Atlantis User's Guide
Part I:
General Overview, Physics & Ecology

Asta Audzijonyte, Rebecca Gorton, Isaac Kaplan,
Elizabeth A. Fulton

April 2017

TABLE OF CONTENTS

Acknowledgements

The authors want to express their sincere thanks to everyone who helped to prepare this manual, especially Shane Richards, Ingrid van Putten, Heidi Pethybridge, Cathy Bulman, Javier Porobic Garate, Thibaut Houitte de La Chesnais, Michael Sumner and many others who read the earlier versions of this manual and provided valuable feedback. We would also like to acknowledge that earliest stages of the compilation of the manual was facilitated by the Atlantis Summit (Hawaii 2015), funded by the CSIRO, US National Oceanic and Atmospheric Administration, and the Norwegian Institute of Marine Resources. Lastly, this work would not have been possible without the funding support through the years of the model development of the CSIRO, Australian Fisheries Research and Development Corporation (FRDC), National Oceanic and Atmospheric Administration (NOAA), the Gordon and Betty Moore Foundation, the David and Lucile Packard Foundation and .

FOREWARD

Before delving into Atlantis we would like to provide a little bit of background on the modelling framework and this manual.

Atlantis is just one of many marine ecosystem models, originally known as BM2 (BoxModel 2) it was christened Atlantis by Villy Christensen in South Africa in 2001. Marine ecosystem models have existed for more than 50 years, though they have only grown in popular use since the advent of (fast) modern computing power. They have grown from a biophysical focus to include more and more of the human dimensions. This is reflected in the structure of this manual, which sequentially works through the physical then biological before getting into the human dimensions. Atlantis was originally developed with an eye to temperate marine ecosystems and fisheries, though it has grown through time.

This manual (and the wiki) is intended to help provide guidance on what is contained in each submodel in Atlantis, with explanation of assumptions and some pointers on parameterisation and calibration. While it is not an explicit step-by-step recipe book on how to build a model, it will hopefully help new users to understand the core requirements pf Atlantis, while providing a handy reference for more experienced users on particular variants or processes they may want to add in a specific instance. For those (rare beings) who chose to read the manual end-to-end there is some repetition as we wanted to make sure those who were flitting in and out would contact the important information pertinent to the topic, similarly there is cross referencing between sections where important information is in another section of the manual, That cross referencing is not just for completeness, it is because the processes are interlinked and if you are having a difficulty it may be because of something in those other components.

Atlantis was never intended for anyone else's use, but was simply the ecological sandbox for me to explore my understanding of the marine world and how to represent it mathematically. It has been a huge shock to see it adopted by others and I remain deeply apologetic for how difficult it is to use (if I had ever known how it would be used I would likely have done things very differently). Atlantis remains a chimera of past decisions (often foist upon us by expediency or technological limitations) and current developments, as we address new questions. We will endeavour to keep the manual and associated wiki up to date as much as possible, but if you feel something is missing or unclear please let us know.

Note that I have shifted from I to we, because Atlantis now represents a group effort and all those who have participated in its development (especially Bec) and the construction of the manual (in particular Asta) should know they have my gratitude and deep appreciation for how outstanding a group of intellectual power they represent. Similarly, the community of Atlantis users represents a welcoming and stimulating group of collaborators!

For anyone new to Atlantis coming to grips with it is like climbing Olympus Mons. It is not for everyone (particularly as it is very data and time intensive!!). Much like when cooking a good stew, go lightly with the spice and concentrate on the hearty base constituents. Most of all try to relax and have fun.

Beth (Somewhere over the Indian Ocean, June 2016).

1. INTRODUCTION

1.1. Goals of this manual and where to go from here

Before launching into what Atlantis is and what it assumes it is important to explain what the intent of the manual is, how it can be used and the meaning of key terms that will be used throughout.

This manual is intended for Atlantis beginners who have some experience with modelling, running programs in command line (yes, this is how to run Atlantis!), and writing or using C or R scripts for analysing the outputs. While we have provided some information useful during model construction, this is not a comprehensive step-by-step recipe book on how to create a new model from scratch. We assume in many sections that you already have a somewhat operational Atlantis model that you want to use. The manual will also be useful for experienced users, because even experienced users often focus on the area of their interest and may not be thoroughly familiar with other aspects of the modelling tool. Finally, the manual will aim to provide starting points for developers of new Atlantis applications, but you will have to venture out and learn a lot more than what is covered in the manual.

NOTE!

As at the time of writing this manual (2016), two versions of Atlantis code are available for to the community1. The main version is called *trunk*, which is the usual way programmers refer to the main line of the code development. It was released in January 2015 and it includes all the latest developments and updates, and should be used for all new model applications. **This manual, with some rare exceptions, will apply to the *trunk* version only.** The old version *bec_dev* is no longer being actively extended and is intended for users who have been using Atlantis for a while and are unable to switch for whatever reason.

Atlantis is a rapidly developing tool. There are three main resources to keep up with updates:

1. The **Atlantis Wiki** is the first place to go for further information. Also remember to look at the page on *Frequently Asked Questions* for troubleshooting issues.

2. The **Atlantis Google group** is a good place to ask questions from the Atlantis user community and search topics of previous discussions.

3. The code itself! The code is commented and is the best way to understand how different processes are implemented. (We know that may sound confronting and challenging to those less used to code, but doing a word search on the code and looking at what it does is really how Beth and Bec go about answering any questions you ask them… even they don't keep it all in their heads!)

1 There is also Beth's development version, that contains features being actively developed at any instant in time. As those features are tested and verified (and so are less likely to contain a bug that will blow up) they are moved to the trunk version for general access.

Some important references for further information on model details and equations (see Atlantis wiki page on useful reading)

Models used as a basis for the physical and biology processes in Atlantis

1) Murray AG, Parslow JS (1999). The analysis of alternative formulations in a simple model of a coastal ecosystem. Ecological Modelling, 119, 149-166.
2) Murray AG, Parslow JS (1997). Port Phillip Bay Integrated Model: Final Report. Technical Report No.44, CSIRO Environmental Projects Office Canberra, ACT, Australia
3) Ebenhöh W, Kohlmeier C, Radford P J (1995). The benthic biological submodel in the European regional seas ecosystem model. Netherlands Journal of Sea Research, 33, 423-452.

Publications explaining which processes from the models above are used in Atlantis (initially called IGBEM and BM2).
READING THESE TWO PUBLICATIONS IS ESSENTIAL FOR ANY ATLANTIS USER!

4) Fulton EA, Smith AD, Johnson CR (2004a). Biogeochemical marine ecosystem models I: IGBEM—a model of marine bay ecosystems. Ecological Modelling, 174, 267-307.
5) Fulton EA, Parslow JS, Smith AD, Johnson CR (2004b) Biogeochemical marine ecosystem models II: the effect of physiological detail on model performance. Ecological Modelling, 173, 371–406

Other Atlantis (or Atlantis related) reports and publications

6) Fulton EA (2001) The Effects of Model Structure and Complexity on the Behavior and Performance of Marine Ecosystem Models. PhD Dissertation University of Tasmania
7) Fulton EA, Smith AD, Johnson CR (2003). Mortality and predation in ecosystem models: is it important how these are expressed? Ecological Modelling, 169, 157-178
8) Fulton EA, Smith ADM, Smith DC (2007) Alternative management strategies for Southeast Australian Commonwealth fisheries. AFMA, FRDC
9) Link JS, Gamble RJ, Fulton EA (2011) NEUS—Atlantis: construction, calibration, and application of an ecosystem model with ecological interactions, physiographic conditions and fleet behavior. US Dept. Comm. NOAA Technical Memorandum NMFS-NEFSC-218. 249pp.

1.2. Terms and conventions used in the manual

This manual is referenced throughout, and you can click on hyperlinks to take you to appropriate web pages or relevant parts of the manual.

Below are a few terms and standards.

Parameters and **variables**, used in various input and parameter files, are marked in orange.

Routine names are shown in *italics().* In computer programming, routines and subroutines are nearly synonymous terms describing a sequence of code that is intended of to be called repeatedly during the execution of a program. A routine performs a specific task and is relatively independent of the remaining code (although it might use variables defined or calculated in other routines).

Routines are grouped into **files**, used to perform a larger function. For example, the **atphysics.c** file includes a number of routines performing physical processes (gas exchange, deposition, diffusion). The code files will be shown as **bold_**. The files are then grouped into larger **libraries**, also shown in **bold**. Currently, Atlantis has 12 libraries, each containing dozens of files that, in turn, each contain several to dozens of routines.

Atlantis parameters can be grouped into **flags** and **parameters** (in narrow sense). **Flags** activate various optional routines; they are used in the if statements in the code (if flagA=1 do routine aaa). For further information about what flags do, check this link to flags in the biology parameter file on the Atlantis wiki. There are no clear rules on naming of parameters and flags, so the name alone does not indicate whether it is a parameter or flag.

Atlantis requires a lot of input files. Some of them are used for external forcing of the model (see Figure 2) and will be referred to as **inputs**. They are all listed in the *force.prm* file (note names of files used to run the model will be given simply in italics, to avoid confusing them with code files). Other input files define parameters or groups used in the submodels or simulation conditions and are referred to as **parameter files**. They are all listed in the BAT file used to initialise the runs.

Throughout the manual important messages and rules are placed in boxes that start with **NOTE!** Suggestions for good practice rules are given as **Good practice tips**. These tips give ideas on how to ensure reproducibility of your work and also highlight some common mistakes.

Most terms will be explained in the relevant sections of the manual. Here we will introduce only absolutely key ones.

Age groups defines age classes used in the model to represent fully age-structure groups (and can be set to anything from a few months to any number of calendar years). Many parameters for these groups are age-group-specific. All individuals in an age group are considered identical, unless the option of multiple genotypes is used, in which case all individuals within one genotype of one age group are identical.

Fisheries are fishing units that can have unique and very customisable characteristics, distinguishing them from other fishing units. The number and names of fisheries are given in the fisheries.csv parameter file. If fishing is simulated dynamically (see below) then fisheries behave more like separate fleets with dynamic characteristics.

1.3. What is Atlantis?

Atlantis is a whole-of-system 3D spatially-explicit marine ecosystem modelling framework based on complementary submodels used to simulate physical and biogeochemical processes, ecology, human uses of marine and coastal systems (primarily fisheries), and management. It is written in the C programming language and was originally based on two other models – the Port Phillip Bay Integrated Model (PPBIM) and the European Regional Seas Ecosystem Model (ERSEM). Atlantis combines different aspects of PPBIM and ERSEM (see Fulton et al. 2004a reference in the box above) and adds a lot of new specific features (Fulton et al. 2004b). Developed by CSIRO scientists in Australia, it has now been applied to more than 30 locations around the world (Fulton 2004; Fulton et al. 2011).

The model explicitly tracks the flow of nutrients through the trophic levels, typically using nitrogen as the "common currency" (carbon and phosphorous may also be used). At the core of Atlantis is a deterministic biophysical submodel that is spatially-resolved in three dimensions using a map made up of user defined boxes (polygons) and slab-like depth layers. The model geometry is flexible and should be matched to the major geographical and bioregional features of the simulated marine system. The oceanographic processes, such as water fluxes, salinity and temperature conditions are forced, typically based on outputs from oceanographic models (although they can be read in from observation data or calculated endogenously within Atlantis). Together with the optional forcing of nutrient inputs, these oceanographic processes drive deterministic primary production. The main ecological processes considered are ***production, consumption and predation, waste production and cycling, migration, reproduction and recruitment, habitat dependency and mortality.*** Biological functional groups include habitat-forming species like kelp, corals and sponges, as well as other benthic invertebrates, vertebrates, phytoplankton, zooplankton, refractory and labile detritus, and carrion. Predation is based on a prey availability matrix (similar in concept to the vulnerabilities used in the feeding arena theory, Walters and Kitchell 2001) with the final realised diet modified by the predators' gape limitations and feeding rates, spatiotemporal co-occurrences, the state of habitat refugia (for habitat dependent species) and in some instances nutritional quality and spawning status. Lower trophic levels are treated as biomass pools, whereas some key invertebrates and all vertebrates are represented as age groups. All individuals within an age group are considered identical and their conditions are tracked through energy allocation to structural and reserve nitrogen. The biological components are replicated in each layer of each of the polygons. Movement of organisms between the polygons is simulated by advective transfer for plankton and optional active migration for macrofauna.

Atlantis is a strategic, exploratory tool for investigating ecological processes, management policies, and broad consequences of alternative scenarios. It is not intended to replace traditional tactical models, such as annually updated stock assessments used to set fisheries quotas.

Atlantis has very flexible Harvest and Economics submodels that allow the user to set a large number of alternative harvest scenarios. A limited set of simple representations of other marine sectors (shipping, aquaculture, tourism, mining, energy generation and coastal land uses) can also be used. Pollution, coastal development, run-off and broad-scale environmental change can also be represented using forcing time series. The human use and impacts submodels are optional, which means that simulations can be run without any explicit exploitation or human impact.

The Harvest submodel is mostly focused on the dynamics of fishing fleets. Either stand-alone, or in combination with the Economics submodel, the human dimensions can be customised to simulate exploitation regimes and potential management schemes. These can range from a simple constant fishing rate on a single functional group to complex, dynamic fisheries comprised of interacting fleets, each with its own gear, bycatch and management characteristics, exploratory fishing, and all affected by compliance decisions and economic issues, such as prices of fish quota trading. Alternatively, Atlantis can be coupled with specific stand-alone fleet dynamics models, such as random utility or agent-based models.

Exploration of management outcomes in Atlantis can be done in two ways. The first is a simple scenario projection, where constant management rules are retained unmodified through time. The second is the full management strategy evaluation mode, where the monitoring, assessment and

decision making steps (and their feedback onto fishery regulations and behaviours) are represented. In the latter case, the Harvest submodel interacts with the ecosystem, but also supplies 'simulated data' to the Assessment submodel. The Assessment submodel is designed to simulate the assessment process with realistic levels of measurement uncertainty, in both measurement bias and variance. These 'simulated data' are derived from the outputs from the Biology and Harvest submodels, given a user-specified monitoring scheme that allows mimicking of a wide range of fisheries and survey information. The 'data' are then fed into an appropriate assessment model chosen by the user, such as Surplus Production, Adapt VPA or Bayesian integrated stock assessments. Additionally a range of ecological indicators can be calculated. The output of the Assessment submodel is fed to the Management submodel, designed as a set of decision rules and management levers that will affect fisheries. These include gear restrictions, days at sea, quotas, spatial and temporal zoning, discarding restrictions, size limits, bycatch mitigation, or dynamic reference points. It is also possible to track the approximate management costs, as monitoring and enforcement costs.

