Such seasonal closures are used to protect stocks during spawning seasons or to limit fishing effort for other reasons.

In Atlantis it is best to think of seasonal closures as the length of a fishing season. This means that a fishery's season opens and closes at a set day of the year. Also the length of the fishing season can change through time.

To activate seasonal fishing for a given fishery set YYY_flagseasonal =1. Then indicate the day of the year when the fishing season opens and closes (YYY_seasonopen and YYY_seasonclose). Remember that Atlantis counts from 0, so the first day of the calendar year is 0 and the last day is 364.

If a fishery operates through several management regions (see chapter 16.2.5 on how to set up management regions), users should also indicate whether seasonal closures apply to each management region. This is set with parameter RegSeason which must have as many entries as there are management regions, and where entries of 1 indicate that seasonal closures apply.

If the length of the fishing season changes during the simulation set YYY_changeseason = 1. However, unlike the other temporal changes in the harvest parameter file, the user does not provide specific scalars to set the new length of the fishing season, but the scalars are taken from the effort scalar used for the stock adaptive management (see chapter 16.2.2) or the management of endangered species (chapter 16.6). If both scalars are available, then the smaller scalar is used. This basically means that if YYY_changeseason=1 then season will be REDUCED in length when stocks of fished species or endangered species fall below threshold levels. This is because stock adaptive management scalar or endangered species is scalar is set to the value <1 when the trigger is tripped (biomasses go below threshold levels). This option will only be applied when either stock adaptive (YYY_flagmanage =1) or endangered species (flagendangered=1) management is used. This handling of the changing season link is because this is how fisheries using season length to control effort operate. If more direct control is required over fishing season please contact the Atlantis developers.

16.6. Management of endangered species

Management of endangered species means that fishing effort will be reduced when stocks of endangered species fall below a set threshold level. The procedure is very similar to stock adaptive management described in chapter 16.2.1. This management option is independent of other management options, such as MPAs, seasonal closures, stock adaptive or TAC based management.

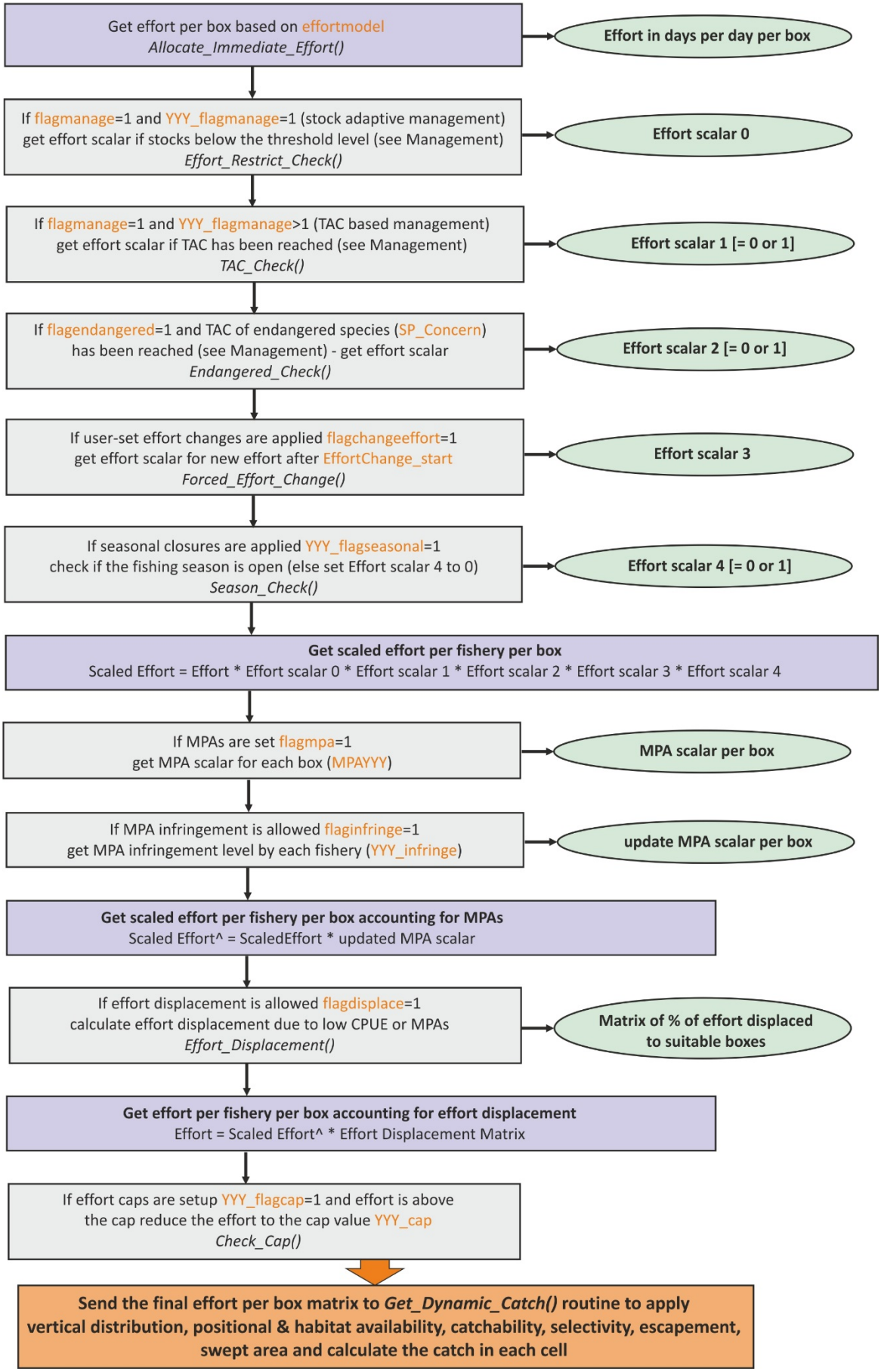
To apply the endangered species management option set the global flagendangered =1. The list of endangered species is given in the SP_Concern parameter, which has as many entries as there are groups in the *functional_groups.csv* file and where the value of 1 indicates that a group is endangered.

The threshold level of endangered species (compared to virgin biomasses) below which the trigger is started is given in the same FC_thresh_XXX parameter that is used for stock adaptive management. This parameter gives threshold values of species XXX abundance for each fishery – if a species XXX is harvested then the parameter is used for stock adaptive management, if it is endangered it is used for endangered species management.

When the abundance of endangered species XXX falls below the threshold given in the FC_thresh_XXX, the fishery effort will be reduced to the proportion of the original effort given in the YYY_FC_restrict_endangered parameter (the value is set between 0 and 1). The period over which reductions take place is given in the YYY_FC_endanger_period, which means that the effort is not reduced in one time step but gradually.

Steps in calculating effort per box in dynamic catch

Manage_Calculate_Total_Effort()



16.7. Closing fisheries due to contaminants

If contaminants are functional in the model then fisheries closures can be triggered. To activate such closures set flag_contam_fisheries_mgmt to 1 for closures to be triggered on a set day (contam_fishery_closure_day) and last a set period (contam_fishery_closure_period in days) – the location of the closure is defined in the vector ContamClosed. This vector is nbox long. If the box is closed due to the contaminant set the value for the cell to 1.0, leave it as 0.0 if the box remains open.

If flag_contam_fisheries_mgmt is set to 2 then the closure is based on contaminants in the cells of groups in the model – any cohort of any group with contaminant content greater than ZZZZ_fishery_thresh_level (where ZZZZ is the contaminant name) will trigger a closure. If contam_fishery_closure_option is set to 0 only the immediate box is closed, if set to 1 then adjacent boxes are also closed.

17. ECONOMICS

17.1. Introduction into the Economics submodel and its main principles

The Economics submodel aims to capture socio-economic pressures that determine harvesting decisions. There is an increasing realisation that human behaviour and socio-economic drivers are important determinants of marine resource use, but many models do not attempt to capture this. To fill this gap, Atlantis includes Economics submodels. **However, users should be aware that the Economics submodel has not been used and tested as widely used as other submodels (e.g. biology) and should be used with caution.** When applying this submodel it is recommended to start with simple options, where model outputs against the expectations. In this manual only the key routines are described; users interested in further details should consult the key publications listed below, the Atlantis wiki, the Atlantis google groups or the Atlantis model developers.

Some key references to consult on application of Economics submodels in Atlantis

1. Kaplan IC, Holland DS, Fulton EA (2014) Finding the accelerator and brake in an individual quota fishery: linking ecology, economics, and fleet dynamics of US West Coast trawl fisheries. *ICES Journal of Marine Science* (Dan Holland economics model version, fish prices and quota trading)
2. van Putten EI, Gorton B, Fulton B, Thebaud O (2013) The role of behavioural flexibility in a whole of ecosystem model. *ICES journal of Marine Science*, 70(1), 150-163. doi:10.1093/icesjms/fss175 (fisher behaviour, flexibility and historical information use)
3. Hutton T, Thebaud O, Fulton B, Pascoe S, Innes J, Kulmala S, Tudman M (2010) *Use of economic incentives to manage fisheries bycatch*: An application to key sectors in Australia's Southern and Eastern Scalefish and Shark Fisheries. Australian Fisheries Management Authority report (economic incentives to minimise bycatch and discards, deemed values, taxes on bycatch)
4. Fulton EA, Smith ADM, Smith DC (2007) *Alternative management strategies for Southeast Australian Commonwealth fisheries*. Australian Fisheries Management Authority and Fisheries Research and Development Corporation report

Original descriptions of quota trading models, now implemented in Atlantis

5. Little et al. (2009) An agent-based model for simulating trading of multi-species fisheries quota. *Ecological Modelling* 220, 3404–3412 – quota trading study implemented in Atlantis.
6. Newell RG, Sanchirico JN, Kerr S (2005) Fishing quota markets. *Journal of Environ. Econ. Manag*. 49: 437–462 – an alternative quota trading version implemented in Atlantis

General principles of the Economics submodel

1) Fisheries are divided into subfleets to allow for differences in gear, costs, hold size, crew characteristics, home port distributions, targeting, quota holding, personal attitude profiles and travelling distances. This allows more flexible fishing effort dynamics.