One of Atlantis' defining characteristics is its modular construction. A wide range of alternative assumptions and model implementations are provided in each of the submodels. This allows the user to choose the desired complexity– from a relatively simple model of a few groups with simple trophic interactions and catch equations through to complex models, with complicated stock structure, multiple fleets, detailed economics and multiple management levers. However, this flexibility requires that users fully understand the processes and do not resort to 'default settings'. **Atlantis is not a model where you can simply sit and pull multiple handles to discern what it is doing. It requires digging deeply into its underlying processes, particularly when first building and calibrating a new application.** It also means that building and parameterising a new and complex model will require a lot of time and data, and can also lead to large uncertainties around model outcomes.

1.4. Existing applications and questions studied

Most Atlantis development and applications have centred on evaluating alternative management strategies for multispecies fisheries, or exploratory analyses of multiple ecosystem pressures. To date, the Atlantis framework has been applied to over 30 ecosystems around the world (Fig. 1) primarily to study questions of:

1) overall assessment of alternative management strategies (Kaplan et al. 2012, Savina et al. 2013, Fulton et al. 2014)
2) historical impacts of harvesting (Link et al. 2010),
3) compliance with fishery regulations (Ainsworth et al. 2012a, 2012b),
4) robustness of ecological indicators (Fulton et al. 2005),
5) exploration of fisher behaviour (Kaplan et al. 2014),
6) impacts of global change, including acidification (Kaplan et al. 2010; Griffith et al. 2011, 2012, Fulton and Gorton 2014; Weijerman et al. 2015),
7) effects of body size changes (Audzijonyte et al. 2013, 2014, 2015).
8) the implications of model complexity (Fulton et al. 2003, Fulton 2004)

More recently Atlantis has been used to consider the impacts of marine and coastal industries other than fishing.

Figure 1. Some existing applications of Atlantis modelling framework and specific model domains (see Table 1 for details on numbered models).

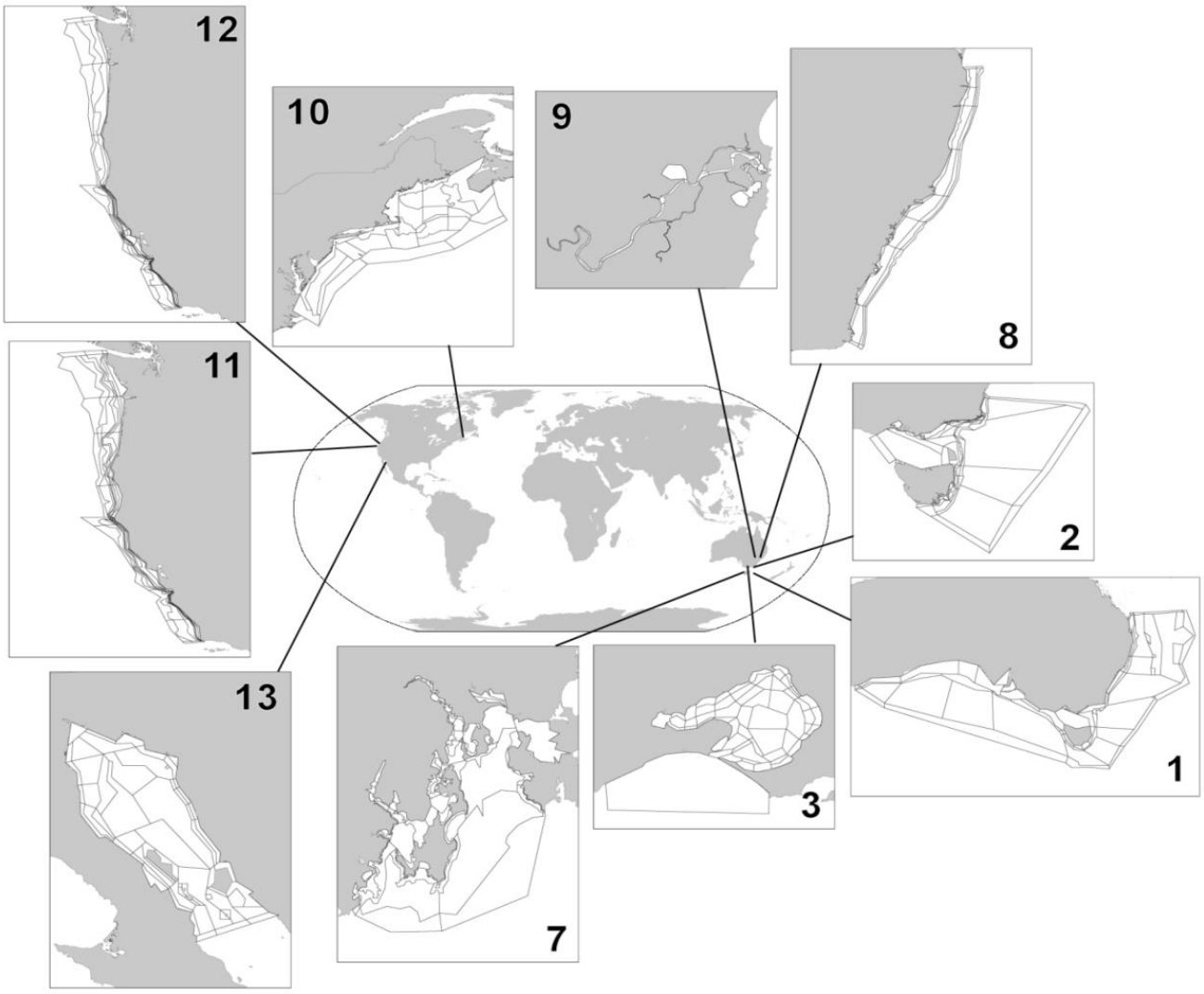


Table 1: Examples of Atlantis model applications and their main purposes: P = primary use of the model, S = secondary use, E = exploratory use beginning (modified after Fulton et al. 2011). Some of the model descriptions are included in the useful reading page on the wiki. An updated list of models under development is also available on Atlantis webpage

System	Com-plexity	Under-standing	System-level MSE	Fishe-ries	Nut-rients	Mining	Climate im-pacts	Catch-ments	Indica-tors	Management	Reference
Australia											
1. SE Australia		P	P	P			S		P	P	Fulton et al 2007
2. Victoria/Tasmania		P		S	S		S		P	P	Smith et al. 2010
3. Port Phillip Bay	P			S	S				P		Fulton et al 2004b
4. Westernport		P			P			P			Savina et al 2005
5. E Tasmania	P	P		S			E			E	Johnson et al 2008
6. Tasmania		P		P						S	-
7. Derwent-Huon		P	E	E	E		E	E		E	Savina 2009
8. New South Wales	P	P		P		S				S	Savina et al 2008
9. Clarence River		P		E	E		E	P	P	E	Hayes et al 2007
North America											
10. NE USA		P	P	P					S	P	Link et al 2010
11. California Current		P	P	P			S		S	P	Brand et al 2007
12. Central California		P		S						P	Horne et al. 2010
13. Chesapeake Bay		P		P	P		S			P	Ihde et al. 2016
14. Gulf of California		P	P	P	S		E	E	E	P	Ainsworth et al. 2012b
15. Gulf of Mexico		P	P	P	S				S	P	Ainsworth et al. 2015
16. Campeche Bay		P	P	P							Ortega-Ortiz and Ainsworth in prep.
17. Desoto Bay		P		P						P	Gosnell S., unpubl.
Other areas											
18. Guam		P		P	P		P	P	S	P	Weijerman et al 2015

The greatest challenge in implementing an Atlantis model is evaluating uncertainty of parameters and model forecasts. Traditional sensitivity analyses are not possible in models as complex as Atlantis, and therefore parameter uncertainty must be assessed in other ways. New means of achieving this are under development, but historically it has been done using 'bounded parameterisation', where extreme sets of parameters still giving realistic biomasses are used to explore sensitivity. For further

discussion on evaluating uncertainty in Atlantis, see chapter 3.2 and 6.1 in Fulton et al. 2007. To read about uncertainty in ecosystem models in general see Link et al. 2012.

For this reason Atlantis is well suited for broader scale exploratory analyses, "what if" scenarios, general evaluation of management strategies and other questions that draw on its ability to simulate multiple interactions between environmental factors, different marine species and humans. It is important to note that predicted biomasses of some functional groups are not necessarily expected to match biomasses predicted from other kinds of models, such as stock assessments (although ideally the trends would match if the same fishing pressure is included in both models). In general Atlantis produces realistic ecosystem structure and function, with biomasses of the correct order of magnitude and plausible dynamics. However, the details can be uncertain and hard to predict with a great deal of precision. For this reason, Atlantis should not be used for tactical short term advice such as setting quotas. However, it could inform fisheries models on factors such as variable natural mortality, as long as the inherent uncertainties are well recognised. The strategic nature of Atlantis is described in detail in Fulton et al. (2011).

Table 2: Some example findings from Atlantis applications (modified after Fulton et al. 2011)

Model	Findings
SE Australia (SE model)	- Human behavioural uncertainty produces unanticipated outcomes of management decisions and are important to future adaptive capacity - If companion TACs are implemented this leads to the ecological decline of weak stocks or significant economic costs (e.g. 75% of the catch foregone) - Relative biomass indicators, ratios of biomass (e.g. pelagic:demersal), size structure and size at maturity are consistently the most robust indicators of the effects of fishing (at fine or very large scales) - Decline in fish body size can greatly increase predation mortality through positive feedback loops in predator-prey age structure. This can prevent complete recovery from fishing, leading to a new and often lower equilibrium biomass - Small pelagic fish (like sardines) play a much less important role in temperate Australian ecosystems than in classical upwelling systems
Victoria/ Tasmania (SM model)	- Relative biomass indicators are consistently the most robust indicators of the performance of spatial management (though their performance can be weak at regional scales due to noisy data resulting from overlapping process scales) - Universal fisheries reference points are unlikely to work - Some indicators are only effective at small or very large scales due to how species are aggregated and distributed - Monitoring design is critical for long-term interpretation of time series
Eastern Tasmania (SETas model)	- off Tasmania there appears to be minor effects of fishing squids, but large impacts of targeting myctophids, suggesting a wasp waist system
New South Wales (NSW model)	- The size of no-take marine protected areas can be important for their ultimate success, but the redistribution of the catch and effort outside the closed areas is even more important - Not all species benefit from spatial management (e.g. prey species do not)

Model	Findings
	- Moderate levels of fishing pressure can lead to an increase in biodiversity - The explicit representation of ontogeny can be important for correctly representing the dynamics of marine species that show large spatial or habitat shifts with age
NE United States (NEUS model)	- Despite their complexity, clear patterns are detectable in large complicated biophysical systems (making calibrated predictions feasible) - Modelling squid and shrimp is extremely difficult (a finding recurring in many ecosystem models), but should be attempted as it can lead to new system understanding - Modelling ecology is much simpler than modelling a coupled ecological-anthropogenic system - Well calibrated models have high model skill, but typically only for those system components informed by data
California Current (EMOCC model)	- Catch shares are more economically efficient, even if catch levels remain unchanged - Monitoring and enforcement is critically important for fisheries management and guidance of fleet behaviour - Ocean acidification is likely to lead to the reduction in biomass of predators dependent on shelled invertebrate prey; in contrast other species (e.g. jellyfish) are likely to increase in biomass - Ratios of functional group biomass (e.g. piscivore : planktivore) are strong indications of community restructuring under fishing
Central California (ECCAL model)	- Fleets are differentially effected by changes in management rules or environmental conditions (some suffer significant economic declines while others see substantial increases) - Shifting gears to minimise bycatch and habitat destruction does not restore age structures, but spatial closures can
Gulf of California (GOC model)	- Reported catch values are insufficient to cause observed declines in species biomasses (by 3-5 times for most species)

1.5. How does Atlantis compare to other marine ecosystem models?

The key characteristics of Atlantis is that it is 1) whole-of-system, 2) spatially explicit, 3) age structured, 4) deterministic, and 5) comprised of submodels that simulate human impacts and management.

1) 'Whole-of-system' means that Atlantis aims to represent the entire ecosystem (even if greatly simplified) rather than focus on a few species of particular interest. It spans environmental drivers, habitats, the foodweb, and major human uses (fisheries in detail, but also simple representations of tourism, mining, energy generation, ports, shipping and the other coastal uses such as urban developments). Many other 'whole-of-system' or 'end-to-end' models are being developed but at present Atlantis tends to include more explicit oceanography and nutrient cycling than foodweb

models such as Ecosim with Ecopath (EwE; Christensen and Walters 2004) and a more extensive foodweb than targeted system models like OSMOSE (Shin and Cury 2004).

2) Atlantis is spatially explicit, which makes it a useful tool to explore spatial dynamics of various ecological and socio-economic processes (assuming you can parameterise them). In this way the model is similar to Ecospace (Walters et al. 1999), OSMOSE , SEAPODYM (Lehodey et al. 2008), and other models that operate on an oceanographic domain (Fiechter et al. 2014).
3) Atlantis includes two means of representing species – biomass pools (similar to those in EwE, these pools can include a small number of life-stage stanzas) and fully age-structured, where the average size-at-age of a typical individual is tracked along with the membership of that age group. Typically, biomass pools are used for invertebrate groups and age-structured for vertebrates, but that is not a strict requirement. The representation used for a species is dictated by the modeller in the .csv file used to define the configuration of the model groups. The realised growth and size of species represented through age-structured groups is dynamically dependent on food intake, whereas feeding interactions and reproductive output depends on the realised size and condition. This flexible age-size relationship adds temporal variation in size-at-age, essential for capturing important ecosystem dynamics and useful for exploring the ecological and fisheries consequences of changes in individual size and life-histories (Audzijonyte et al. 2013, 2014).

   The combination of spatial structure and temporal age-size dynamics means that Atlantis can produce realistic ecosystem dynamics both temporally and spatially and a lot of attention has been given to developing the models ability to capture age-size dependencies, temporal, habitat and spatial variation (Fulton et al., 2004c). The Atlantis users are therefore encouraged to make full use of these aspects. Alternatively, if the spatial and temporal age dynamics is not important, it might be better to consider using simpler and most likely computationally faster models.
4) In its standard form, Atlantis is not an agent-based (individual-based) model, like OSMOSE or InVitro (Gray et al 2006)2. It simulates temporally dynamic size-age relationship, but at a given time all individuals in an age group in one cell are identical (variation between cells is possible if a species is sedentary). Thus, Atlantis models processes based on representative "average" individuals for an age group, not true (separate and unique) individuals. This makes Atlantis computationally faster, but it does not simulate inter-individual variation and any potential emergent system properties.
5) Atlantis does not simulate stochastic environmental and ecological processes. Although Atlantis can include a small amount of recruitment timing variation, it is generally a deterministic model. This makes Atlantis different from models such as OSMOSE where stochastic processes are an integral part of the model formulation.
6) Atlantis was the first ecosystem model to include management strategy evaluation options - i.e. the inclusion of each part of the adaptive management cycle – the biological world, the human uses (especially fisheries), monitoring and reporting, assessment and management. EwE and OSMOSE now also include an MSE capacity. New features of human impact are constantly being added to Atlantis and the existing set-up is very flexible to customisation. Fisheries are represented in the most detail, but human impacts can also include nutrient and pollutant inputs, tourism, ports, shipping, urban areas, energy generation, heavy industry, mining and other marine and coastal industries.