2) Fishing effort is modelled through market dynamics, which means that fish prices and fishing costs vary through time, and fishers mostly distribute their effort to maximise profit.

3) Under a TAC system, quota can be traded dynamically among fishers. Unlike in non-economics based fisheries effort models, TACs may not be fulfilled (i.e. no, or not all quota will be used) if fishing is unprofitable such as when fish prices are low or fishing costs are high.

4) Some level of non profit maximising human behaviour is allowed. Many fisheries decisions in the real world are based on criteria other than profit maximisation (lifestyle, beliefs, habit). In Atlantis, this is implemented through an extra parameter defining the flexibility of fishers to learn and so change effort to match the areas of highest CPUE (see chapter 17.5), and by introducing stochasticity in gear switching, vessel update routines and quota trading routines (see chapters 17.3 and 17.4.1).

5) There is the potential to implement a quota trade network where for instance, there is preferential trading between fleets or subfleets. This can emulate empirically based or hypothetical preferential trading based on kinship, friendship, or business relationships. This is not activated in all versions of the code and anyone interested in using it should approach the Atlantis developers for support in its use.

6) The module outputs some simple economic indicators. However, these should be used with caution as the Atlantis developers are not economists and have not always used standard economics conventions with regard to terminology and the use of discount factors.

What is a subfleet and why is it needed?

Subfleets are used to model fisher behaviour at a higher resolution, along the lines of metiers as defined by Pelletier et al (2000) or Deporte et al (2012). The idea is to divide up fishing fleets into subfleets (metiers) with similar properties. The key aspects of subfleets is that they can have different costs of fishing, behavioural flexibility and quotas. The subfleet behaviour is used to better reflect dynamic effort changes of different fisheries. The profits of fishing operations are also

tracked at the subfleet level and dynamic changes in the numbers or activities of subfleets shapes the realised fishing practices at the fleet level.

Subfleets are only defined in the Economics submodel – in the Harvest submodel the actual catches are done at the level of fisheries, as described in chapter 15.4. This is to maximise compatibility with the other ways of representing fisheries. However, it also means that regardless of the vessel sizes or fishing costs, all subfleets within one fishery are characterised by the same gear selectivity, fish escapement, incidental mortality and other fishery specific parameters given in the dynamic fishing equations in chapter 15.4. The only thing that is affected by subfleets in the Harvest submodel is the effort per fishery per box per timestep.

The subdivision of fisheries into subfleets is used to have a more dynamic representation of effort allocation per box per time step. When the Economics submodel is used, effort allocation reflects fisher behaviour and subfleets can respond to markets and trade quota.

The Economics option is only available when dynamic fishing is used (so imposed catches or user defined fishing mortality do not have the option to turn on Economics).

Parameters in the economics.prm *are mostly subfleet specific and include:*
Number of boats, hold size, crew size, behaviour flexibility, costs of fishing, quota allocation

All fisheries related parameter defined in harvest.prm *file are identical among subfleets within a fishery, including:*
Gear selectivity, target species, fish escapement, incidental mortality, discarding, minimum and maximum fishing depth, fishing habitat characteristics and so on

Table 15. General Economics and subfleet parameters

Parameter	Description
YYY_nsubfleets	Number of subfleets per each fishery. This value is only used if economics is switched on for that fishery. If the economics sub-model is being used then this value must be at least 1 or Atlantis will crash.
YYY_nlicence	Number of licences for each fishery across all subfleets. It defines the maximum number of vessels that may be active in a fishery.
YYY_nboat	Number of boats in each subfleet. The array should have as many entries as the maximum number of subfleets. For fisheries that have fewer or only 1 subfleet, give 0 for the remaining "empty" subfleets. For fisheries that have an effortmodel value < 12 then set all values to 0 (as they are not modelled using subfleets but instead use one of the effort models defined in the harvest sub-model).
YYY_boat_size	Boat size (m) for each subfleet.
YYY_hold_capacity	Hold capacity (in tonnes). Used in the Dan Holland effort allocation option.
YYY_crewsize	Number of full time equivalent people in the crew
YYY_home_port	Port for each subfleet (with the number corresponding to the port

Parameter	Description
	numbers defined in the *harvest.prm*). This will be important when weighing the distance of boxes to ports in effort allocation
YYY_triplength	Average length of a fishing trip (in days)
YYY_FishableLength	Length of time (hours per day) spent fishing
YYY_ShotLength	Length of one fishing effort or shot (in hours) for each subfleet. This is used to calculate the final effort (in days per day) for each fishery.
YYY_MaxShotVol	The volume of one fishing effort or shot (in tonnes). This is only used in the shot-by-shot fine scale calculations and not in the general economics sub-model.
YYY_minDownTime	The proportion of the total time that each subfleet must stay in the port (for maintenance, etc.). This will decrease the maximum possible effort each subfleet can apply
month_scalar	Global parameter setting up the scalar in number of days per month that each subfleet can apply. It is used in calculating max_month_effort, see chapter 17.
BycatchCountDiscards	This flag indicates whether the discard of bycatch species are used in calculations involving total catches (e.g. for calculating total catch per box vs profit or taxes, used in deciding where to target fishing effort). The inclusion of bycatch may be important if there are penalties or taxes for taking endangered species.

17.2. Key routines of the Economics submodel

If the Economics submodel is turned on (flagecon=1 in the *run.prm* file) the *Economics()* routine is called at every time step, setting up fishing dynamics based on sub-fleet parameters.

The key steps done in the *Economics()* routine are:

1. **Allocate catch** from the Harvest submodel to subfleets of each fishery. In the Harvest and Ecology submodels all catch is applied and recorded at the level of a fishery. In the Economics submodel, however, fisher behaviour and dynamic fishing is more resolved and modelled at the level of subfleets within each fishery. Once the harvest per species per age group per fishery per box is calculated in the Harvest submodel it is then passed back into the Economics submodel, where it is divided among subfleets. This allocation to subfleets is proportionally based on the effort for each subfleet in that cell in the previous time step (stored in the "black books" of each subfleet, see below) and the availability of quota for that subfleet. The allocation of catch among subfleets is done by the *Allocate_Catch()* routine **ateconresponse.c**.

2. **Assess the economic situation or market behaviour**, i.e. calculate fish prices (used in deciding what to fish), calculate quota prices (for quota trading) and a range of economic indicators. This is done in *Get_Fish_Prices()*, *Total_Quota_Price()* and *Economic_Indicators()* routines. The economic indicators are used for reporting purposes only and do not affect the simulations. See chapter 17.3 for further description.

3. At the start of the month assess the subfleet's economic status and prospects, any **responses**

to the market leading to buying or selling of boats, and update number of boats associated with ports and different markets. This is done in *Update_Vessel_Numbers()* and *Port_Growth()* routines. Subfleets can run at a loss (i.e. costs are greater than returns) for certain periods of time – and debts (negative profits) are created. If expected profit is not sufficient to reduce the debt levels and debts become 'crippling', after a certain amount of time a subfleet will start decreasing effort. A subfleet may exit the fishery (i.e. boats are decommissioned) if the debt is large and continues for too long. See chapter 17.4 for further description.

4. **Make an annual** (start of the year) **and monthly** (start of the each month) **plan for fishing effort**. Here fishers make decisions where and how much they are planning to fish. The plan is first made for the whole year based on the historical catches and profits (stored in "black books" of each subfleet) versus expected profit for this year (the profit expectation might change depending on the costs of fishing, prices of fish and quota availability) and flexibility of fishers to change their behaviour. The main routines for annual effort planning is *Annual_Effort_Schedule()* and *Dynamic_Effort_Allocation()* in **atEconomicAnnual.c**. Then each month the annual effort plan is updated depending on whether realised catches are close to expected catches and (how the fishery is tracking in terms of the ratio of realised : expected catches versus the same time in the previous year) and on the availability of quota (remaining quota and traded quota). If realised catches are much lower than expected, then the monthly effort plan is updated and an increase in effort is planned (pending having quota in hand and/or the availability of quota on the quota market). If realised catches are much greater than planned then caps on monthly effort may come in to place (as maintenance time is required or it is physically impossible to fish more days). The monthly revision of the fishing plan is done in the *Monthy_Effort_Schedule()* routine. Note that this is still not the actual final effort, but only a monthly effort plan. See chapter 17.5 for further description.