2 An agent-based extension exists for Atlantis, but it is not currently included in the standard Atlantis package and will not be discussed in the manual

1.6. Main assumptions

Like any model, Atlantis has a lot of assumptions and it is important to keep them in mind when using the model and interpreting its outputs. The key assumptions are listed here, and they will also be listed in the description of different submodels and processes.

1. While not an ecological assumption the first and foremost assumption of Atlantis is that box 0 is a boundary box and box 1 is a dynamic box. If this assumption is violated things will go **terribly** wrong (most likely you model will not run at all).

2. Atlantis is a deterministic model, which means that a specific combination of parameters and initial conditions will produce the same results (except for possible very minor differences due to the way computers round small numbers).

3. State variables, such as salinity, oxygen, nutrients, sediments, organisms – are tracked as passively or actively advected or moving tracers. Passive tracers are distributed by water flows, whereas actively moving tracers (mostly benthic invertebrates and vertebrates) can have different types of movement (density-dependent, seasonal migrations, sedentary).

4. Each cell of the model (one depth layer of one spatial box) is considered uniform in its environmental variables and hydrodynamic processes. The conditions of these variables can change at each time step (usually every 12h) depending on the water fluxes between cells. Epibenthic cells can include different habitats, modelled as a proportion of the whole cell. This proportion is treated abstractly (via statistical means) and is not given a specific geographic location within the cell. The availability of habitat refugia can scale ecological and fisheries processes, such as predation intensity or vulnerability to capture.

5. Predation is determined by the user defined predator-prey interaction matrix that sets a maximum availability of each prey biomass available to a specific predator. The realised consumption is then modified by the actual spatial overlap, biomass of the prey, habitat refuge for the prey, gape limitation of the predator, spawning time of the predator, possible effect of acidification or contaminants on the predator, and availability of fisheries discards as food. The realised consumption therefore can be very different from what is set in the predator-prey matrix. However, if the predator-prey matrix does not allow a specific interaction (the value is set to 0), predation will never occur.

6. Within a layer of a single box, all individuals within an age group or within a single genotype of an age group (if several genotypes per age group are used) of a fully age-structured group are identical. Condition of these cohorts may vary among cells depending on the local food availability and parameterisation used. Fully age-structured groups are typically used to represent vertebrates.

7. Biomass pool groups are represented as a single pool per model cell (one layer of one box). Age structured biomass pool representations can be used (where a separate biomass pool use used per life history stage for the group). Invertebrates are often represented using biomass pools.

8. Energy flow through the system is primarily tracked using nitrogen as a "currency". The initial source of nitrogen in the system and each trophic group is set in the initial conditions file. Throughout a simulation nitrogen can be added into the system through optional forced inputs (e.g. river inputs) and atmospheric deposition. Nitrogen is lost through optional burial in the sediments ("decay"), denitrification, loss to the atmosphere (outgassing) and any optional forced sinks. The energy flow

through diatoms or other user defined Si limited functional groups is tracked through both N and Si. It is possible to setup the energy flows to be tracked through nitrogen, carbon and phosphorus, but that requires significant additional input file preparation and computing resources, and is recommended only in systems where P limitation is essential.

8. Energy allocation in vertebrates is tracked as structural and reserve nitrogen. Structural nitrogen represents bones, scales, and other hard parts that cannot be reabsorbed if feeding conditions are poor. Reserve nitrogen represents muscle, gonad, and other soft structures that may be used as an energy reserve if conditions are poor. Structural nitrogen cannot decrease throughout an individual's life-time (an individual cannot get smaller). Reserve nitrogen is used during reproduction or starvation and its ratio to structural nitrogen can change through time.

9. Maintenance energy costs are assumed to be covered through assimilation efficiency and optional total-weight specific respiration. However, most currently developed Atlantis applications do not include the optional respiration. Energy loss from reproduction is subtracted from spawn production as a functional group specific constant. There is no explicit inclusion of movement costs, although the time taken to move between cells is explicitly accounted for.

10. The earliest, larval life stages are not modelled explicitly. The recruitment type is defined by the user and occurs at a set time after spawning.

2. INSTALATION AND RUNNING

2.1 How to check out and install the code

Atlantis is dynamic and new modifications are introduced almost every month. In addition, Atlantis can run on Windows, Mac and Linux (Unix) operating systems. Therefore, you cannot download one executable, but have to check out and compile the code yourself. This ensures that you have the latest version of the code (and please remember to update it regularly!) with all associated libraries required to run Atlantis. As new functionality is added to Atlantis, sometimes code updates also means that new parameters must be added to the parameter files. While this can be frustrating, it is still highly recommended to update the code at least once a month, as updates also include bug fixes. To keep informed on new parameters the users are advised to subscribe to Atlantis wiki blog posts. In addition, if you need help from the model developers you are much more likely to get help quickly if they don't have to update your model first!

Atlantis source code is hosted in a subversion (abbreviated as SVN) repository. SVN is a commonly used version control system that is used to manage the collections of files that are changed over time. It keeps a history of the code base and the changes that have been made to it, so developers can recall past code versions for reproducibility. It also allows several people to work on the code simultaneously and merge it back together. **As a normal user you will have read-only access to the Atlantis code repository.**

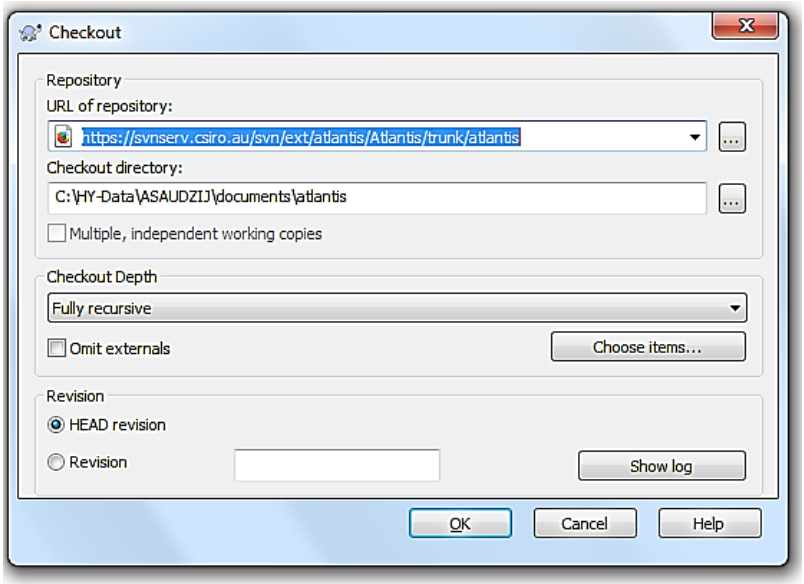
If you want **to modify the code**, please check out wiki for instructions on how to create a new branch from your working copy or merge your modifications with new updates. The Atlantis wiki also has more information on modifying the code, such as creating and adding a new input parameter or adding a new type of functional groups

To check out the code and install the code on your computer follow these steps:

1. Get access to Atlantis SVN repository by joining the Atlantis Wiki, signing the user licence agreement and requesting access to the SVN repository. The response time on this should not take more than a couple of days (if it does poke Beth again as she is likely on the road and has not seen the email).

2. Check out the code from SVN to your computer. Experienced users can check out the code by using svn commands through a command line terminal (dos or terminal window). Alternatively, you can use special software to make things easier.

Windows users can use the free package TortoiseSVN, which works like a plugin for Windows Explorer. To check out the code with TortoiseSVN first install it on your computer. Next, create a new folder where you will keep the code, e.g. C:/AtlantisCode. Rick-click the mouse on this folder and chose "SVN Checkout…" option. This will bring up the TortoiseSVN menu:



Enter the URL of the code depository, provided on the Atlantis Wiki source code page, making sure the directory to install the code on your computer is correct, and chose "Fully recursive" Checkout Depth and "HEAD revision". Detailed instructions on how to check out the code with TortoiseSVN are available on the wiki (here).

Mac users can check out the code using the app Versions, while Linux users can use RedBean.

NOTE!

When updating the code, make sure you always update to your main directory where all your code is located, such as C:/AtlantisCode, rather than any subfolders within it. This will ensure that libraries are updated properly.

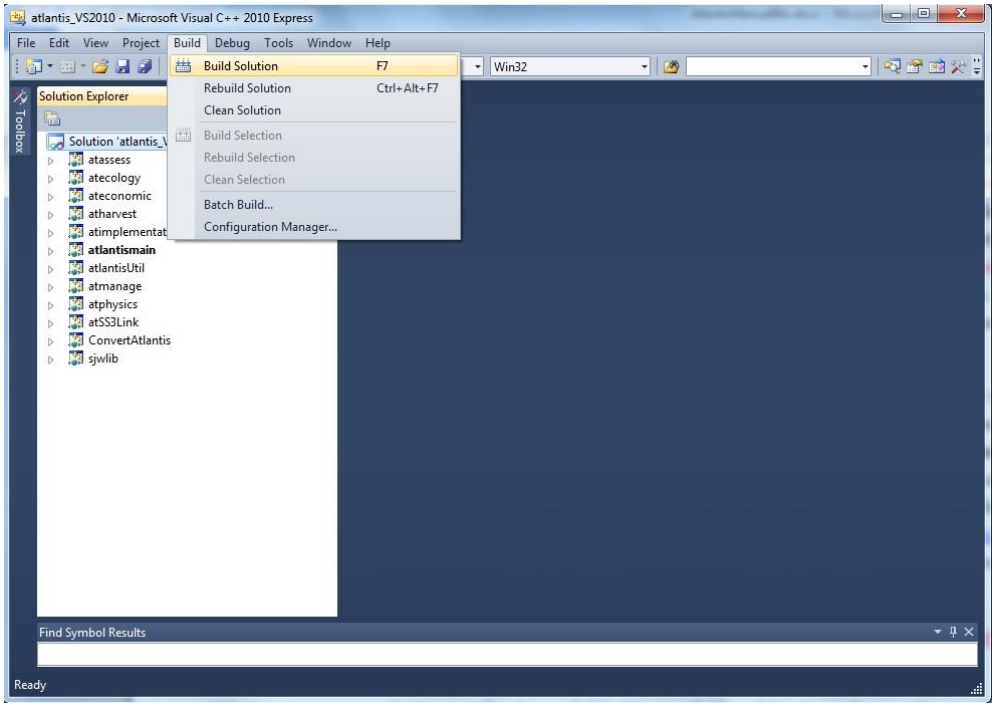
3. Install software for compiling the code. Compiling Atlantis code in Windows and Linux operating systems is easy. It is **a lot harder to set up on Mac operating** systems (although once installed everything runs smoothly enough).

Windows users can build the code using freely available Microsoft Visual Studio software. Currently supported versions are 2005/2008/2010/2012 and, as of March 2016, you can download 2010 and 2012 Visual Studio installers from the Atlantis wiki by clicking on the paperclip sign on the wiki.

Instructions for software and building Atlantis under Linux and other platforms are available on the wiki. If you are a Mac user you may find the use of comfort food at this stage a good idea, each Mac installation appears to face its own challenges, sometimes it goes very easily other times you can spend a day or more wrestling to get one of the supporting libraries to work with your architecture. There does not seem to be any rhyme or reason to this! Macs are just special that way.

4. Install the NetCDF 4 libraries. These libraries are required for running Atlantis. Detailed instructions on installing NetCDF and updating the path are available on the wiki. It is very important that you do this properly, as Atlantis will not run otherwise. Windows users can download pre-compiled libraries, Mac and Linux users will need to down load the source and compile locally.

5. Compile the code. In Windows compiling the code does not require any programming skills. All you need to do is double-click the Visual Studio SLN file (chose the version suitable for your Visual Studio, e.g. Atlantis_VS2010.sln) located in the main directory of where you checked out the code (e.g. C:\programs\Atlantis). This will load the project into Visual Studio. Then go to the Visual Studio Build menu and chose "Build Solution". Detailed instructions are given on Atlantis Wiki building page.



Once the build has successfully completed you will find the latest executable *atlantismain.exe* in C:\...\Atlantis\atlantismain\Debug folder (where … indicates your designated directory for the code)

NOTE!

You will need to recompile or build the code anew every time you update it from the SVN repository

On Linux (and Mac) platforms GCC is recommended for compiling the code. Other options for compiling are also available for users who for any reason do not want to use Microsoft Visual Studio or GCC. These options include the Intel compiler and Cygwin. However, these are not recommended compilers and users will potentially have to work out solutions to issues themselves.

6. Create a directory for your simulations.

This is where you will keep all your parameters and run outputs. This could be called C:\AtlantisRuns\... This is NOT the same directory as the code itself, so please make sure you understand the difference. If you use several models, such as California, SE Australia and Antarctica, might want to keep parameter files of each model in separate directories, e.g. C:\AtlantisRuns\California\, C:\AtlantisRuns\SE_Australia\ and so on.

Please make sure not to use spaces in the name of the directory (e.g. use C:\AtlantisRuns or C:\Atlantis_Runs not C:\Atlantis Runs as the core code of Atlantis was written before spaces were allowed in directory names and it may not correctly read file names if the directory structure includes spaces3).

The **South East Tasmanian (SETas) model** is freely available from the Atlantis SVN repository as a learning tool. It is a simple model with only a few boxes and few functional groups. All required input and parameter files for SETas model can be downloaded from the wiki and used as templates for new model applications.

Make sure you use SVN checkout to download these files. Don't use the web browser to copy the files. Also make sure you download files for the correct version of the Atlantis code (*trunk* or *bec_dev*)

7. Check out or copy files of your specific model and latest Atlantis executable into the run directory.

Also copy the latest executable *atlantismain.exe* from the C:\...\Atlantis\atlantismain\Debug folder into the run directory you created before (e.g. C:\AtlantisRuns).

The executable **must** be in the same directory as the main BAT run file, required to initialise your runs

NOTE!

3 We do endeavour to update things for new naming conventions, but we also find it is better to err on the side of caution.

Make sure you copy the new executable file from …Atlantis\atlantismain\Debug into your run directory every time you update and recompile the code. Forgetting to do this is one of the more common mistakes.
Alternatively, you can insert a line into the start of your BAT file to copy it before the start of each run if you want to make sure you always have the latest version.
This would look like:

copy C:\...your Code directory \atlantismain\Debug\atlantismain.exe .
The dot at the end of the line means that the file atlantismain.exe from Debug directory will be copied to the same directory as the BAT file.

See the wiki for details.