5. **Do quota trading**. If fishers are making a positive return they will assess their quota availability, fish prices and fishing expenses. If they do not have enough quota in hand – and they therefore have a demand for quota, they make decisions to trade quota (or *vice versa* if they have quota left over, particularly if they are approaching the end of the calendar year). Quota trading is done among subfleets. Once the quota is traded then the final effort allocation can be made depending on how much quota each subfleet has for different species. This is done in *Quota_Trade()* routine in **atquota.c**. The quota trading is described together with quota prices in the chapter 17.3.

6. Make **final effort allocation** per sublfeet per box trip (done in *Allocate_Final_Effort()* routine), with the same effort assumed to hold for the length of the trip (in most cases this trip length is assumed to be weekly, but it can also be on shorter time frames). This can be done using the basic Atlantis economic model (*Multi_Plan_Effort_ Final_Allocation()* routine used in the Fulton et al. publications) or the Dan Holland version of the economic model (used in Kaplan et al. 2014) (*Holland_Effort_Final_Allocation()* routine).

   In the basic model, fishers can update their monthly effort plan depending on the last month's catch per unit effort (CPUE), value per unit effort (VPUE) or discards per unit effort (DisPUE). This means that if CPUE in some boxes is high, fishers might decide to change their planned fishing schedule and instead go to those boxes. The speed of plan updating depends on the fisher's flexibility (set for each subfleet in a separate parameter). For example, some subfleets can be considered as traditionalist and will go to places where they always go. Others might be

more flexible and quickly change their planned effort allocations depending on latest catch (see VanPutten et al. 2013 for an example study). The DisPUE is important in cases where bycatch avoidance is used. The final allocations also take into account the quota availability. See chapter 17.7 for further description.

Once the final effort allocation per subfleet is done, the effort from subfleets is pooled to the level of fisheries and passed on to the Harvest submodel to execute the actual catches given the effort, gear selectivity, and other parameters, as described in the chapter 15.4.

17.3. Calculating prices and costs

Fish prices, taxes, penalties and fishing costs determine the profitability of fishing operations and are at the core of the Economics submodel. The profits of a subfleet are determined by the revenue from the fish sold, revenues from the deemed values for bycatch (if used), revenues of leased or sold quota units minus operating costs. The cash flows are tracked at the level of subfleet; a subfleet can have surplus cash reserves or go into debt (negative profits). If a debt is large and the fleet continues to make a negative profit for too long, a subfleet will decommission boats.

17.3.1. Fish prices

Fish prices can remain static (constant) through the simulation (set through UseConstPrice = 1) or be dynamic (e.g. prices can vary according to a seasonal pattern) (UseConstPrice = 0). Static prices for each species and fishery combination are set in XXXsaleprice parameter ($/kg). Dynamic prices are updated at each time step of the simulation and calculated in *Get_Fish_Price()* routine.

Dynamic prices can vary between markets (for instance reflecting domestic and export markets), in response to yearly trends, monthly fluctuations and also incorporate price residuals supplied through external forcing files. Currently the Atlantis code allows a maximum of two markets only; the number of markets (1 or 2) are set in the K_num_markets parameter in the *run.prm* file.

The price of a species CX from a fishery Y sold in a market x at a time step *t* is calculated as

$$Prc_{CX,Y,x,t} = \left(Val_{CX,Y} \cdot a_{CX,x} + Trend_{CX,x} \cdot year + b_{month,CX,x} + c_{CX,x} \cdot residuals_{CX,x,t}\right) \cdot Wgt_x$$

Where $Val_{CX,Y}$ is the baseline sale price of a species ($/kg) given in the XXXsaleprice parameter; and $a_{CX,x}$ is the market-specific scalar for each species given in the MARKET_intercept1 and MARKET_intercept2 parameters for the two markets. The scalar is used to reflect different demand for different species in different markets. The $Trend_{CX,x}$ is the price trend coefficient for each market and species in the MARKET_trend1 and MARKET_trend2 parameter and *year* is the year number in the simulation run. The trend term is intended to reflect long-term trends in the fish market prices (both inflation driven and otherwise) and is usually a small constant ($/kg) (remember, it will be multiplied by the year number and added to the total price).

The $b_{month,CX,x}$ is the montly market constant for each species and market (in $/kg), given in the JanMarketCoefft1, FebMarketCoefft1, … parameter. These monthly values are intended to represent monthly fluctuations in consumer demand (e.g. higher prices are often observed around Christmas

time when demand is high). The relative fluctuation in the prices will be added to the total price, so they should be set according to the original sale price.

Finally, it is possible to add the additional coefficient with externally read price residuals represented by term $residuals_{CX,x,t}$. The residuals for each species and market through time must be provided through an external forcing file. If this file is not provided, then no residual value is added. The $c_{CX,x}$ is the market and species specific constant (MARKET_autocoefft1 and MARKET_autocoefft2) to be applied to the externally read residuals. The residuals are derived from fitting the empirical historical market prices to a model that includes yearly trends and monthly fluctuations. The Wgt_x represents the proportion of total catch that goes to a given market. This is determined by the proportion of total landings in each port, where each subfleet is associated with only one port (YYY_homeport) and each port is associated only with one market (given in the ports_markets parameter in the *harvest.prm* file).

17.3.2. Deemed value prices

What is a deemed value and why are they needed?

Thanks to Ingrid van Putten for clarifications

Deemed values are one of the policy tools to discourage fisheries bycatch. Deemed values are applied to catches in excess of annual catch entitlements (or quota entitlements) or the fish caught without the relevant quota holdings. In essence, the deemed value systems are intended to both provide a disincentive for fishers to catch bycatch, while at the same time allowing the authorities to better assess the bycatch amounts because the catch has to be landed and reported. The approach is a disincentive to exceed quota, highgrade or catch bycatch species, because the fisher gets invoiced for the deemed value of the fish.

The deemed value can increase with higher amounts being overfished. For instance, at 5% overfishing the deemed value rate might be $8/kg while at 10%-20% overfishing the rate might be $12/kg. This will mean that at higher rates of overfishing the reduction in the overall profit for that fisher will be higher (the total amount invoiced will be greater).

The landed (by)catches that incur these payments called "deemed values" are referred to as 'deemed' because normally these fish would/should not be landed and therefore their price is a 'deemed' price (a believed or considered price) (Squires et al. 1998). The deemed price needs to be high enough to provide a disincentive.

For further review about deemed values and their applications see Hutton et al. 2010 report.

In Atlantis, the deemed values will be used from the day of the simulation set in DVstart parameter. If no deemed values are needed, set DVstart parameter to a large value. Deemed values are applied when catch exceeds the TAC, where catch can either include or exclude discards, as set in flagTACincludeDiscard parameter in *harvest.prm* file. In addition, the Economics submodel has an additional parameter of BycatchCountDiscards, which means that the catch of these deemed species are added to the total catches (and costs) used when planning the spatial distribution of fishing effort – so that fishers can actively avoid areas where they will encounter excessive levels of bycatch.

Deemed values can be simple or complex, set in YYY_flagDV parameter (0=no deemed values for the fishery YYY, 1= simple, 2=complex). Simple deemed values are calculated in the same way as dynamic fish prices, but the final price of a species for each fishery is calculated by the scalar given in the XXXdeemed parameter (given for each fishery for each species). The scalar allows the deemed values to be higher or lower than market sale values. Typically deemed values are lower than the market values, so the scalar is <1. This means that generally fishers will avoid bycatch, because the 'deemed' price of landed fish will be lower than the price they can get at the markets, and they will be invoiced for the deemed price. However, in some cases, where market conditions are bad, even after being invoice for the deemed value of the fish, this at least guarantees some income, and may encourage fishing of bycatch species (see further discussion in the Hutton et al. 2010 report).

Complex deemed values take into account the costs of fishing, and are calculated as

$$DV_{cx,Y,x,t} = Prc_{cx,Y,x,t} \cdot \left(1 - CrewPr_{Y,J} - deemed_{CX,Y}\right) - Cost_{fixed,Y,J}$$

Where $Prc_{CX,Y,x,t}$ is the dynamic market price as calculated above; $Cre_{PrY,J}$ is the proportion of profit that is a crew share in fishery Y subfleet J (YYY_crewshare); $deemed_{CX,Y}$ is the deemed value scalar given in the XXXdeemed parameter (for each fishery and species); and $Cost_{fixed,Y,J}$ is the fixed cost of fishing ($/day) per fishery Y subfleet J.

17.3.3. Fishing costs

Costs are tracked at the subfleet level. The total operating costs ($Cost_{J,Y,m,y}$) per subfleet *J* of fishery Y in month *m* of year *y* consists of fixed, market dependent, harvest dependent and effort dependent costs:

$$Cost_{J,Y.m.y} = \left(Cost_{fix,J.Y} + Cost_{cap,J.Y}\right) + \left(Cost_{quota,J.Y,m.y} + Prc_{fuel,J.Y,m.y} \cdot Eff_{J,Y,m,y}\right) + \left(Cost_{unload,J.Y} \cdot H_{J.Y.m.y}\right) + \left(Cost_{gear,J.Y} \cdot Eff_{J.Y.m.y}\right)$$

where $Cost_{fix,i,j}$ are the fixed costs of subfleet J (YYY_fixedcost, $/day), which represent boat licence, management, insurance and maintenance costs and $Cost_{cap,i,j}$ are capital costs (YYY_capitalcost, $/day), mostly representing depreciation. These two costs are fixed during the simulation. The $Cost_{quota,i,j}$ are quota lease costs for subfleet *i* of fleet *j* for quota traded in month *m* of year *y* (calculated dynamically using quota price and quota trading routines, see below); Prc_{fuel} is the fuel price calculated dynamically (see below); $Eff_{J,Y,m,y}$ is the effort (in days) that a subfleet has expended in the month *m* of the year *y*. The second term of the equation include dynamic costs determined by the market (quota and fuel price). The $Cost_{unload,i,j}$ are unloading costs (YYY_unloadcost, $/tonne) and $H_{i,j,m,y}$ is the landed catch by subfleet *i* of fleet *j* in month *m* of year *y* – this is the harvest dependent component of the cost. $Cost_{gear,i,j}$ are gear maintenance costs (YYY_gearcost, $/day of fishing effort) currently set to be stable during the simulation. Like the fuel related cost, the gear maintenance costs are related to the effort expended.

The fuel price is calculated similarly to the fish price described in chapter 17.3.1.

$$Prc_{fuel,J,Y,m,y} = \left(a_{fuel} + Trend_{fuel} \cdot year + b_{fuel,month} + c_{fuel} \cdot residuals_{fuel}\right) \cdot \delta_{J,Y}$$

a_{fuel} is the baseline fuel cost (fuel_intercept, cents/litre); $Trend_{fuel}$ is the price trend coefficient fuel_trend_coefft (cents/litre) to be multiplied by the year of the simulation; the $b_{fuel,\ month}$ is the monthly market constant given in the FuelMonthCoefft parameter; and c_{fuel} is a constant (fuel_auto_coefft) to be applied to the externally read fuel price residuals $residuals_{fuel}$. The $\delta_{J,Y}$ is the subfleet's specific fuel consumption value (YYY_fuelcost, liters/day of effort).

Table 16. Parameters relating to fish prices and fishing costs

Parameter	Description
UseConstPrice	Parameter determining whether to use constant fish prices given in XXXsaleprice parameter (if set to 1) or if prices should be updated dynamically using market based calculations (if set to 0)
XXXsaleprice	Original fish price ($/kg) for species XXX and given for each fishery
K_num_markets in the	Number of independent markets determining fish prices (currently can

Parameter	Description
run.prm	only be 1 or 2)
MARKET_intercept1 MARKET_intercept2	Market-specific scalar for each species fish price (unitless)
MARKET_trend1 MARKET_trend2	Long term price trend coefficient constant for each market and species, to be multiplied by the simulation year number to get $/kg
JanMarketCoefft1 FebMarketCoefft1 …	Monthly market constant ($/kg) in fish price
MARKET_autocoefft1 MARKET_autocoefft2	Market specific constants to be applied to externally read price residuals
YYY_homeport	An array indicating which port each fishery and subfleet are associated with (each subfleet can only be associated with one port)
ports_markets in the *harvest.prm*	An array indicating which port is associated with which market (each port is only associated with one market)
DVstart	Day of the simulation run when deemed fish price values will start being used
YYY_flagDV	Fishery specific flag determining whether to use no deemed value (=0), simple deemed value (=1), or complex deemed value (=2). See chapter 17.3.2
XXXdeemed	Scalar applied to dynamically calculated fish price to get deemed value prices
YYY_crewshare	Proportion of profit (from 0 to 1) that goes to the crew in each subfleet, used for complex deemed fish value calculations and for calculating subfleet levels of profitability and future economic status.
YYY_fixedcost	Fixed cost of fishing, $/day, applied to each day of the simulation (not fishing days)
YYY_capitalcost	Capital costs of boat, $/day (each day, not fishing days)
YYY_unloadcost	Costs of unloading the catch, $/tonne
YYY_gearcost	Costs of gear maintenance, $/effort_day (only applied for fishing days)
fuel_intercept	Baseline fuel cost, cents/litre, used in calculating fishing costs
fuel_trend_coefft	Fuel yearly trend price coefficient, multiplied by simulation year to get cents/litre
FuelMonthCoefft	Montly fuel price constant (cents/litre)
fuel_auto_coefft	A constant on fuel cost to be applied to externally read price residuals
YYY_fuelcost	Subfleet specific fuel consumption, litres/effort_day

17.3.4. Quota trading

Quota trading constitutes an important aspect of the Economics submodel and has large effect on the profitability of a subfleet. It is based on the principle that individual quota is transferable (commonly referred to as ITQs). In Atlantis quota is traded among subfleets whereas in many fisheries quota is traded among individuals (quota owners). Quota trading is based on the principle that each subfleet aims to maximise profit. In other words, a subfleet that can increase its profit by purchasing quota and thus increasing its fishing activity will do so. Similarly, a subfleet that can increase its profit by selling quota and reducing fishing activity will also do so. Transferability of ITQs allows fishers to either

continue fishing or transfer (by sale or lease) their quota holdings to other fishers or, for efficient fishers, to accumulate quota by purchase or lease. Quota trading therefore occurs between vessel (subfleets) wanting to increase fishing effort and a vessel (subfleets) wanting to reduce their effort. Quota trading in Atlantis is based on priciples described in Newel et al. (2005) and Little et al. (2009).

Total and marginal revenue, rent and profit in the economics

There is an important distinction between the **revenue**, **rent** and **profit**. The **revenue** includes all payment received from the sales of the product (in this case fish). The **rent** is the revenue left when all costs except for labour costs are taken into account. Finally, the **profit** is the revenue left after accounting for all costs (including labour costs). Profit over a given timeframe (e.g. $/month or $/year) thus represents revenue after rent and labour costs are subtracted.

Marginal rent or **marginal profit** is the value generated from one extra sale operation. For instance, marginal profit from an extra tonne of fish sold, or an extra unit of quota leased and so on. Marginal values are important when assessing whether increases in economic activity will pay off. For example, initially spending money on a faster engine may bring high additional profits. However, increasing the scale of this fishing operations further may entail substantially larger expenditure and costs and marginal profits from the extra fish caught may be low or even negative. The latter large investment will therefore be less worthwhile. The shape of the marginal rent or profit function determines the optimal magnitude of the economic activity (e.g. the scale of the fishing operation).

Quota trading is done at the subfleet level and is applied only if a global flag quota_trading =1. Quota will be traded only for those species which are under quota. Fishing will only occur if fishing without quota is allowed (fish_withoutQ = 1) or if some quota is held (fish_withoutQ = 0); in the later case effort may be scaled down if insufficient quota is held for expected catches. Note this handling of fishing stems from the way in which quota fisheries are implemented around the world. In some countries quota must be held before fishing commences (fish_withoutQ = 0) or fishing may occur and quota is then sort to cover that catch before it is officially landed and sold (fish_withoutQ = 1).

The initial allocation of quota (TAC) among subfleets is given in OwnQuotaXXX_sub1, OwnQuotaXXX_sub2, etc. parameter (an array has as many values as there are fisheries). These values supersede those from TAC_XXX in the *harvest.prm* file – a warning to this effect is printed at the start of any run where the economics sub-model is used. For species that do not have quota for a given subfleet, the value should be set at 10^{12}.These quota values are used through out the run unless the Assessment submodel is used and TACs are determined dynamically (see Note! in chapter 16.1). If dynamic quotas are being used then the allocation of quota per subfleet is based on the proportion of the previous year's quota held by that subfleet.

Quotas are typically allocated for one year and quota trading only applies to subfleets and species that have quotas (TAC) imposed. Quota trading is done with the same frequency as the "final effort allocation" step of the economics sub-model; in most cases this trip length is assumed to be weekly, but it can also be on shorter time frames. As the year passes quota trading operations intensify, as the subfleets aim to either offload the quota they are unlikely to use up or buy more quota if their catch

exceeds expectations. Quota trading is determined by the cumulative versus expected catch, the proportion of quota left (in-hand), and the price of quota for that species and subfleet (determined dynamically, see below). The quota trading model assumes that subfleets whose marginal rents are highest (most positive) have the greatest incentive to lease or buy quota, while those with the lowest (most negative) marginal rents have the greatest incentive to sell their quota and not fish.

The trading is done in several steps:

1) *Decide if the subfleet 'needs' any quota to fill its scheduled effort*

A subfleet will consider buying additional quota if it expects to catch more fish than its current quota holdings. This expectation and consequently the quota demand is determined by: cumulative catch versus expected catch, amount of quota the subfleet has used up and expects to use. A subfleet will try to get more quota if the expected catch is higher than the user set proportion (prop_within) of the available yearly quota that is left. The proportion of quota that is left triggers the demand (often set at 0.9) and it decreases gradually through the year. The decrease occurs because subfleets do not want to be left with unused quota at the end of the fishing year and are less likely to buy extra quota at the end of the year needlessly.

2) *Decide if a subfleet has any extra quota it is willing to lease or sell*

The decision to lease out or sell quota is similar to the decision to buy or lease in quota and is based on cumulative and expected catch and the proportion of quota still available (quota in-hand). The proportion of unused quota that triggers the leasing out of quota is set by the user (prop_spareend) and gradually increases through the year, so that a subfleet is not left with any unused quota at the end of the year.