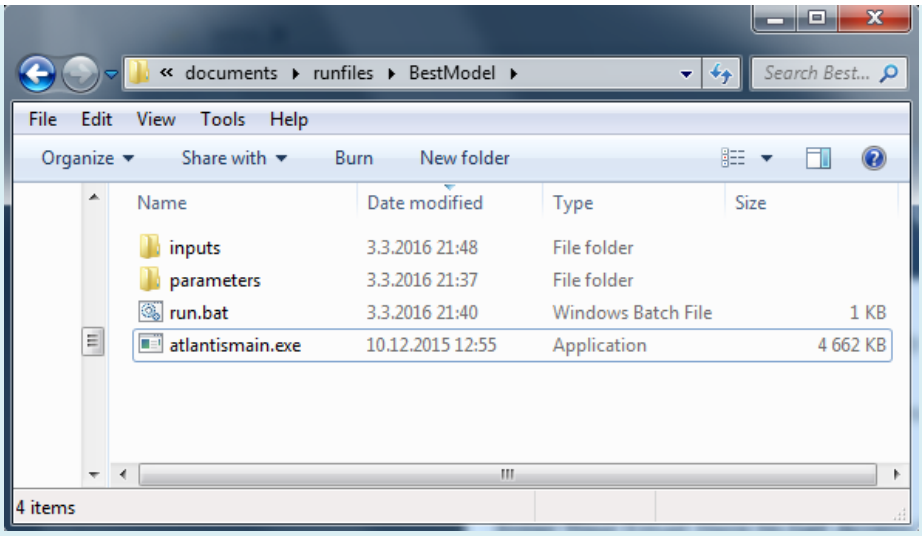
Good practice tip 1

A good practice Atlantis run directory is uncluttered and contains separate folders for:

1) external forcing files or **inputs**. These files are referred to in *force.prm* file and will rarely need to be modified after the parameterisation of the model. They are usually placed in the 'inputs' directory

2) other **parameter** files. These parameters will be listed in the *.bat* file used to initialise simulations. Depending on the research question, you will probably modify at least some of the biological, harvest or socio-economic parameters in each of the simulations. Although it is not obligatory we suggest that you keep all your parameters in a separate 'parameters' directory.

It is a good idea to give parameter files meaningful names. For example, it would be hard to remember what *A.prm* refers to, whereas *Biol_noclimate_change.prm* is a lot more informative. However, there is a limit to the length of a filename Atlantis will read-in so if your model won't run because it can't find a file and that file does exist but has a long name try shortening the name and see if that helps.



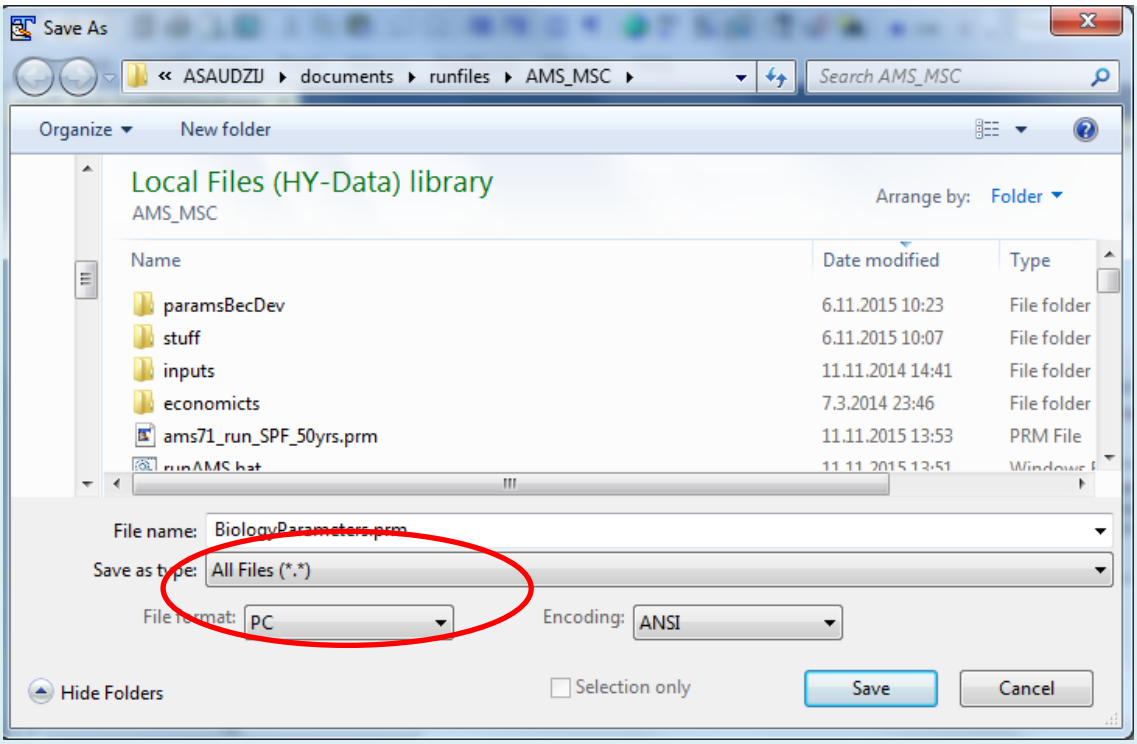
2.2. Input files

Atlantis input files come in six file formats:
1) BGM file defining model geometry (see chapter 3)
2) NC initial conditions and forcing files (see chapter 2.8)
3) CSV files defining biological and fisheries groups (see chapter 6)
4) PRM files listing flags to activate optional calculations and parameters values (described in detail in subsequent chapters)
5) TS files defining time series of external forcing drivers (see chapter 8)
6) BAT file used to initiate the run (see chapter 2.5)

All files except for NC files can be opened and edited using any basic text editing software (such as TextPad or NotePad). Until recently this was the usual way to setup and modify model parameters. However, some PRM files are thousands of lines long, which means that hand editing can be tedious and a source of error. New R-based tools are currently under development to help users visualise and modify parameter files in a more streamlined way (see below)

NOTE!

PC, Mac and Unix computers use different syntax to indicate line end. This means that should you receive files from someone using a different operating system, the PRM files will have to be converted to match your local operating system. In Linux this can be done using the 'flip' software. If you only have a few files this can be done by simply opening the parameter files in a text editor (e.g. TextPad in Windows or TextWrangler in Mac) and saving them in a format suitable for your operating system (see menu "File Format"). More info on line terminators can be found on the wiki here or by searching in your favourite search engine.



2.3. Overview of Atlantis submodels and the *main()* routine

Detailed descriptions of Atlantis submodels and specific input files are provided in subsequent chapters. For a start you need to understand which of the files are required to set up a simple simulations and what they mean (Figure 2). The figure below gives a schematic of the Atlantis submodels overlaid one over another. At the very basis there is the model geometry, Physics and Biogeochemical (Biology and Ecology) submodels. The other, optional, submodels use their outputs to simulate the human domain.

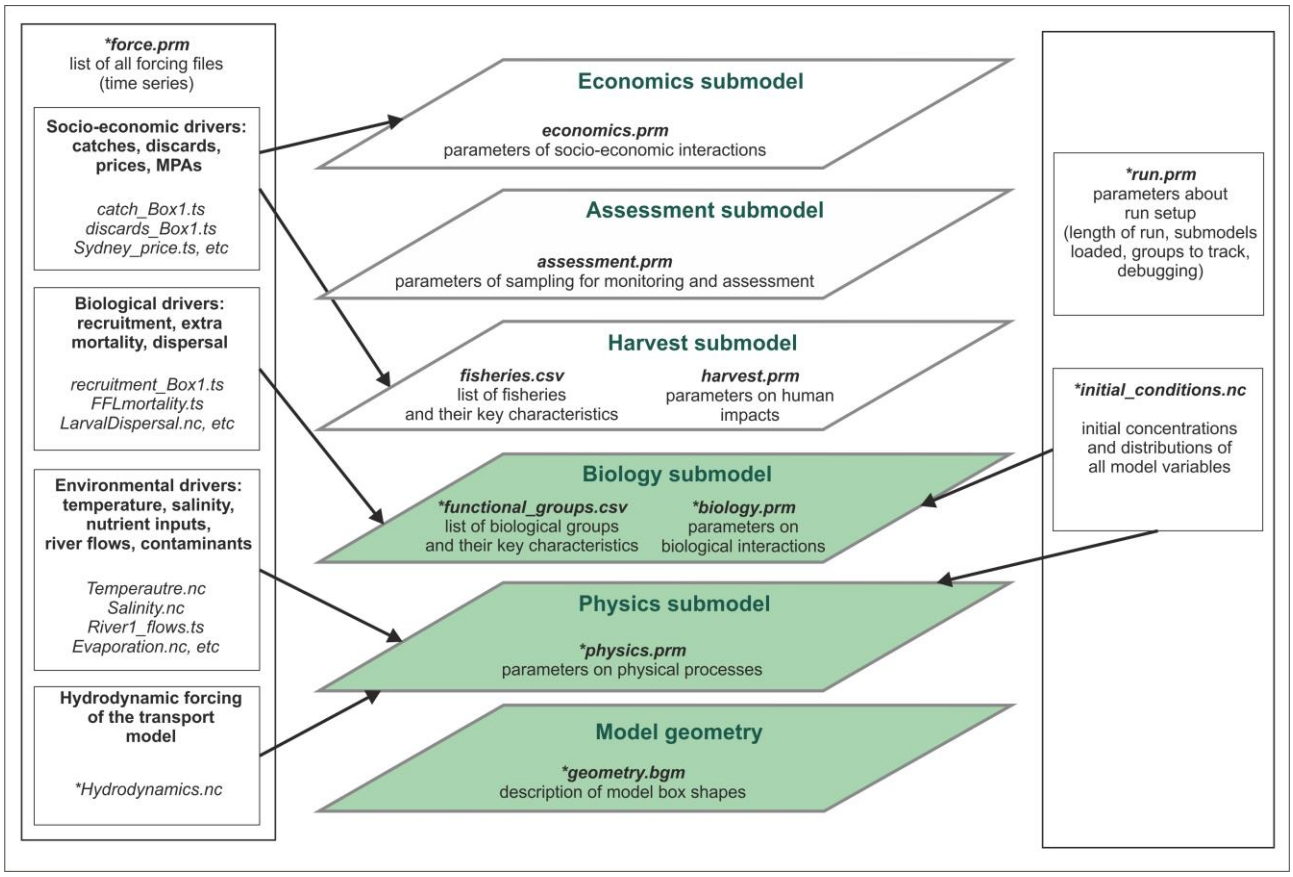


Figure 2. **Schematic representation of the Atlantis model structure and input files**. **Submodels coloured in green are obligatory. Files marked with * are required, others are optional.**

The 3D structure of your model domain, or **Model Geometry**, is described in the BGM file. The model geometry is used to simulate all processes in the five submodels.

Physics and **Biology** submodels are required (although you can switch some parts off for debugging purposes), whereas **Harvest**, **Assessment** and **Economics** submodels are optional. Each submodel has a specific PRM file describing parameters used in it. Biology and Harvest submodels also require a list of the relevant groups, such as biological functional groups or list of fisheries, in the CSV format.

Initial conditions are the spatially explicit starting values for all modelled state variables (e.g. spatial distribution of numbers of age 1 cod at the start of your simulations). The initial conditions are specified in the initial condition NC file**.**

In addition, all simulations require two supporting files *force.prm* and *run.prm* (see below for a brief description of all parameter files).

The *main()* routine is in the **atlantismain.c** file in the **atlantismain** library. It calls in a set order all other routines that initialise and run the requested submodels and routines and write output files. Each submodel and their main routines will be described in detail below, but a general overview is given in Fig. 3.

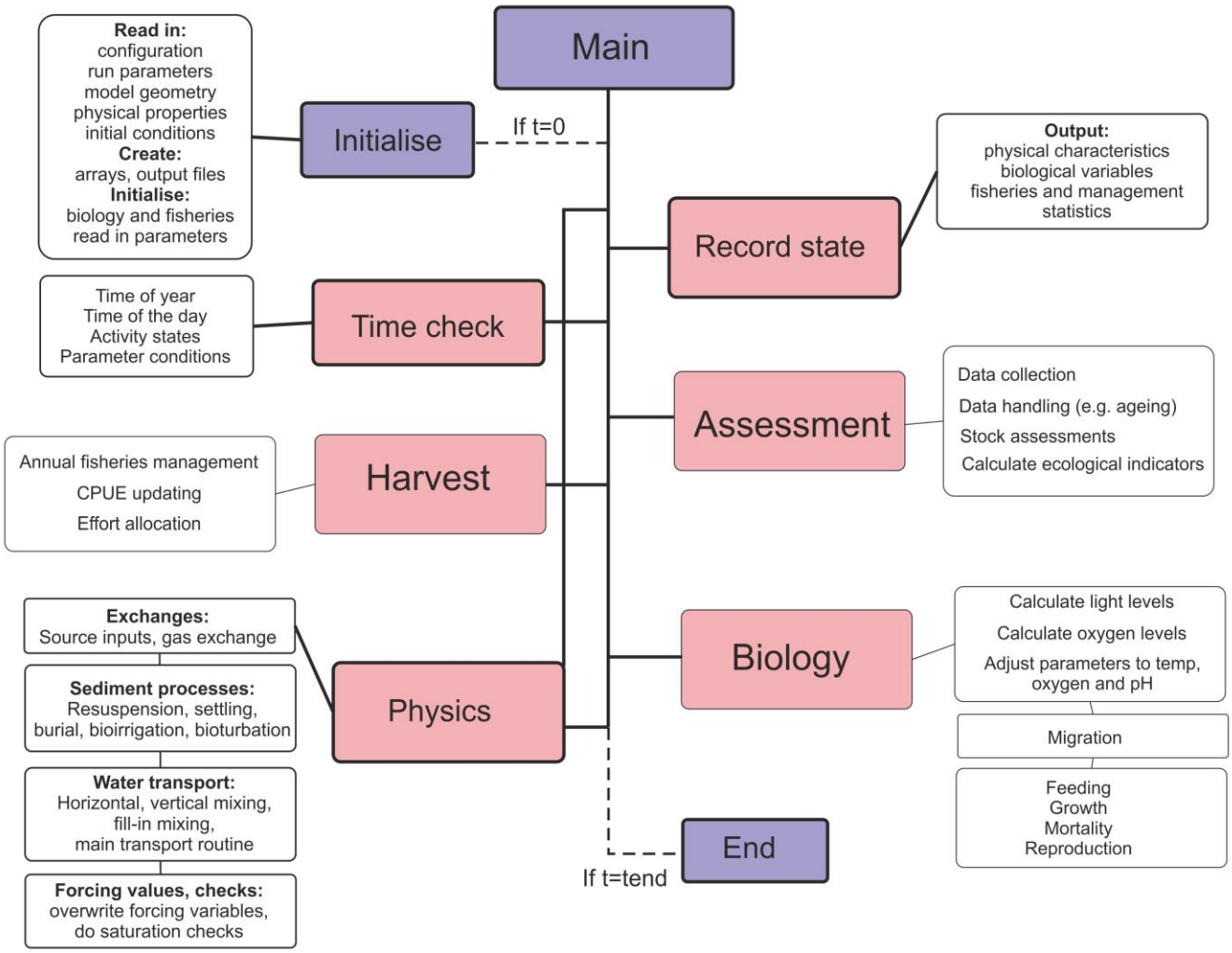


Figure 3. Brief overview of the Atlantis *main()* routine that calls other routines implemented in various submodels.

2.4. Delving into the code: finding the files and routines that describe the processes you are interested in

This chapter aims to give a very brief introduction into the Atlantis code. You do not need to use the code to run simulations. However, if you can navigate around the code and find routines that use the parameters and execute the processes you are interested in, it will most helpful and informative. In the end, this is the best way to understand what exactly the model does.

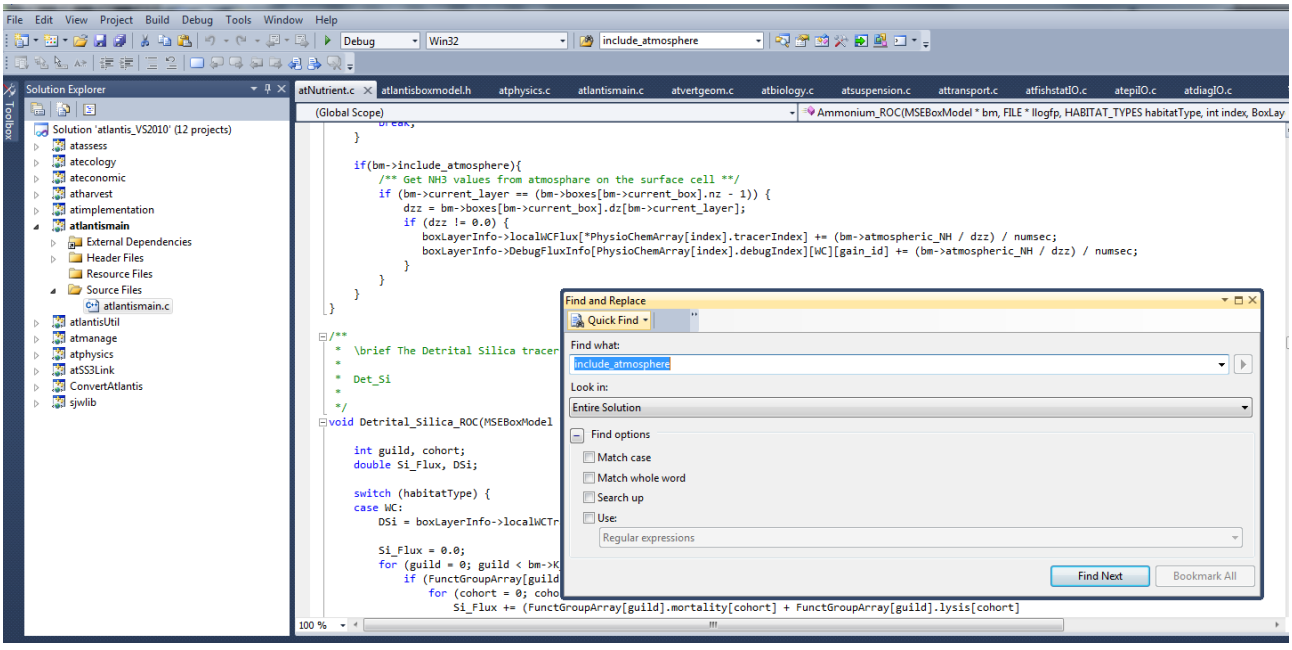
Of course, reading the code will require at least some programming skills. Check out a number of C tutorials available online for basic understanding about programming and C.

Two approaches might be useful in starting to navigate through the code.

The first is the **Search** tool. Each parameter from the parameter files is used somewhere in the code. In Visual Studio (or GCC, or another package) go to Edit->Find and Replace->Quick Find, or simply click Ctrl+F. This will bring up the Search menu where you type in the search string, such as the name of parameter.

In the example below, we search for the include_atmosphere parameter, used in the *physics.prm* file. Make sure you search in the "Entire Solution" if you want to browse through the whole code. You can either scroll through the entire list of search results and click directly on the line of interest (if you have some familiarity with the code) or you can click "Find Next" (if you go this route you may need to click a few times to finally get to one of the routines that uses the parameter). For example for include_atmosphere the relevant routine is called *Amonium_ROC()*, which is indicated on the top right drop-down menu in Visual Studio, and it is located in the file **atNutrient.c**

Here we can see that if include_atmosphere flag is on (set to 1) the code will transfer NH3 values from the atmosphere into the surface cell.

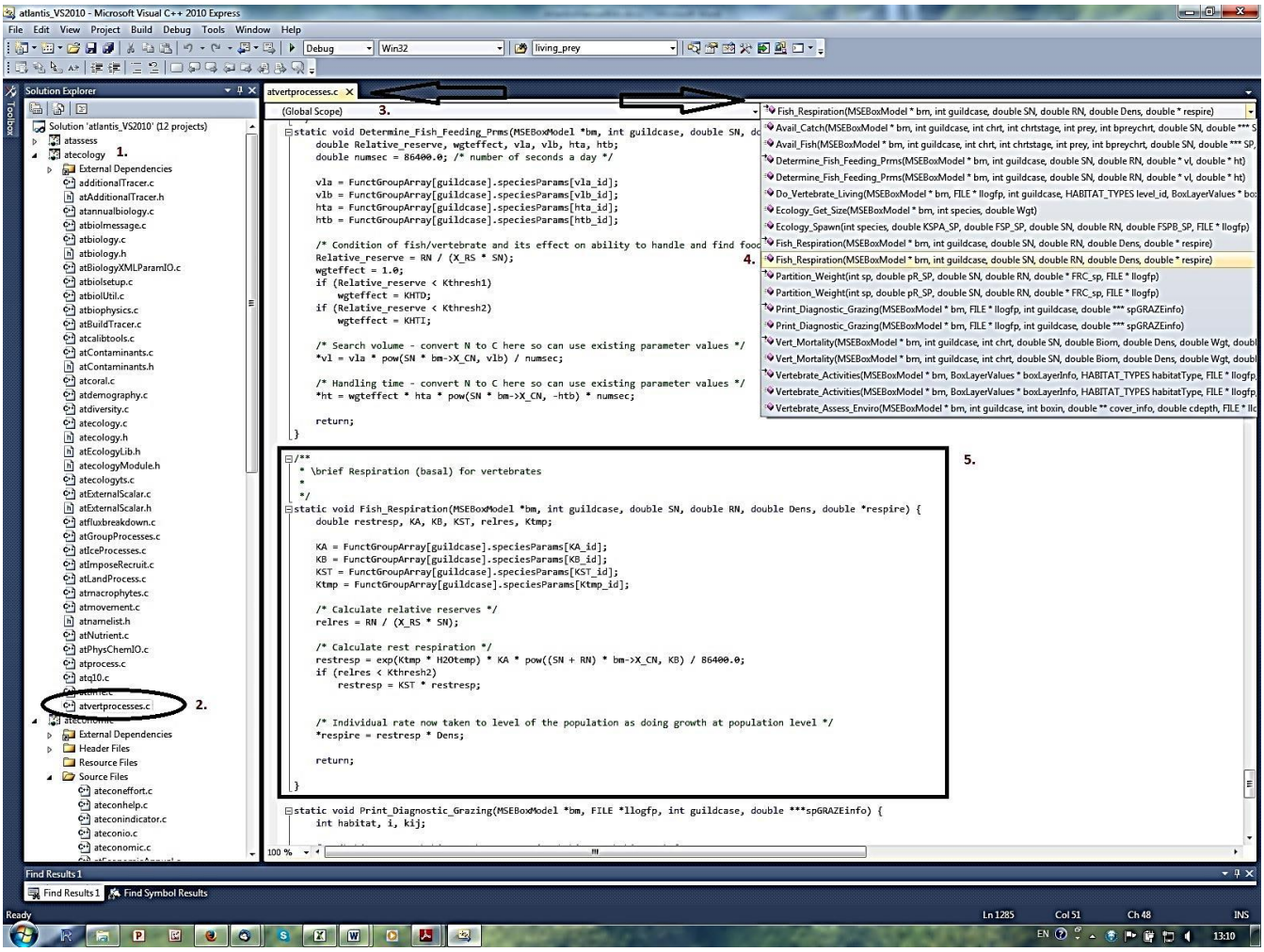


The second approach is to simply browse the libraries, files and routines you might be interested in. Of course, to do this you will need to know the names of routines. This can be done by searching for relevant parameters as shown above or by looking in the drop down list of routine names (most GUIs for interrogating code have such a feature).

As an example we will look at the formulation of the respiration processes in fully age structured groups.

Once you open your solution file in Visual Studio C++, expand the submodel **atecology** (**1.**) and double click the **atvertprocesses.c** file (**2.**). The file will be opened (**3.**) and on the right side you have a drop-down menu to browse through the routines defined in the file (**4.**). By clicking on the *Fish_Respiration()*routine you will find the description of parameters and equations used to calculate it Note that the first reference to *Fish_Respiration()* in the drop-down menu has a small arrow next to it. By clicking on this option you will find where in the code the routine is called.

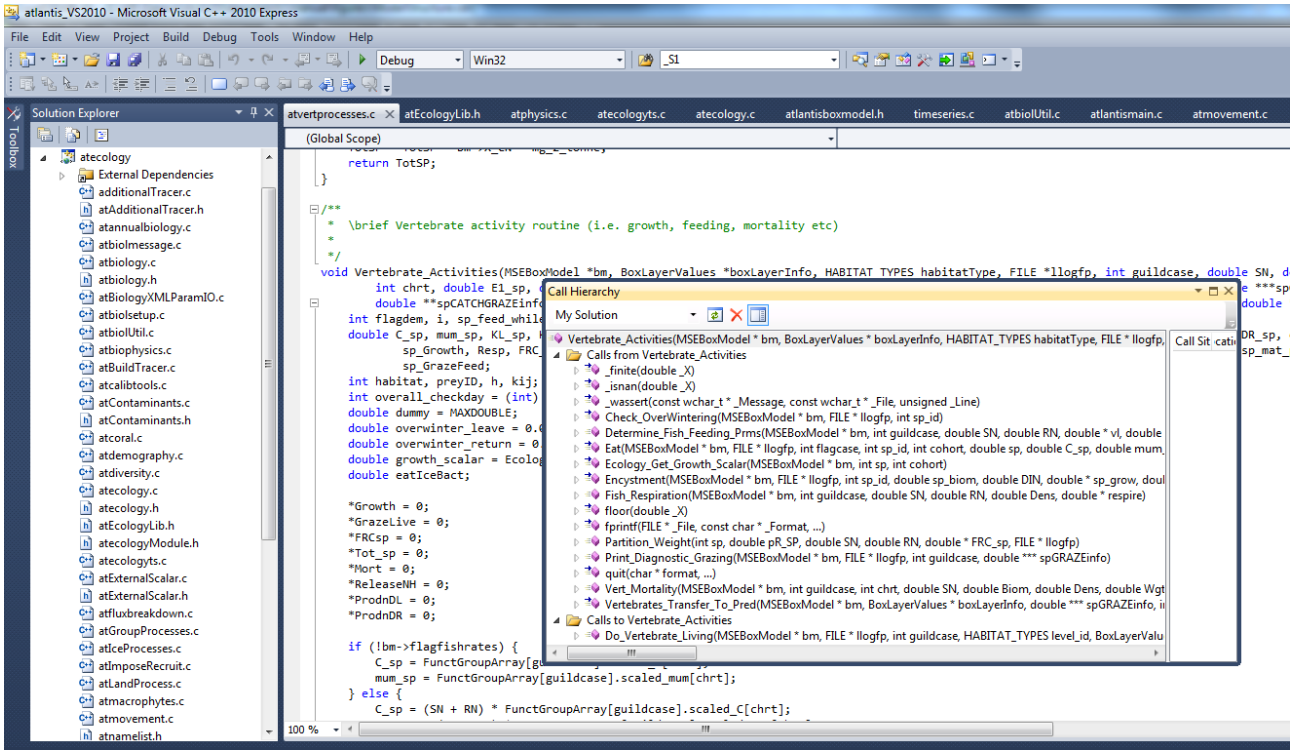
By exploring the code for the routine (**5.**) you can see that the routine needs variables calculated or defined elsewhere in the code, such as *guildcase*, *SN*, *RN*, *Dens* and **respire*, these variables are described within the parentheses *( )* right after the routine name. The routine also uses six parameters – *KA_id*, *KB_id*, *KST_id*, *Ktmp_id, Kthresh2* and *X_CN*. You can find them in the biological parameters file by replacing the '*id*' with your functional group abbreviation (e.g. *KA_FCD)*, as defined in the functional group .csv file. Sometimes the *_id* represents larger categories, such as _fish or _shark, as is the case for *KST_fish*, but you should be able to work it out by searching for the parameter name.



Now you only have to look through the equations and explore different parameter values in R or Excel, or on a piece of paper, and assess the shape of the response curve to different parameter values.

Another useful tool in exploring the code is the **Call Hierarchy** tool. It allows you to see the routines that call or are called from the routine of your interest. In Visual Studio this can be called by clicking

the right mouse button while holding the pointer over the routine name and selecting “View Call Hierarchy” option. In the example below, we can see that routine *Vertebrate_Activities()* is called from *Do_Vertebrate_Living()* routine, and that many other routines are called from *Vertebrate_Activities()*



2.5. How to setup and start simulations

2.5.1. Crash introduction into Command line interfaces

Atlantis is an “old style” program that runs from a command line. This is done using "Command Prompt" on Windows or "Terminal" on a Mac or Linux. Please familiarize yourself with the command line interface and how to navigate among folders. For most uses you only need to know a few commands and can pick up the required skills in a few minutes. There are plenty of online tutorials to quickly learn about the Command line interface for your operating system. Here are a few commands just to get you going (in fact you do not really need much more than that).

What to type:	**What this does:**
cd\	go to the root directory of the drive you are currently in
cd..	go one directory level up
cd C:\AtlantisRuns	go to the specified directory
cd C:\A + **TAB** key (type letter A and keep pressing TAB key)	scroll through all folders within **C:** that start with A (case insensitive). This way you don’t need to type long folder names (and make spelling mistakes)

Press arrows up and down	go to the previously typed commands
C:	Go to drive C: from any other drive you are in
Note: In Mac and Linux use / instead of \	

2.5.2. Setting up the BAT file

At this stage you already have 1) compiled the code on your machine, 2) created a run directory for your simulations (C:\AtlantisRuns\MyModel), 3) copied all input and parameter files into the run directory, 4) copied the Atlantis executable (atlantismain.exe) from C:\...\atlantismain\Debug into your run directory.

The next step is to create a BAT file such as *run.bat* (or any name you want, e.g. *NoFishing.bat*) and then call it from the command prompt to initialise the simulations. **The BAT file must be in the same directory as the Atlantis executable.**

The easiest way to setup and modify the .bat file is in a text editor.