3) *Match sellers with buyers to maximise the profit and desirability of species*

Atlantis has two options to do quota trading. Just like in real life, quota can be traded on an individual species basis (sp_by_sp=1) or in species packages (sp_by_sp=0). The package quota is often used in real fisheries, where quota sellers have a collection of quota for different species and only lease them in combination, as they do not want to be left with quota of less profitable species. When quota trading is done in packages Atlantis will do profit maximisation calculations aiming to match the best combination of quota packages based on quota prices. The single species quota trading is described in Little et al. (2009) whereas multi-species trading is described in Fulton et al. 2007 (Appendix C, chapter B.8). Briefly, in the quota trading routine Atlantis will:

a) Rank subfleets on the basis of their marginal rent and identify those with the highest to lowest incentive to buy and sell quota (or lease quota in or out)
b) Assess whether a subfleet needs any extra quota (buy or lease in), or if the subfleet is willing to sell or lease quota out (steps 1 and 2 described above)
c) Match the quota buyers and sellers to maximise the available and desired species combinations and profits given the quota price. Once ranked the buyers and sellers are selected randomly with the probability proportional to their marginal rent
d) Allocate temporary leases or permanent sales. The quota can either be leased out temporarily and the leases are updated every month, or it can be sold permanently (see below).

e) Allocate temporary and permanent leases and permanent sales and update the quota in-hand for each subfleet. Repeat these steps until quota demand and/or supply runs out.

Temporary lease, permanent lease and sale of the quota

Quota trading is done with the same frequency as the "final effort allocation" step of the economics sub-model; in most cases this means weekly intervals, but it can also be on shorter time frames. If a subfleet has any available quota it does not need (i.e. it cannot fish efficiently or profitably) it can either lease or sell it. A temporary lease of quota is done on an annual basis. The owner of the quota retains ownership of the quota but leases it for one year. A temporary lease means that a subfleet (who retain ownership of the quota) will get lease income from their quota every month and that at the end of the year it gets all the quota allocation back.

A permanent lease is still a lease but based on a "gentlemen's agreement" among fishers. A subfleet agrees to permanently lease the quota and even though they retain ownership they do not get it back to reallocate to new leases at the end of the year. In a permanent lease situation the owner of the quota will earn monthly income from the lease (as in the temporary lease situation). Empirically this permanent lease arrangement occurs, for instance, when older fishers decide to lease quota to younger fishers and use it as a source of permanent income ("superannuation").

Finally, the sale of a quota means that a subfleet gets a set price for a quota and loses its ownership. Also once the sale is done there is no monthly income from the quota lease coming back to this subfleet.

The price of a temporary quota lease is calculated dynamically. The permanent quota lease is typically more expensive – the price is scaled by the perm_coefft scalar on the base quota lease price. Finally, the complete sale of the quota is usually even more expensive, as set in the buy_coefft scalar on the base quota lease price. Permanent transfers are more expensive because they should equal the discounted value of future profits that could be earned using that quota. Temporary transfers should equal the profit (or net revenue) from that unit of quota in that year.

Quota can either be leased temporarily, leased permanently, or sold. This is determined by the relative cost scalar of the sale versus lease for a single quota unit (see the box above) and randomly drawn probability (similarly to the gear switching or boat selling probability described in chapter 17.4.1). See Fulton et al. 2007 appendix C equations B.21-22 for further details. Briefly, the price of the lease and sale of the quota unit sets a probability that it will be leased and sold. Because the sale price is typically much higher than the lease price, the probability of sale is much lower than the probability of lease; the probability of a permanent lease usually lies between the temporary lease and permanent sale price. Then Atlantis draws a random number and compares it to the probabilities above. If the random probability is lower than the temporary lease probability but higher the permanent lease probability, the temporary lease operation occurs. If the random probability is lower than permanent lease and but higher than permanent sale, then permanent lease occurs.

17.3.5. Quota prices

Since quota trading is based on profit maximisation, the amount of quota traded will depend on quota prices. In Atlantis quota prices are calculated dynamically using either the model described in Little et al. (2009) (flagLease=0) or Newel et al. 2005 (flagLease=1). The quota prices are calculated in *Total_Quota_Price()* routine in **atquota.c**

In the model that is based on Little et al. (2009) (flagLease=0), the price of a unit quota of a species CX is calculated as the average marginal profit ($/kg) over the subfleets discounted by the interest rate (given in the global interest_rate parameter). The marginal profit is calculated as total profit divided by the total catch.

$$Prc_{quota,CX} = MargProf_{CX} \cdot \left(1 + \frac{1}{interest}\right)$$

In this model the quota prices change in response to the dynamic fish prices and fishing costs that determine the final marginal profits.

In the Newell et al (2005) model (flagLease=1) the average quota lease price (per subfleet) is a function of the market price of fish, the marginal cost of fishing, the annual catch by the subfleet, the monthly catch for that year, total quota available for that subfleet, and GPD growth rate. The environmental index, market fixed effects, seasonal fixed effects and error terms can also be included. The price of lease quota is unresponsive to demand. This method uses a more complicated 11 parameter function and is described in detail in Fulton et al. 2007 (Appendix C, Chapter B.8) and Kaplan et al. 2014 (equation 3).

If the UseMinValue global parameter is set to 1, then quota prices cannot fall below the minimum value given in a global minValue parameter (same for all species for which no quota price is estimated, or price is below the minimum value). If UseMinValue=0, quota prices for all species with no quota are set to 0

A parameter targetscale is used to give the relative weighting key target species have in making quota trade decisions. Effectively, each kilo of a target species up for trade is targetscale times more attractive than a kilo of a secondary or byproduct species (i.e. a lower value but still saleable fish under a quota).

Table 17. Parameters relating to quota trading and quota prices

Parameter	Description
quota_trading	If set to 1, then quota is traded in the simulation
fish_withoutQ	Whether fishing without quota is allowed (if set to 1)
OwnQuotaXXX_sub1 OwnQuotaXXX_sub2 …	Total allowable catch of quota (in tonnes) for each subfleet of each fishery (superseding the values for TACs set in *harvest.prm*). If set to 10^{12} then no quota for that species is applied.
prop_within	Proportion of yearly quota fulfilled before attempting to buy more quota (a subfleet will not buy new quota if it has used up less than the set proportion of the quota). This proportion applies to the start of the

	year and will increase automatically and slowly through the year
prop_spareend	Proportion of quota left at which point a subfleet will try to sell quota. The proportion will increase automatically and gradually through the year, so that a subfleet is not left with any unused quota at the end of the year
sp_by_sp	Whether quota trading is done by single species (=1) or by species packages (=0)
perm_coefft	Scalar (unitless) for permanent quota lease compared to the temporary lease
buy_coefft	Scalar (unitless) for permanent quota selling compared to the temporary lease
flagLease	Flag indicating whether to use Little et al. 2009 model to calculate quota price (=0) or Newel et al. 2005 model (=1).
interest_rate	The interest rate used for quota price calculation in the Little et al. 2009 quota price model
UseMinValue	If set to 1, quota prices are set at minimum values given in minValue parameter. Otherwise quota prices are allowed to fall to 0
targetscale	A scalar (unitless) used to give the relative weighting of key target species in making quota trade decisions.

17.4 Economic responses

17.4.1. Monthly updating of boat numbers

In the Economics submodel boat (vessel) numbers are regulated by the number of licences. A licence is necessary for a boat to be allowed to fish. The number of licences for each fishery is set in YYY_nlicences parameter and are read in at the start of the simulation. Boats are distributed at the subfleet level, but each fishery can only have as many boats (for the entire subfleet) as it has licences. The number of licenses cannot increase during the simulation and fisheries do not trade licenses; but if boats switch among fisheries (by switching gear, see below) they take their licence with them. To ensure consistent results during the parameter set up the users should pay attention to the number of licences held by the fishery – making sure that it is equal to or higher than the number of boats summed over all the subfleets in that fishery. Atlantis will quit on start-up if this is not the case.

Users can set up a 'buy back' scenario to reduce the number of licences per fishery. The buybacks are intended to imitate governmental licence/boat buyback incentives to reduce over capacity and investment in a fishery. The buyback date for each fishery is set in YYY_buybackdate parameter (set this to a large value if no buybacks are wanted) and the proportion of boats per subfleet that will be reduced as set in the YYY_propbuyback parameter. Note, that forced buybacks in Atlantis do not involve any cash flows – no money is actually paid for the buybacks, only the number of boats is reduced.

If the baseline Atlantis economics model is used (MultiPlanEffort =1) fishers can move across fisheries to simulate gear changes. This is not allowed in the alternative Dan Holland economics model (MultiPlanEffort =0). The vessel update routines are described in detail in Fulton et al 2007 Appendix C, chapter B5 and are only briefly explained here.

At the start of each month the *Update_Vessel_Numbers()* routine will assess the profitability of different vessels and assess whether any fishers want to switch to different fisheries. This is done to imitate changes in gears applied by fishers. In the real world, fishers will sometimes switch gears to target different, more profitable species, or at least supplement their gears with additional ones for some time. In Atlantis this is done by redistributing boats across fisheries. This is because gear selectivity and target species are defined at the fishery level, whereas all subfleets and boats in a fishery must have the same gear.

The *Update_Vessel_Numbers()* routine redistributes boats across different fisheries based on their current and expected profits. This is done in several consecutive steps:

1) Check if switching of boats across fisheries or gear flexibility is allowed at all. This is set if flagsupp_allowed > 0.