The BAT file initialising Atlantis simulation has one long command line calling the Atlantis executable and giving the names of the parameter files required for the run. For example in windows the command would look like:

atlantismain.exe -i MySea_inital.nc 0 -o simulationOutput1.nc -r MySea_run.prm -p MySea_physics.prm -f MySea_force.prm -b MySea_biol.prm -s MyBiologyGroups.csv *-h MySea_harvest.prm -q MyFisheries.csv –a MySea_assess.prm –e MySea_econ.prm -d OutputFolder1 -t C:\AtlantisRuns\parameters*

Required parameters and files are shown **in bold**, optional parameters are in regular font.

You will still have to make sure that correct paths of forcing files (for hydrodynamics, salinity, temperature and others) are given in the *force.prm* file. In the *force.prm* file the path of the inputs/ folder name often has to given as ../inputs/ which tells the program that the inputs/ folder is in the same higher level directory as the *force.prm* file.

If the *–t <name of folder containing parameter files>* is omitted then Atlantis will expect to find the parameter files in the same directory as the batch file used to run the model. If the Atlantis executable is formally installed after it is compiled (e.g. using sudo make install in Linux or adding it to the Windows recognised path names, see the wiki) then you do not need to copy the executable file to the run directory and you only need to type the name of your Atlantis executable (e.g. *atlantismain* in the example) rather than name.exe in the command line.

Names of the parameter files must be given after an appropriate one letter argument which indicates what the following filename represents. For example, ***-i*** tells the executable that the following filename is the input file that details initial conditions. The actual names of the parameter files are flexible, as

long as the file extensions are correct. For example, the initial conditions file could be named as initial_conditions.nc, start_of_the_run.nc, start_of_the_fun.nc, a.nc or fred.nc. It does not matter so long as it is an NC file. It is **only** the one letter flag command (e.g. -i, -o, -p and others) preceding the filename that tells Atlantis which kind of file to expect.

For example, your BAT file could look like the following and it will run perfectly well:

BestModellingTool.exe -i BigBang.nc 0 -o TheLastSimulationEver.nc -r HowToRun.prm -p CleverPhysics.prm -f AllThoseInputs.prm -b EatEachOther.prm -s Critters.csv *-h HowToCatchFish.prm -q Fishers.csv –a TryToAssess.prm –e MBAdomain.prm -d PutEverythingHere -t C:\AtlantisRuns\parameters*

Don't forget to add number 0 after the name of the initial conditions file. This requirement is a legacy of the model development, and Atlantis will not run without it.

By default all simulation outputs will be placed in the same folder as the BAT file, but you can create an optional output folder (*-d OutputFolderName*). Specifying output folders is really important when running batch simulations (see below).

2.5.3. Parameter files called in the BAT command line

This section gives a brief description of the parameter files. They will be described in further details in subsequent chapters

-i initial conditions of your model run, including physical water column properties, seabed properties and biomasses and size per group per box

-o provides a name for the output files. Please make this an nc file (e.g. *simulationOutput1.nc*). Atlantis will create a lot of different outputs and will append titles of different output files to your specified name using the nc filename as the core of those names (e.g. *simulationOutput1BiomInd.txt*, *simulationOuput1Catches.txt*).

-r parameters defining the run setup, such as which modules to turn on (fishery, socio-economics), time step (12 or 24hr), run and stop times, frequency of outputs, flags for debugging and verbosity, details of notes to store in log.txt

-p physics parameters, such as resuspension, point-source scaling, eddy strengths and others. These parameters are not typically changed, although you may want to modify eddy strengths or turn off resuspension.

-f forcing file provides pathways to all external (i.e. not simulated within the model) input files, such as hydrodynamics, salinity, temperature, nutrient input sources from river flows or upwelling, climate time series (precipitation, irradiance), historical catch, fuel prices, GDP and complex spatial zoning through time (the list of allowed external forcing files is given the chapter on *force.prm* below). These files are usually stored in your "inputs" folder.

-b ecological parameters, including feeding preferences, distributions, migration, mortality, reproduction and others

-s list of functional biological groups and their key characteristics, such as number of age groups

-h parameters on fishing fleets, target species, catch history, discards, changes in selectivity over time, etc. Note that if you use external forcing of catch or fishing mortality, those forcing files must be specified in the *force.prm* file and not in the *harvest.prm* file

-a sample design, sampling error structures and basic assessment model parameters

-e socio-economics parameters for fleet dynamics, market model, trading model, and black-book based effort allocation model

-d an optional destination folder. If provided, a new folder with this name will be created during the run and all outputs placed in this folder. Using this option is highly recommended

- t an optional, but recommended folder, for storing input parameters (see Good Practice Tip 1). If this folder is not provided, all parameters (not inputs) must be in the same directory as the Atlantis bat file, if Atlantis has been formally installed to the command line, or in the same directory as the executable if you are simply copying the exe file to the run directory.

Note, the **forcing files** (such as hydrodynamics, temperature, catch, recruitment and other forcing files) are not included in the *BAT* command line, because they are all listed in the *force.prm* file

Good practice tip 2

It is a good idea to give informative names to the parameter files. This way you can remember changes made in them. For example, *Baltic_biol_CodMort05.prm* could denote a biological parameter file with cod mortality set to 0.5. Also make sure you rename the parameter file if you modify it. This will help you keep track of which parameter combination was used for each simulation.

It is **strongly recommended that you keep a good record of all parameter modifications** and outcomes they had on model simulations. We cannot stress this strongly enough! Almost every modeller has experienced difficult moments when trying to trace back what exactly was done at different steps of model development or scenario analyses.

2.5.4. Setting up batch simulations

To set up **batch simulations**, simply add more command lines in your BAT file:

*atlantismain.exe -i MySea_inital.nc 0 -**o NoFishing.nc** -r MySea_run.prm -p MySea_physics.prm -f MySea_force.prm -b MySea_biol.prm -s MyBiologyGroups.csv -**h MySea_harvestNoFishing.prm** -q*

MyFisheries.csv -a MySea_assess.prm -e MySea_econ.prm ***-d OutputNoFishing*** *-t C:\AtlantisRuns\parameters*

atlantismain.exe -i MySea_inital.nc 0 ***-o MedFishing.nc*** *-r MySea_run.prm -p MySea_physics.prm -f MySea_force.prm -b MySea_biol.prm -s MyBiologyGroups.csv* ***-h MySea_harvestMedFishing.prm*** *-q MyFisheries.csv –a MySea_assess.prm -e MySea_econ.prm* ***-d OutputMedFishing*** *-t C:\AtlantisRuns\parameters*

atlantismain.exe -i MySea_inital.nc 0 ***-o HighFishing.nc*** *-r MySea_run.prm -p MySea_physics.prm -f MySea_force.prm -b MySea_biol.prm -s MyBiologyGroups.csv* ***-h MySea_harvesHighFishing.prm*** *-q MyFisheries.csv -a MySea_assess.prm -e MySea_econ.prm* ***-d OutputHighFishing*** *-t C:\AtlantisRuns\parameters*

Batch simulations would normally have some differences between runs (**shown in bold** in the example above). Make sure you place each output in a different folder (-d) and don't overwrite it! Also make sure you carefully check your command lines to ensure you call the correct parameter files. It is really disappointing to wait for a few days for your batch simulation outputs just to realise that the BAT file had wrong combinations of parameter files in it!

NOTE!

If you run a lot of simulations it is good idea to keep your BAT files handy, so you can easily trace the parameter combinations used. The LOG file produced for each simulation includes the command line of the BAT file. However, some users find it convenient to add new commands to the BAT file and comment (deactivate) the old ones out with # sign.

#atlantismain.exe -i MySea_inital.nc 0 -o NoFishing.nc -r MySea_run.prm -p MySea_physics.prm -f MySea_force.prm -b MySea_biol.prm -s MyBiologyGroups.csv -h MySea_harvestNoFishing.prm -q MyFisheries.csv -a MySea_assess.prm -e MySea_econ.prm -d OutputNoFishing -t C:\AtlantisRuns\parameters

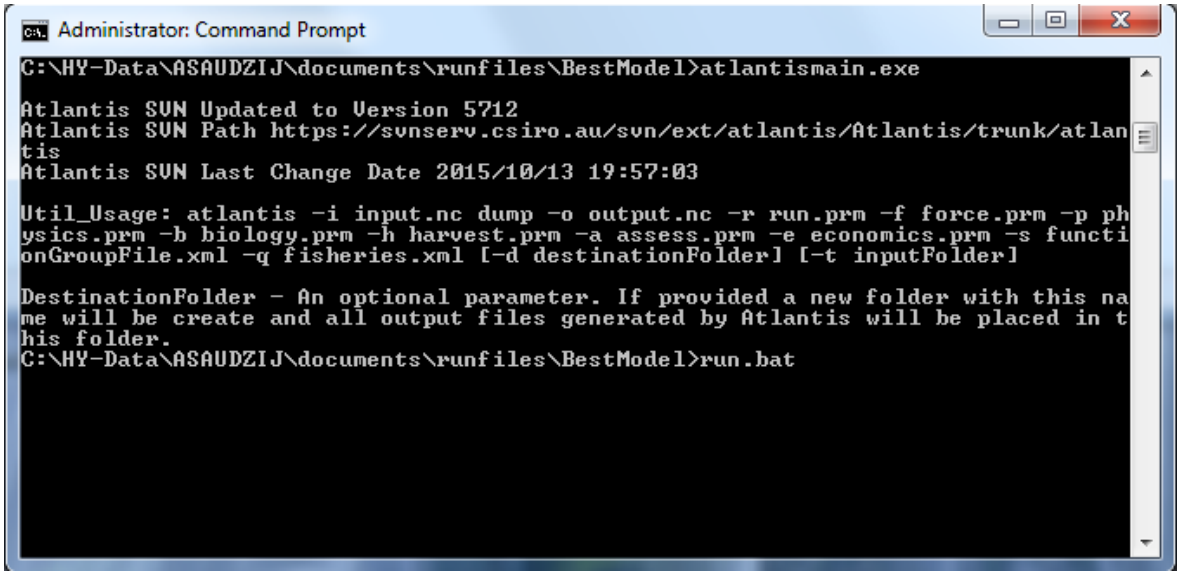
atlantismain.exe -i MySea_inital.nc 0 ***-o MedFishing.nc*** *-r MySea_run.prm -p MySea_physics.prm -f MySea_force.prm -b MySea_biol.prm -s MyBiologyGroups.csv* ***-h MySea_harvestMedFishing.prm*** *-q MyFisheries.csv –a MySea_assess.prm -e MySea_econ.prm* ***-d OutputMedFishing*** *-t C:\AtlantisRuns\parameters*

In this example, Atlantis will skip the first line and will only execute the second one (where the fishing parameters, harvest and output files are changed). This way you can easily keep a record of parameter combinations used in earlier simulations. Of course, if you run dozens of simulations at some point this BAT file will get too long and cluttered and you will need a different bookkeeping system (but make sure you have some system to trace each parameter combination in each simulation!)

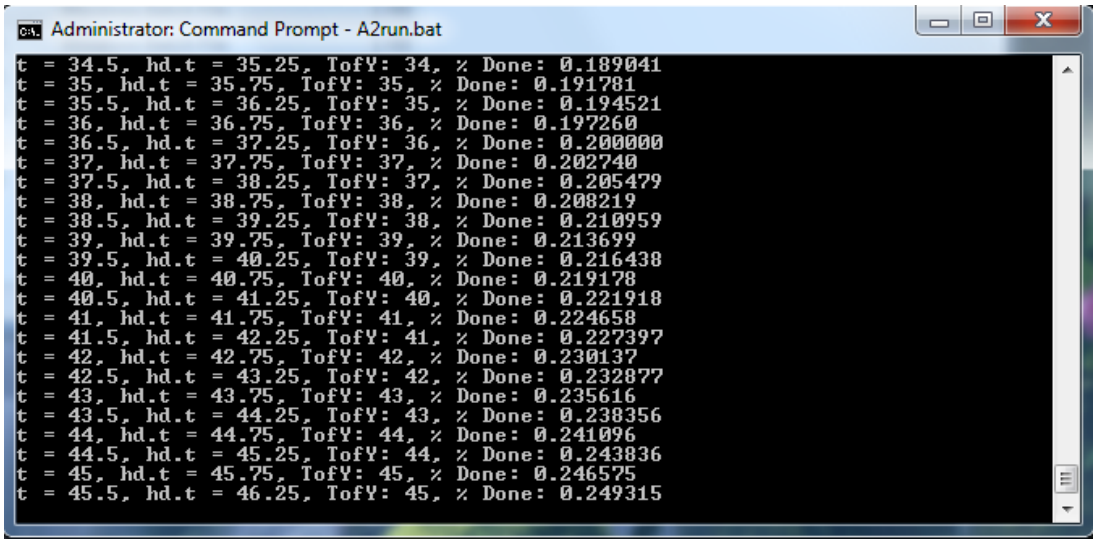
Note, that the line here refers to the whole command string, separated by line spaces. One command line is wrapped into four lines in the example above, but it does not contain any line breaks.

2.5.5. Initialising simulations

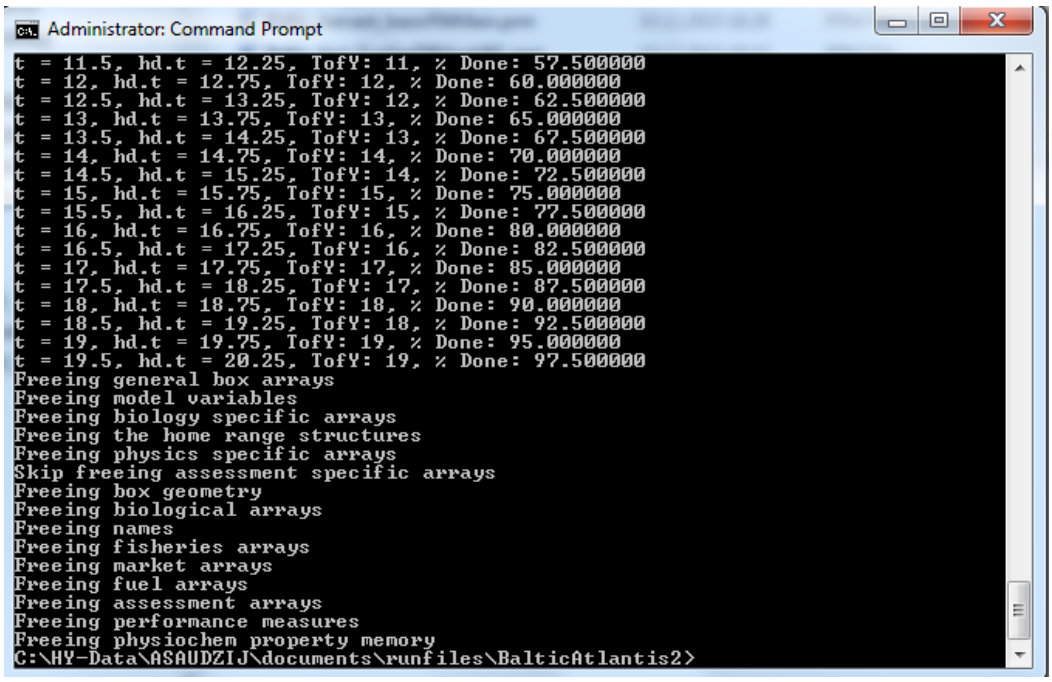
Once you have all the input and parameter files, EXE and BAT files ready, open the Command Prompt (or Terminal) window, navigate to your model run directory that has the BAT files in it (and the EXE if you have not formally installed Atlantis to the command line paths), type the name of your BAT file and press ENTER.



Atlantis should start running now! If all is going well, you should be getting short daily messages, looking like this (45.5 days are completed below):



When the run is complete, Atlantis will finalise writing all the outputs, free arrays and close (the simulation below was run for 20 days):



If, for some reason you need to stop your run before it is finished, go to the output folder and delete the file called "***delete_to_halt_run***". This way Atlantis will write all the outputs to date and free memory before stopping the simulation. If you stop Atlantis any other way the files will not be completed properly and will most likely be uselessly incomplete.

NOTE!

During the simulations Atlantis runs many outputs into its internal memory. This means that even though all the output files are created at the beginning of the simulation, they may not contain much information until the runs are actually finished. So, do not get disturbed if you want to peek into some output file during the simulation and find it empty, even if the simulation has already been running for a long time.

Don't despair if the simulation does not start so smoothly. Error messages are an important part of this adventure! Check out the chapter on troubleshooting and the dedicated Atlantis wiki page

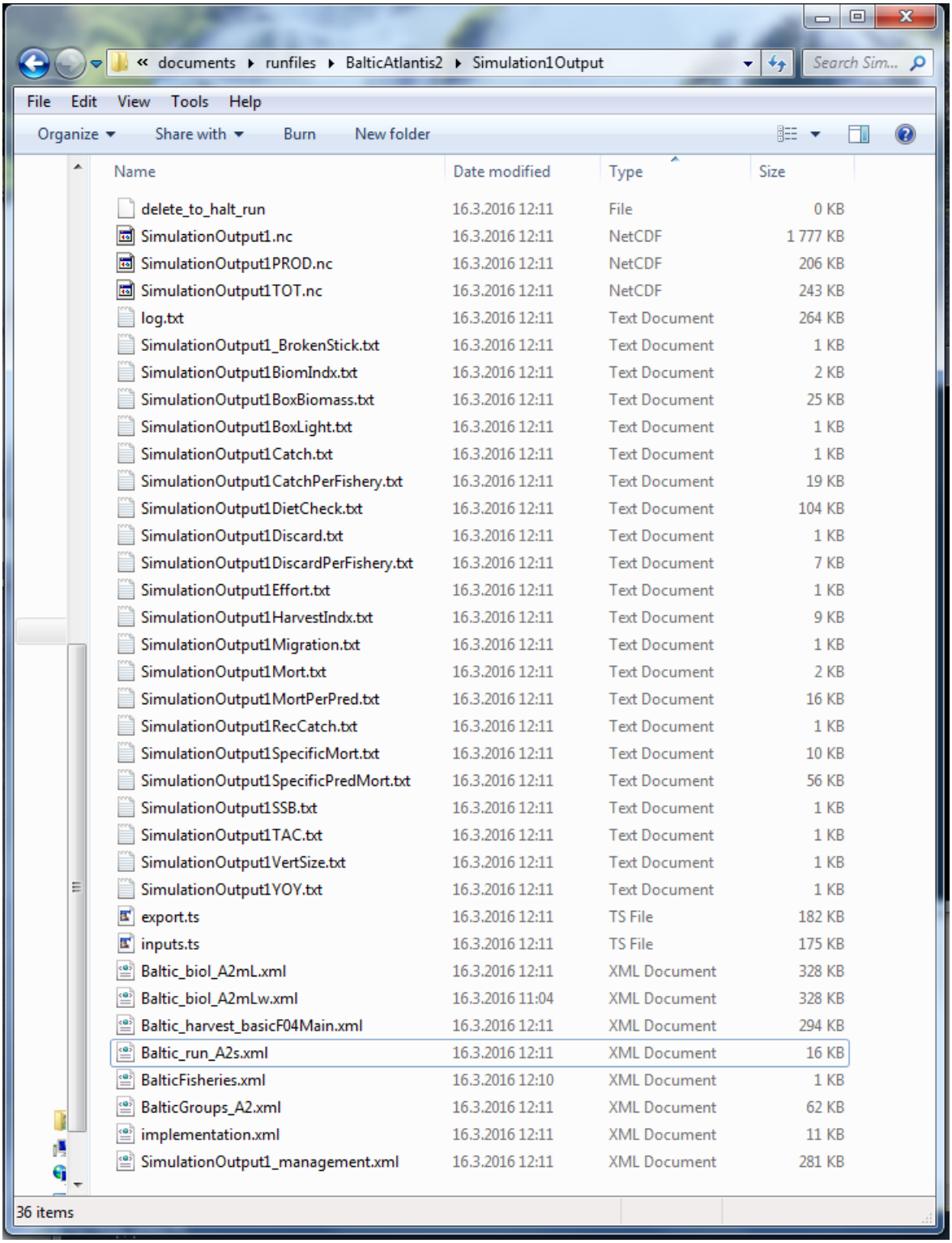
2.6. Simulation outputs

Once the simulation finishes Atlantis will generate **a lot of output files** in TXT and NC formats. The simulation output folder will also contain the *log.txt* file (with various details or messages generated during the run), as well as *exports.ts* and *inputs.ts*, which store information on flows into/out of the model domain, and a number of XML files, which are mostly just automatically generated files that are converted input files restructured to a format that Atlantis can more easily read in. Most users never touch the TS and XML output files. The XML files might be a useful means of storing the input parameters in a database but are otherwise of little ongoing use. The TS files however can be quite informative for showing how much material enters and exits the domain due to advection.

NOTE!

The *log.txt* file does not start with the name given in the ***–o*** command. This means that if you accidentally start another simulation with the same output folder name (or without giving the output folder), the *log.txt* file will be overwritten.

The list of output files in your output folder will look like this (it indeed has **a lot of** outputs!):



The table below lists the main output files. Some output files will be produced only if relevant flags are active (e.g. evolution, different management or fisheries options) and are not included in the table.

Table 3. Main Atlantis output files. The main output files of interest for beginner users are shown in grey boxes. The table first lists Biology related outputs and then Harvest submodel related outputs. Please note, that the list of output files are continuously updated, so check the wiki for any changes.

Name	Contents	Units	Comments

BiomIndx.txt	**Snapshots** of biomasses of all active species (set in the *functional_group.csv* file) across the entire model domain, excluding boundary boxes. Usually this is the first file to explore, but don't forget to look at the box-specific biomasses!	tonnes	Frequency of output controlled with tsumout in *run.prm*
BoxBiomass.txt	**Snapshots** of biomasses of all active species for each of the model domain boxes, excluding boundary boxes	tonnes	Frequency of output controlled with tsumout in *run.prm*
AgeBiomass.txt	**Snapshots** of biomasses of all active species per age group across the entire model domain	tonnes	This output is given if flag_age_output is set to 1 in *run.prm*
DietCheck.txt	Proportional make up of a diet of each age class of each functional group. This is the cumulative proportion since the last output	propor-tion	The total sum for each age group is 1 for all prey items
DetailedDietCheck.txt	As above but proportions are given for each cell (box and layer) of each age group of each functional group	propor-tion	Very useful for understanding model dynamics. Check out R codes for querying these outputs (chapter 2.9)
Mort.txt	Annual predation (-M) and fishing (-F) mortality of each functional group. This file is currently only useful for looking at **relative M vs F values** for a species, as it does not give accurate mortalities	per year	Note, that if a species is set as isImpacted in the *functional_group.csv* file, it will have some F value even if it is not explicitly targeted by fishing.
SpecificMort.txt	Three values of instantaneous mortality for each functional group, age group and stock. M2 is predation mortality, M1 other natural mortality (starvation, linear mortality), F is fishing mortality. At present the M and F values for this output are written **before** Atlantis checks that consumption and fishing are not trying to take more than is available in each cell. This means that **M or F in the file can sometimes exceed 1**. This will be corrected in a future release of Atlantis as the code is restructured.	per year	At this stage the file is useful only as an **indication of relative morality** for qualitative mortality assessment (high/low)

SpecificPredMort.txt	**Cumulative** predation mortality of each stock and age group of each functional group (in rows) as imposed by each predator (in columns) over the course of a year. As with the SpecificMort.txt above, **values can exceed 1**. This will be fixed in the future	per year	At this stage the file is useful only as an **indication of relative morality** for qualitative mortality assessment (high/low)
YOY.txt	Biomass of recruits per year (which is the same as per spawning event, because a group only spawns once per year) summed over the total model domain. **Snapshot** at the time of spawning, regardless of when the output is written out.	tonnes	
Migration.txt	Total and migrating biomass (and proportion migrating) of migrating groups per adults and juveniles.	tonnes	Data is given for groups that are tagged as migratory in *biology.prm* file
VertSize.txt	Numbers, average sizes and biomass of each migrating age structured group in the model domain at the time of the output. The output does not include sizes or numbers of individuals outside the model at the time of the output.		Data is given for groups that are tagged as migratory in *biology.prm* file
.nc	3D (time, box, layer) **snapshots** of the tracer values at the time of the output	mg/m^3 or numb	Frequency of output controlled with toutinc in *run.prm* file
TOT.nc	2D (time, box) **snapshots** of tracer values per box (summed over layers)	tonnes/ km^2 or numb	Frequency of output controlled with toutinc in *run.prm* file
PROD.nc	2D (time, box) **snapshots** of diagnostic tracers, such as production and grazing for biomass groups, growth and consumption for each age group for each fully age structured group, and a number of other diagnostic indices	mg/m^3/ day or mg/day /individ	Frequency of output controlled with toutinc in *run.prm* file
ANNAGEBIO.nc	Numbers in each **annual** age class of a species (**NOT** the model age group) per species. This can be useful when tracking annual changes in species that have multiple years per age group	numb/ box/layer	The output is written only when flag_age_output is set to 1 in *run.prm* file
BoxLight.txt	Proportion of sun hours per time step (how long during that time step there was light)		
Catch.txt	**Cumulative** total catch per functional group over all commercial fisheries	tonnes	

CatchPerFishery.txt	**Cumulative** total catch per functional group per fishery	tonnes	
Discard.txt	**Cumulative** total discards per functional group over all commercial fisheries	tonnes	
DiscardPerFishery.txt	**Cumulative** total discards per functional group per fishery	tonnes	
Effort.txt	**Cumulative** number of days each fishery spends fishing	days	Output given only if dynamic effort is turned on in *harvest.prm* file
HarvestIndx.txt	Fisheries and management indices, e.g. average size of catch, total landings, total discards, the number of changes in gear, bycatch of threatened or endangered species, habitat impacts, total effort expended, social acceptance, conflict between fisheries sectors, management access, management costs		Many indices only make sense when appropriate fisheries flags are turned on (see chapter on the Harvest submodel)
RecCatch.txt	**Cumulative** total catch per functional group over all recreational fisheries	tonnes	The fleet must be marked as recreational (isREC) in *fisheries.csv* file
SSB.txt	**Snapshot** of mature biomass at the time of spawning	tonnes	
TAC.txt	Total allowable catch per species per year. It can be fixed or dynamic dependent on the fishing scenarios	tonnes	Only written when fisheries use TAC for management, set in flagManage in *harvest.prm*
BrokenStick.txt	Indices related to broken stick harvest strategy (current F, target F, current biomass and resulting scalar to apply as calculated by the broken stick).		Only written when broken stick harvesting is turned on in *harvest.prm*
CATCH.nc	**2D** (time, box) file of **cumulative** values of catch and discard per species per fishery in numbers and tonnes per box: catch per species per age group (numbers), discards per species per age group (numbers), catch per species per fishery (tonnes), discards per species per fishery (tonnes)	tonnes or number	Frequency of output controlled with toutfinc in *run.prm* file
CATCHTOT.nc	**2D** (time, box) file of **cumulative** values of catch, discards and recreational catch per each box	tonnes	Frequency of output controlled with toutfinc parameter in *run.prm* file

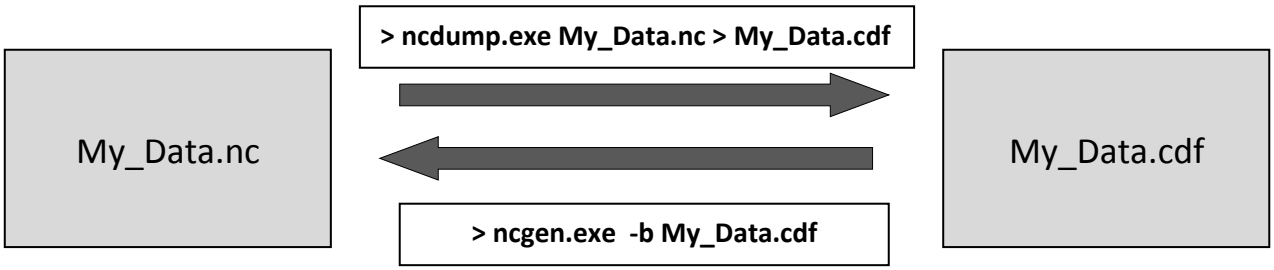
ANNAGECATCH.nc	Numbers in each **annual** age class of a species (**NOT** the model age group) per species in the catch and discards. This can be useful when tracking annual changes in number for species that have multiple years per cohort	numb	The output is written only when flag_age_output is set to 1 in *run.prm* file
inputs.ts	amount of tracers coming from all boundary boxes and time series into all dynamic boxes		Frequency of output controlled with inputs_tout in *run.prm* file
exports.ts	amount of tracers coming from all dynamic boxes and time series into all boundary boxes		Frequency of output controlled with inputs_tout in *run.prm* file

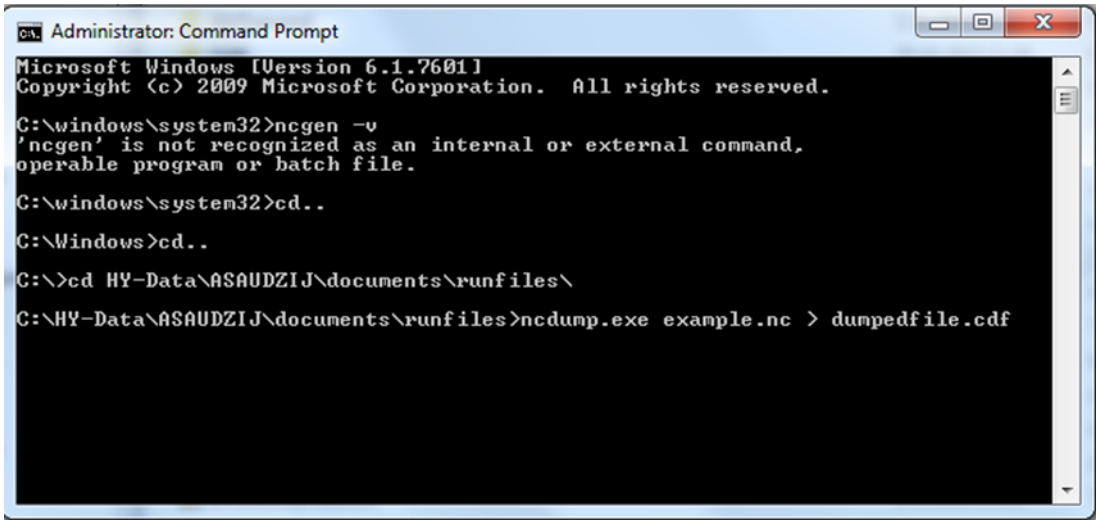
2.7. Introduction to NetCDF files

NetCDF (Network Common Data Form) is a set of software libraries and data formats that allow for the creation, access, and sharing of array-oriented scientific data irrespective of type of computer hardware. They are typically used by oceanographers and atmospheric scientists and modellers to store spatial time series information. The NC file format is the computer-readable version, while the CDF format is its unpacked human-readable version that can be viewed and edited with a text editor (such as Notepad or Textpad). The key characteristics of these files are that they are platform-independent and "self-describing". This means that the file has a header which describes the layout of the rest of the file and array dimensions, followed by sections of data and variable attributes. There are many matlab, R and other tools for interacting with NC files.