2) If a subfleet is profitable, complete gear switching is not needed, but the subfleet may still want to supplement its gear with additional types. This can be done in two ways. First, a fishery can "supplement" gear (flagsupp_allowed=1) or they can have "flexible" gear (flagsupp_allowed =2). In Atlantis gear supplementing means that at **least one entire boat** from a subfleet must switch to another fishery. This will be done only if expected profits (based on **last month's** profits per species) from gear switching for an entire boat are higher than gear supplementing costs set in the YYY_suppcost parameter (specific for each subfleet) (in $ per operator making the change). The "flexible" gear options means that **any proportion of a single boat** can be temporarily moved to another fishery, if it is profitable given the gear supplementing costs YYY_suppcost parameter. In Atlantis this simply means that some proportion of one boat's effort is moved into another fishery.

   The actual final proportion of boats that will switch a fishery (either due to supplementing or flexible gear) are then drawn from **a random distribution** with the maximum value set in a global prop_supp parameter. This means that the actual movement of boats among fisheries will vary across simulations and results from simulations with the same parameters will differ slightly.

3) If a subfleet is not profitable, then complete gear switching might be done if flagswitch = 1. Again, as with the gear supplement it will be done only if expected profits are higher than the gear switching costs set in YYY_switchcost parameter ($ per switching operation). However, gear switching is intended to imitate long-term changes and here Atlantis assesses expected profits **over a 10 year period** and compares them to the costs of gear switching set in YYY_switchcost. This means that users should set the YYY_switchcost value to a considerably higher value than YYY_suppcost parameter, in order to reflect a complete remake of a boat and its gear to a new fishery. If switching gears is profitable, then a **random proportion of boats** (but not higher than prop_switch value) moves into another fishery (switches gears).

4) If a subfleet is not profitable and switching gears is not profitable either, given the cost, some proportion of boats will be sold, i.e. fishers will leave the fishing business; the money from selling the boat leaves the fishery, it is not stored in the subfleet's cash reserves. The maximum proportion of fishers leaving is set in the prop_leave parameter. The actual proportion is drawn from **a random number**, with a prop_leave set as a maximum. This means that not all unprofitable boats will leave the business; other will be running into debt.

5) When a subfleet is unprofitable and its' debt levels are too high, subfleets will be forced to sell some boats. The tolerable debt is set in the YYY_TolDebt parameter ($ per subfleet). When the debt of the subfleet exceeds the tolerable debt for a period of time given in the cripple_period parameter, then some boats from the subfleet will be lost. The proportion of boats lost due to crippling debt is set in cripple_nboat_lost parameter. Note, that unlike prop_leave, prop_switch and prop_supp parameters that set a maximum proportion of gear switching, the cripple_nboat_lost is the exact proportion of boats from the subfleet that will be decommissioned due to too high debt.

6) Finally, if a subfleet is profitable and has enough cash reserves to afford a new boat (with boat price per subfleet given in YYY_newcost parameter, in $ per boat) AND purchases of new boats is allowed in the simulation (flagnewboat=1), then it might purchase a new boat if there are spare licenses in that fishery. Remember, a fishery is allocated the number of licenses at the start of the simulation, and the number of boats cannot exceed the number of licenses. So a subfleet of a fishery can buy a new boat if in the previous steps some boats had to be sold, or if the fishery initially had fewer boats than licences. The purchase of a new boat is set as a random event, so even if conditions are right at the start of a given month, the purchase may not happen (but it will happen eventually if a subfleet stays profitable and has enough cash reserves).

17.4.2. Port growth

Currently in Atlantis port growth includes growth of the number of boats associated with each port and population growth per port. The base port population growth is set in the *harvest.prm* file and their change can be set as explained in chapter 15.5.4. Port population growth is also influenced by port activity levels (in the *Port_Growth()* routine), the number of boats coming into the port. Port population sizes will affect the effort in recreational fishing effort models.

The number of boats per port will change as boats switch fisheries, because different fisheries are associated with different ports (see chapter 17.3.1). The number of boats associated with each port will affect the proportion of total fish weight that is sold in each market (chapter 17.3.1) and in this way will affect fish prices.

Table 18. Parameters determining economic responses through gear switching and changes in the number of boats.

Parameter	Description
YYY_nlicences	Numbers of licenses in each fishery, i.e. number of vessels/boats a fishery is allowed to have across all of its subfleets
YYY_buybackdate	A day of the simulation run when a proportion of boats given in the YYY_propbuyback parameter is taken out of the fishery (bought back). If buy back is not wanted set this date to a large value
flagsupp_allowed	If set to 1, subfleets are allowed to move across fisheries to simulate gear switching; if set to 2 subfleets are considered to have flexible gear (see chapter 17.4.1 for further details)
flagswitch	If set to 1, subfleets are allowed to switch between fisheries

YYY_suppcost	Cost of gear supplementing and switching (in $ per operator making the change).
prop_supp	Maximum proportion of boats in a subfleet that can supplement gears in any one month
prop_switch	Maximum proportion of boats in a subfleet that can switch gears in any one month
prop_leave	Maximum proportion of boats in a subfleet that can leave fishing business in any one month
YYY_TolDebt	The debt level (in $) above which a subfleet may be forced to sell some boats and will have reduce effort
cripple_period	A period of time (in months) above which subfleet will be forced to lose (sell) boats
cripple_nboat_lost	Proportion of boats lost after the cripple_period time of debt above the tolerable level
flagnewboat	If set to 1, a subfleet can buy boats when it is profitable
YYY_newcost	Cost of a new boat ($ per boat) for each subfleet

17.5. Scheduling of fishing effort and final effort allocation

Each subfleet makes an annual fishing plan based on the expected and historical catches and effort and based on the available quota. This annual plan is then updated monthly in response to the realised catches (if catches are too low, the effort might be increased). Details and equations are provided in Fulton et al 2007, Appendix C, chapters B.3, B.7 and B.12.

17.5.1. Annual fishing plan

The annual fishing plan first distributes the total yearly effort temporally through the months of the year and then allocates that effort spatially across boxes proportionally to the historical effort expended in each box (stored in "spatial black books" of each subfleet). The historical effort is read at the start of the simulation from the *economics.prm* file (JanYYYEffort_sub1, FebYYYEffort_sub1, etc.) for each subfleet and month for each fishery. During the simulation the spatial black books are updated dynamically to store average effort expended by each subfleet in each box.

The temporal effort plan is made based on the information of the historical temporal harvest and CPUE for each species (stored in the "black books" of each subfleet), fish prices and fishing costs. As with the historical effort, historical temporal harvest per species per subfleet per fishery is read in from parameters JanCatchXXX_sub1, FebCatchXXX_sub1, etc. and is then updated dynamically during the simulation (historical catch values are not spatial, but represent the total catch over the entire model domain). The historical CPUE values per month, per subfleet, per box are read in from JanYYYCPUE_sub1, FebYYYCPUE_sub1, etc. parameter and then are updated dynamically.

The historical harvest and CPUE information gives expected profit per species for each month of the year. This simulates the situation where fishers concentrate their effort in locations that are known to yield high CPUE at different seasons and also target species that have the highest prices, when accounting for any expected monthly price fluctuations.

There are two ways to calculate the expected profit for each month, determined by the OrigEconCalc parameter. If OrigEconCalc=1 then expected profit Pft_e for each species CX caught by subfleet J in the fishery Y in month *m* is calculated as

$$Pft_{e,CX,Y,J,m=}Prc_{CX,Y,m} \cdot H_{e,CX,J.Y,m} - Eff_{h,J,Y,m} \cdot Cost_{Eff,J,Y,m}$$

where $Prc_{CX,Y,m}$ is the historical price for that species in month m got by subfleets in a fishery Y (prices can be fisheries specific!); $H_{e,CX,J,Y,m}$ is the expected harvest of species CX in month m (based on historical records); $Eff_{h,J,Y,m}$ is the historical effort by the subfleet in that month; and $Cost_{Eff,J,Y,m}$ is the fishing cost per unit effort in that month ($/day) of the previous year. The fishing cost per unit effort is calculated as total cost minus fixed and capital costs (e.g. minus the first term in the equation given in chapter 17.3.3), divided by effort in that month.

Alternatively, if OrigEconCalc=0, the expected profit calculation takes into account the proportional contribution of each species to the total profit and is also based on the historical CPUE rather than harvest information only. In this case the expected profit R_e for each species CX caught by subfleet J in the fishery Y in month *m* is calculated as

$$Pft_{e,CX,Y,J,m=}Prc_{CX,Y,m} \cdot \frac{H_{e,CX,J.Y,m}}{Eff_{h,J.Y,m}} - \gamma_{CX,J,Y,m} \cdot Cost_{Eff,J,Y,m}$$

Here $Prc_{CX,Y,m}$ is multiplied by the harvest to effort ratio (so effectively the annual plan is based on CPUE information); and $\gamma_{CX,J,Y,m}$ is the proportional contribution of species CX to the total catch in that month.

If bycatch of endangered species (identified by SP_Concern parameter in *harvest.prm* file) is penalised by settting TemporalBycatchAvoid=1 both options of the annual effort schedule will take into account the historical bycatch (which is the same as catches) of species of concern and will downscale the effort in those months where bycatch is highest. **The steps described above results in an annual fishing plan based on profit maximisation principle only.**

In the next step, Atlantis allows for some human behavioural drivers to be included that are not directly economic. The final annual fishing plan is weighted by the fisher behavioural flexibility parameter YYY_flexweight set for each fishery and subfleet. The YYY_flexweight parameter reflects the flexibility of fishers to update their long term fishing tradition with new profit information. The final planned effort by subfleet J for each month *m* $Eff_{planned,J,m}$ is:

$$Eff_{planed,J,m} = flexwgt_J \cdot \left(newEffort_{J,m} - oldEffort_{J,m}\right) + oldEffort_{J,m}$$

Where $flexwgt_J$ is the subfleet behaviour flexibility (YYY_flexweight); $newEffort_{J,m}$ is the monthly effort schedule made above based on maximum profit and bycatch avoidance; and $oldEffort_{J,m}$ is the historical effort per month (read in from JanYYYEffort_sub1, etc and updated dynamically in "black books"). The equation above shows that if *flexwgt* parameter is 1 (very flexible) then the planned effort is equal to the new most profitable effort. Alternatively, if flexwgt is 0 (not flexible at all), then new information about the expected profits and bycatch is not taken into account at all, and planned effort is the same as historical effort. Some fishers seem to always go fishing to the usual places, no matter the profitability.

The general expectation used in the annual effort planning is that catches per unit effort will be the same as in the last year. The main routine used for annual effort planning is *Dynamic_Effort_ Allocation()* in **atEconomicAnnual.c.**

Target groups (the groups that the subfleet is specifically targeting and bases effort allocation decisions upon) are dynamically updated to match those with the highest expected returns (target groups that are no longer both profitable and desirable are stripped from the list and an effort switching notice is entered in the run log/event file). The target species identified in this way replace the list from the *harvest.prm* file, with the current target species list influences effort decisions, quota trading and any other fishery related decision dependent on target species.

17.5.2. Monthly updates to the fishing plan

If the base economic model is used (MultiPlanEffort=1) the annual effort plan is updated each month depending on the realised catches, availability of quota and subfleet specific YYY_choicebuffer parameter that determines how quickly the subfleet will update the effort if it deviates from the yearly plan. If realised catches are lower than expected, then an increase in effort might be planned. If catches are higher than expectation, the effort will be reduced. Also, high levels of debt can decrease the effort. This is done in the *Monthy_Effort_Schedule()* routine. The monthly update is not done for the Dan Holland economic model (MultiPlanEffort=0).

At the monthly update Atlantis will calculate the planned effort scalar $EffScale_{J,m}$

$$EffScale_{J,m} = \max\left((1 - choiceBuff), \max_{CX}\left(\frac{H_{expected,CX,J}}{H_{realised,CX,J}} \cdot \frac{1}{(1 + rank_{CX})^2}\right)\right)$$

The complicated equation above can be understood as this:

1) Rank all species CX targeted by a subfleet J by their CPUE (or Value per Unit effort, if UseVPUE=1). The most valuable species gets a rank of 0, the next one a rank of 1 and so on.

2) Calculate $\frac{H_{expected,CX,J}}{H_{realised,CX,J}} \cdot \frac{1}{(1+rank_{CX})^2}$ term for each species. This would be a species-specific effort scalar. The graph below shows the effort scalar for a range of expected to realised catch for the highest rank species (rank 0) shown in black, for the second one in blue, and for the third (rank 2) in green dashed line. Remember that in real simulations for each species only one value of the monthly effort scalar will be produced and it will depend on the expected versus realised catch of that species

3) Find the maximum monthly effort scalar across the species. If expected/realised catch < 1 (realised catch is higher than the plan), then the monthly effort scalar is <1 and the effort should be decreased. If the expected versus the realised catch of all species is the same, then the highest ranked species will give the maximum scalar. But the actual values across species will differ, e.g. if the ratio of expected to realised catch of rank 0 species is 1, then its effort scalar is 1. If the expected/realised catch ratio for the second ranked species is 5, then its effort scalar is 1.25. Here Atlantis will take the maximum value across species, i.e. 1.25 in this case.

4) Constrain the minimum effort by the (1-choicebuffer value), where YYY_choicebuffer is the subfleet specific parameter. This parameter defines how quickly a subfleet will downscale the effort if realised catch is higher than expected. In the graph below the red horizontal line corresponds to choicebuffer=0.2 and orange horizontal line corresponds to choicebuffer=0.8.

5) Contrain the maximum effort by the maximum monthly effort $Eff_{monthly,max}$ which is calculated as

$$Eff_{mothly,max} = monthSc \cdot \left(1 - downTime_J\right) \cdot Nboats_J$$

Where *monthSc* is the monthly scalar in the month_scalar global parameter giving a maximum number of days of effort a boat can apply (it can be bigger than 30 if for example one boat has several gears), and $downTime_J$ is the proportion of time a boat from a subfleet J must stay in the port for maintenance given in the YYY_minDownTime parameter. A hypothetical maximum monthly effort limit is shown with a dashed red horizontal line in the graph below.

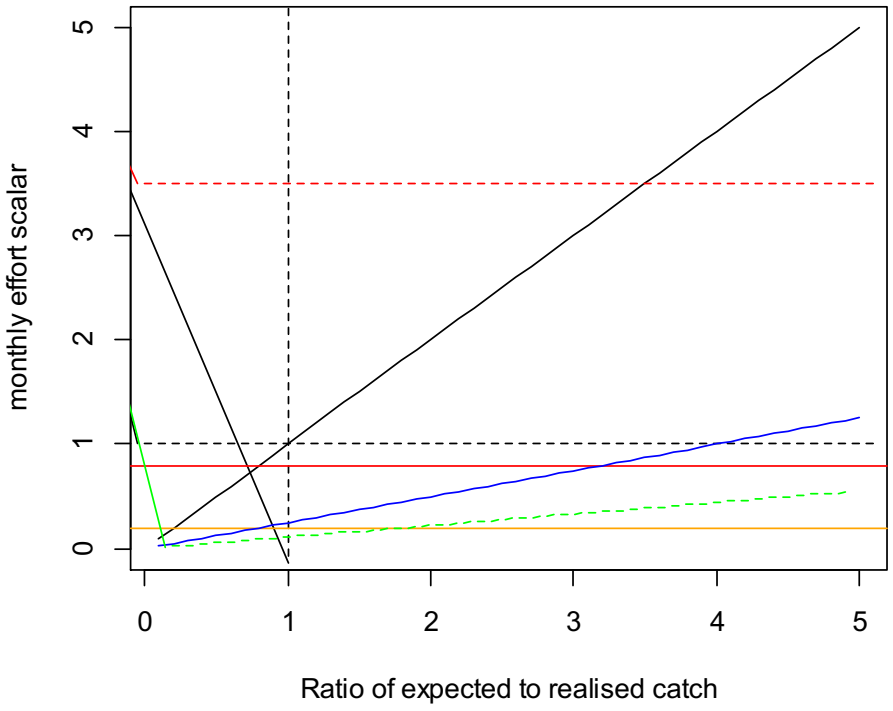


Figure 12. The monthly effort scalar calculation. See explanation in the text above.

6) Assess whether a subfleet has very high debt levels which will reduce its ability to fish. The tolerable debt levels for each subfleet are set in YYY_TolDebt parameter. If the expected monthly profit from fishing will not increase the debt above the tolerable levels then monthly planned effort is reduced by a proportion set in a global effort_reduction parameter.

17.5.3. Final effort allocations

There are two ways to determine final effort allocations. The baseline model (MultiPlanEffort=1) and Dan Holland model (MultiPlanEffort=0). Here we describe only the baseline multi-plan effort, used in Fulton et al. 2007. The Dan Holland model has been described in Kaplan et al. 2014.

In the baseline model with each new week of the simulation the fishers make the final decision on the temporal and spatial effort allocation (i.e. where to go and how long to fish). This is done based on the updated monthly effort plan, available quota left (if fishing without quota is not allowed and fish_withoutQ=0), and information on spatial CPUE distribution (or value per unit effort or bycatch per unit effort). The calculations are done by the *Multi_Plan_Effort_Final_Allocation()* in **ateconeffort.c**

The final effort allocation is done through the following steps:

1) Determine the spatial information about the CPUE values. This is done by calculating the ratio of CPUE in each box by the total CPUE. For these calcualtions the CPUE information of the last month is used (so not the memory period, as applied in other effort models, described in chapter 15).

   If value per unit effort information is used (UseVPUE=1) then boxes will be ranked not by CPUE but by VPUE details. Further, if spatial bycatch avoidance is used (SpatialBycatchAvoid=1) then boxes will highest bycatch in the last month will be downscaled.

2) Next, each box is weighted by their $mpasc_{J,j}$ (MPAYYY parameter from the *harvest.prm* file, see chapter 16.4), which will change the actual CPUE in this box if some of the box area is closed to fishing. This scaled value is used to calculate the ideal CPUE based effort ($CPUE_{eff}$) given the latest CPUE knowledge, monthly effort plan (as explained above) and fisher behavioural flexibility.

$$CPUEeff_{J,j} = mpasc_{J,j} \cdot Eff_{planned,J,j} \cdot \frac{Flexwgt_J \cdot \left(cpue_{t,j} - cpue_{t-1,j}\right) + cpue_{t-1,j}}{\sum_j cpue_{J,j}}$$

Where $cpue_{t,j}$ is the CPUE of subfleet J in the latest memory period t (typically the latest month); and $cpue_{t-1,j}$ is the CPUE of the subfleet J in the previous memory period (typically a month). The flexwgtJ (YYY_flexweight) parameter reflects how quickly fishers respond to the changing CPUE information – if flexwgt is 1 then effort is based entirely on the latest CPUE data, when flexwgt is 0 then new CPUE information is completely disregarded.

3) Next, the fisher flexibility parameter is used once again to weight the effort matrix, but this time it is not used to assess flexibility to the new CPUE information (old and new CPUE as in the step 2 above), but between the $CPUE_{eff}$ matrix calculated above (or VPUE, or DisPUE based effort, if appropriate flags are selected) and the monthly effort plan. This means that if fishers are flexible (YYY_flexweight is high or even 1), then the monthly effort plan is entirely overridden by the CPUE information, and if fishers are not flexible (flexweight = 0) they completely follow their monthly effort plan regardless of any information about CPUE.

$$idEff_{J,j} = Flexwgt_J \cdot \left(CPUEeff_{J,j} - Eff_{planned,J}\right) + Eff_{planned,J}$$

4) The ideal spatial *idEff* effort matrix of subfleet J by box j should further be scaled according to box distance to ports and the scalar determining travel costs of each subfleet $FCwgt_J$ (YYY_FCwgtscale unitless, ranging rom 0 to a very high value) and the variable cost per each subfleet $Cost_{var,J}$ (YYY_varcost, $/day of effort). This is used to account for the fact that fishers may be unwilling to go to the potentially profitable boxes, if they are too far from the ports.

The scaling is done by the sufbleet-box distance matrix $DistanceWgt_{J,j}$

$$DistanceWgt_{J,j} = \sum_{Z} \frac{Speed_Y \cdot dt}{Z \cdot distance^*_{z,j} \cdot Cost_{var,J}} \cdot FCwgt_J$$

where $Speed_Y$ is the speed of the fishery boats, not defined specifically by fisheries but for the entire commercial fisheries (Speed_boat, m hour^{-1}), dt is the number of hours between the timestep (usually 12, but can be 6 or 24), $distance_{Z,J}$ is a distance of the subfleet J to port Z, and Z is the total number of **ports active** for the subfleet J (a port is active if a fishery and subfleet operates from that port and the port is open, see chapter 15.5.9 for a similar case in dynamic fishing effort allocation).

5) Finally, the final effort matrix of subfleet J in each box j is calculated by interpolating between the ideal effort matrix calculated in step 3 versus the effort matrix in the previous step while accounting for boat speed and width of the model domain. This is

$$finalEff_{J,j} = DistanceWgt_{J,j} \cdot \max\left(1, \frac{Speed_Y \cdot dt}{width}\right) \cdot \left(idEff_{Y,j} - oldEff_{Y,j}\right) + oldEff_{Y,j}$$

The (*Speed*dt/width*) term was explained in the chapter 15.5.8 and simply calculates what distance a fishery Y can travel in one time step compared to the entire model domain (e.g. 0.1 of the maximum possible distance in the model). It shows that if the fishery can cross the entire model domain in one timestep and therefore find itself in any box of the model domain in one timestep, then the scalar will be 1 and the new effort will be the same as the ideal effort. If the scalar is smaller than one, then new effort will be a combination of the old effort and the ideal effort determined above. The $DistanceWgt_{J,j}$ matrix has been calculated in step 4.

NOTE!

The role of YYY_flexweight parameter in effort allocation

The YYY_flexweight parameter is used three times in the baseline model (MultiPlanEffort=1) effort allocation and should be carefully considered. It is used to:

1) update fisher response to latest price and CPUE information when making the annual fishing plan - chapter 17.5.1.
2) weight the fisher response to the latest CPUE information (whether to use the latest CPUE data or old CPUE data) – step 2 above
3) weight the fisher flexibility in updating the monthly fishing plan with CPUE information (if flexweight=0 then CPUE information is completely ignored and monthly effort plan is followed) – step 3 above

Once the effort has been allocation to each box of the model domain, the effort for all subfleets of a fishery is pooled within each box (to get fishery-specific effort per box) and passed to the Harvest submodel.

Table 19. Parameters related to effort allocation

Parameter	Description
MultiPlanEffort	If set to 1, then baseline economic Atlantis model is used to determine effort allocation. If set to 0 – Dan Holland economics model is used (see 17.5.3. for further details)
JanYYYEffort_sub1 FebYYYEffort_sub1 …	Historical average fishing effort (in effort days per box per month) – given for each subfleet of each fishery. This represents a part of subfleet's "black books"
JanCatchXXX_sub1 FebCatchXXX_sub1 …	Historical average catch per species (kg of catch per box per month) - given for each subfleet of each fishery. This represents a part of subfleet's "black books"
JanYYYCPUE_sub1 FebYYYCPUE_sub1 …	Historical average CPUE (kg of catch per one day of effort per box per month) - given for each subfleet of each fishery. This represents a part of subfleet's "black books"
OrigEconCalc	If set to 1 then the expected yearly profit is calculated as earnings minus costs. If set to 0, then profit calculation takes into account CPUE data and proportional contribution of each fish in the catch (see chapter 17.5.1)
TemporalBycatchAvoid	If set to 1, effort planning will take into account monthly historical bycatch and avoid months when bycatch is highest
SpatialBycatchAvoid	If set to 1, effort planning will take into account spatial historical bycatch and avoid boxes where bycatch is highest
YYY_flexweight	Parameter setting behavioural flexibility of a subfleet (scalar from 0 to 1). See box in chapter 17.5.3 for an explanation on how this parameter is used in three different calculations
YYY_choicebuffer	Parameter (scalar from 0 to 1) setting how quickly a subfleet will update the effort if it deviates from the yearly plan
UseVPUE	If set to 1, instead of CPUE calculations, Atlantis will use Value per Unit Effort values (which takes into account fish prices at a given time)
YYY_minDownTime	The minimum amount of time (proportion of a month, ranging from 0 to 1) a subfleet must stay in port and therefore cannot fish.
effort_reduction	Global parameter setting a scalar of effort reduction that a subfleet will experience if its debt is higher than the value given in YYY_TolDebt parameter
YYY_FCwgtscale	Scalar (from 0 to a very high value) that will affect the willingness of a subfleet to travel to distant boxes
YYY_varcost	Cost of fishing per subfleet per day ($ / effort_day).

18. OTHER INDUSTRIES

18.1. Pollution

At present this is done via forcing. Details on how to do this is under forcing in Volume 1 of the manual.

18.1. Other industries

Code exists for other coastal industries and will be added to the publicly available Atlantis in future. Please contact the development group for more details.

REFERENCES

Deporte N., Ulrich C., Mahévas S., Demanèche S., Bastardie F. (2012) Regional métier definition: a comparative investigation of statistical methods using a workflow applied to international otter trawl fisheries in the North Sea. ICES Journal of Marine Science 69: 331-342.

Dichmont SM, Fulton EA, Punt AE, Little LR, Dowling N, Gorton R, Sporcic M, Smith DC, Haddon M, Klaer N (2017). Operationalising the risk-cost-catch trade-off. CSIRO Oceans and Atmosphere. Fisheries Research and Development Corporation Project 2012-202, Brisbane, October 138pp.

Fulton EA, Smith ADM, Smith DC (2007) Alternative management strategies for Southeast Australian Commonwealth fisheries. Australian Fisheries Management Authority and Fisheries Research and Development Corporation report

Hutton T, Thebaud O, Fulton B, Pascoe S, Innes J, Kulmala S, Tudman M (2010) Use of economic incentives to manage fisheries bycatch: An application to key sectors in Australia's Southern and Eastern Scalefish and Shark Fisheries. Australian Fisheries Management Authority report (economic incentives to minimise bycatch and discards, deemed values, taxes on bycatch)

Kaplan IC, Holland DS, Fulton EA (2014) Finding the accelerator and brake in an individual quota fishery: linking ecology, economics, and fleet dynamics of US West Coast trawl fisheries. ICES Journal of Marine Science (Dan Holland economics model version, fish prices and quota trading)

Little et al. (2009) An agent-based model for simulating trading of multi-species fisheries quota. Ecological Modelling 220, 3404–3412 – quota trading study implemented in Atlantis.

Newell RG, Sanchirico JN, Kerr S (2005) Fishing quota markets. Journal of Environ. Econ. Manag. 49: 437–462 – an alternative quota trading version implemented in Atlantis

Pelletier D, Ferraris J. (2000) A multivariate approach for defining fishing tactics from commercial catch and effort data. Canadian Journal of Fisheries and Aquatic Sciences, 57:51-65.

Squires et al (1998) Individual transferable quotas in multispecies fisheries. Marine Policy 22: 135-159.

van Putten EI, Gorton B, Fulton B, Thebaud O (2013) The role of behavioural flexibility in a whole of ecosystem model. ICES journal of Marine Science, 70(1), 150-163. doi:10.1093/icesjms/fss175 (fisher behaviour, flexibility and historical information use)