The *initial_conditions.nc* file in Atlantis must list all variables to be tracked throughout the simulation, their initial values or concentrations in the spatial domain, and some of their key characteristics. It is important to get them right. The *initial_conditions.nc* file is huge and it is not advisable to edit it by hand (see below for tools available to set and modify it). However, it is useful to open and explore the file by converting the NC file to a CDF file.

Two small programs, **ncdump.exe** and **ncgen.exe,** can be used to convert NC files into CDF files (to create human readable and editable files) and from CDF to NC respectively (in case you modified the CDF file and need to convert it back into the computer-readable format). The two packages are included in the NetCDF libraries required for compiling Atlantis (see step 4 in the Chapter 2.1). To convert the files, open the Command Prompt, navigate into the folder that includes the NC or CDF files you want to convert and call the appropriate command with its parameters, as shown below. For example, if you want to convert NC into CDF, type: ncdump.exe somedata.nc > otherdata.cdf. Note, the name of the NC file created by the ncgen.exe, will be the same as the name given in the first line of the CDF file (so NOT necessarily the same name as typed in the command prompt). If you install NetCDF and update the paths as described on the wiki the Command Prompt will automatically locate ncdump.exe or ncgen.exe files, so you don't need to have the EXE files in the same folder as the file you want to convert. The names of the NC and CDF files don't have to be the same.





Once you converted the NC file into CDF, you can open it using a text editor. Here is an example of an output.nc file (comments after ## are added here for explanation):

```
netcdf example {                                    ## name of the file

## header section of the file – describes file contents and dimensions
dimensions:
        t = UNLIMITED ; // (111 currently)          ## number of time steps
        b = 29 ;                                    ## number of boxes
        z = 9 ;                                     ## number of layers
        icenz=2;                                    ## OPTIONAL: number of ice layers
                                                    ## the icenz attribute activates the  ice model

variables:                                          ## start listing variable descriptions
        double t(t) ;                               ## double precision 1D variable - time
        t:units = "seconds since 2005-01-01 00:00:00 +10" ;     ## start time of the simulation: time
stamp
        double volume(t, b, z) ;                    ## 3-dimens. variable: volume (t, b, z)
                volume:bmtype = "phys" ;            ## description of Physical variables
                volume:units = "m3" ;
                volume:long_name = "Volume" ;
        …
         double Det_Si(t, b, z) ;                     ## description of Tracer variables
                 Det_Si:bmtype = "tracer" ;
                 Det_Si:units = "mg Si m-3" ;
```

```
                Det_Si:long_name = "Detrital Silica" ;
                Det_Si:dtype = 0 ;
                Det_Si:sumtype = 1 ;
                Det_Si:inwc = 1 ;
                Det_Si:insed = 1 ;
                Det_Si:decay = 0. ;
                Det_Si:dissol = 0 ;
                Det_Si:partic = 1 ;
                 Det_Si:passive = 1 ;
                Det_Si:svel = -2.894e-005 ;
                Det_Si:xvel = 0. ;
                Det_Si:psize = 1.e-005 ;
                Det_Si:b_dens = 1000000000. ;
                Det_Si:i_conc = 200000000. ;
                Det_Si:f_conc = 200000000. ;
          …
          double Filter_Soft_N(t, b) ;                    ## description of Epibenthic variables
                Filter_Soft_N:bmtype = "epibenthos" ;
                Filter_Soft_N:units = "mg N m-2" ;
                Filter_Soft_N:long_name = "Soft substrate filter feeders Nitrogen" ;
                Filter_Soft_N:dtype = 0 ;
                Filter_Soft_N:sumtype = 1 ;


        …
// global attributes:                                   ##global attributes or variables
                :title = "Baltic" ;
                :geometry = "Baltic_aea_v2BGM" ;
                :parameters = "atlantismain2.exe -i init_Baltic.nc 0 -o BalticClimate1.nc -s
BalticClimate1TOT.nc -g BalticClimate1PROD.nc -c BalticClimate1TOTCATCH.nc -d
BalticClimate1CATCH.nc -r Baltic_run.prm -f Baltic_force2015.prm -b Baltic_biol_Climate.prm -h  -p
Baltic_physics.prm";          ##details on BAT and other outputs

                  :wcnz = 8 ;                          ## number of water column layers
                :sednz = 1 ;                           ## number of sediment layers

## data section of the file contains a data entry for each variable declared in the header section

data:                                                  ## start of the data for each
variable

t = 0, 7862400, 15724800, 23587200, 31449600, 39312000, 47174400,      ## time steps of the
simulation
55036800,  62899200, 70761600, 78624000, 86486400, 94348800, …

volume =                                               ## variable values per layer,
box, time
 0, 0, 0, 0, 1762842878.80397, 8664747499.65088, 2166186874.91272,
```

```
  2166186874.91272, 216618687.491272,
 0, 0, 0, 0, 0, 15949302157.943, 27451466709.024, 27451466709.024,
  2745146670.9024, ...
 ...                                                    ## continued until all variables
are listed

}
```

2.8. Introduction to Atlantis variables

There are three kinds of obligatory and two kinds of optional variables used in Atlantis. Variable types are defined by the bmtype parameter:

1) "phys" - physical properties of the model
2) "tracer" – three-dimensional dynamic variables (either in the water column or the sediments) of the model domain
3) "epibenthos" – two-dimensional dynamic benthic variables
4) "icetracer" – a variable that exists in ice layers (if the model explicitly includes ice)
5) "terrestrial" – tracers used to identify land variables (if the model explicitly includes land)

Each variable declaration includes the expected numerical precision. A few variables (e.g. numlayers) are marked as short, which means they are expecting integer input variables, but most variables are doubles, which indicates that they are a double precision floating point number format (a computer way saying this is a real number not simply an integer). The precision identifier is followed by the variable name and the expected dimensions. Variables can be one dimensional (time), two dimensional (time, box) or three dimensional (time, box, layer). It is important to make sure the data entered has this many dimensions (with no omissions) as Atlantis assumes the dimensions are fully filled when it loads the data.

Next, the file lists the characteristics of the variables. All variables have four general parameters: units, long_name, dtype, sumtype. Units and long_name are only needed for the user and interpretation of results; they are entered in inverted commas " " as they will be loaded as strings of text by Atlantis.

The parameters dtype and sumtype are flags whose value determines where the output of the variable value is to be written.

The dtype describes the information that will be written to the output files. It can have four values:

0: Standard snapshots, sent to the base *out.nc* output files and summarised in 2D in *outTOT.nc*
1: Fisheries total values, written to *fisheresTOT.nc* files when the fisheries submodel is active
2: Ecological diagnostics, sent to diagnostic *PROD.nc* files (production, consumption etc)
3: Detailed fisheries information, typically used for tracers in the dynamic fisheries, sent to base *fisheries_out.nc* file

The sumtype parameter can have two values:

0: indicates that data is written in the 3D format (per layer per box per time step) in the main *out.nc* file and given in detail
1: indicates that, in addition to the 3D output given in the main *out.nc* file, summary (e.g. total value per box integrated over all layers) is also written to the *outTOT.nc* file

The term "**tracer**" will be used often throughout the manual. Most of the variables belong to this category. They include temperature, salinity, oxygen, nutrients (NO3, NH3, organic nitrogen), bacteria, pelagic organisms and description of age structured groups. Most tracers, like bacteria_N, cod1_numbers are dynamic, which means they participate in the processes and change through time. The *initial_conditions.nc* file includes some important characteristics of these tracers (Table 4). Other tracers are in fact processes, such as Nitrification, Denitrification, Light adaptation, Sediment Porosity and others. They are initiated as tracers in order to create arrays for them, so Atlantis has a dedicated place to write the output for them.

Table 4. Parameters used to describe characteristics of Atlantis variables in the *initial_conditions.nc* file. These parameters are read in by the *readBMTracerInfo()* in **attracerIO.c**

Parameter	What it means	Where is it used
inwc	Tracer present in water column[1]	Used in most routines that deal with processes in the water column. If a tracer is not present in water column in will be skipped in these routines. This is only important for tracers that are used by diffusion, decay, deposition, settling or injection routines (nutrients, sediments, plankton). They are not important for fully age structured groups
insed	Tracer present in sediments[1] [1]A tracer can be present both in water column **AND** in sediments	Used in most routines that deal with sediment processes. For example, if a tracer is not allowed in the sediments the deposition routine will not deposit it in the sediment, but will leave it in the bottom water layer
dissol	Tracer dissolved in water [2]	Used in routines that deal with diffusion and advection processes
partic	Tracer is particulate[2] [2]A tracer can only be "dissolved" OR "particulate"	Used in routines that deal with particulate tracers. It is recommended to put this value to 1 for all tracers that are not dissolved (which naturally includes biological group tracers)

Parameter	What it means	Where is it used
passive	Tracer can be passively advected	Important for particulate tracers. Only passive particulate tracers are advected (moved by water between cells). Anything that is particulate and passive also require – particle size, sedimentation velocity (svel), and extra settling velocity (xvel)
decay	exponent in the burial or removal rate per second Note, this is not an actual decomposition into other nutrients (like decay of organic nitrogen into NO3), but a complete removal of the variable from the system, such as burial in the sediments	Used in the Physics submodel to calculate decay rates This parameter is very important and will determine **the rate of tracer removal** from the system. It is typically set to 0 for most tracers)
svel	Sedimentation velocity, m/second	Used in the Physics submodel to calculate sedimentation rate of particulate tracers
xvel	Extra settling velocity - due to migration of nutrient limitation. This parameter was inherited from PPIB and ERSEM models (see below) and is typically set to 0	Used in the Physics submodel to calculate sedimentation rate of particulate tracers
psize	Particle size, in meters Particle size will determine deposition of particulate matter in sediments	Used in the Physics submodel to calculate sedimentation and resuspension rate
b_dens	Particle bulk density ($mg*m^{-3}$)	Used in the Physics submodel to calculate deposition and porosity. Used in *deposition()* and in calculating sediment density. These parameters in the real world dictate particle movements in real water, but in Atlantis default values would do, because sediment properties are not modelled in such detail.
i_conc	Particle initial concentration in the deposits	Used in the Physics submodel to calculate deposition and porosity. It is used in *deposition()* routine only
f_conc	Particle final concentration in the deposits, after compaction	**This was only used in old models. Current code does not read this in.**

Parameter	What it means	Where is it used
FillValue	Default value to use when automatically filling empty cells in the initial_conditions.nc arrays	This is often used when tracer the initial concentration in most cells is identical

2.9 Tools and packages to view and analyse Atlantis input and output files

The Atlantis community has developed a number of different tools to view and analyse these outputs. The main tools used currently include:

1) An R package "atlantistools" contributed by Alexander Keth, available from https://github.com/alketh/atlantistools

2) A number of excel template files, matlab and R scripts developed by different users and available on Atlantis wiki

3) An R package "rlantis", available on https://github.com/mareframe/rlantis

4) netcdf viewing packages Olive and Dive useful for spatial plotting of variables in the NC files

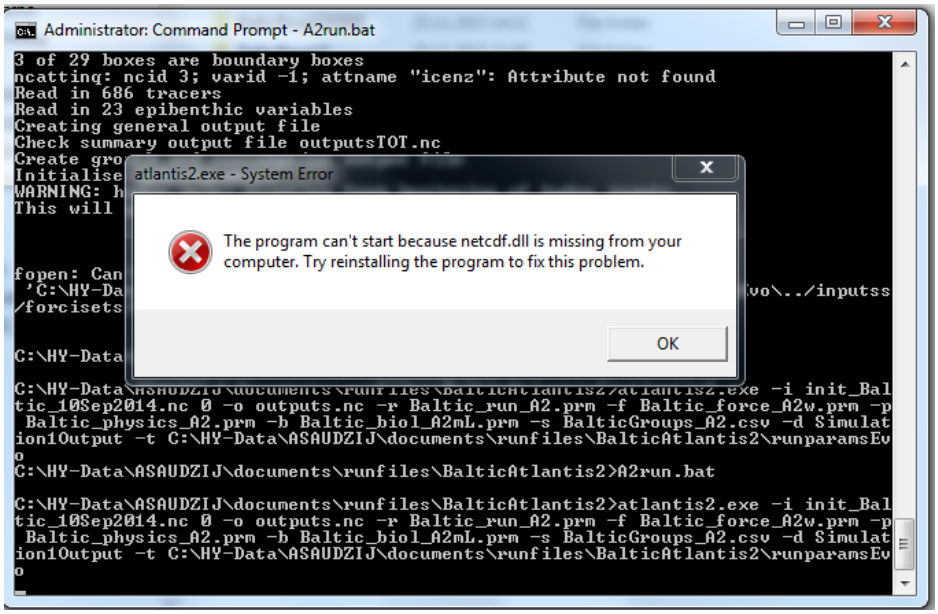
5) R based Shiny application "shinyrAtlantis" contributed by Shane Richards and available on https://github.com/shanearichards/shinyrAtlantis. This package is very useful for visualising the *initial_conditions.nc* and parameter files in a spatial context.

A number of other packages and tools are in different stages of development. So stay tuned with updates by joining the Atlantis google group and regularly checking for updates on the Atlantis wiki. If you have a tool you think others would benefit from please feel free to share it via the wiki or the Atlantis google group.

2.10. Troubleshooting on installation and running

Don't be disappointed if your first run attempt fails. This is very common! Below is a list of some common errors and more are available on the Atlantis wiki:

1. Missing DLL files. This is one of the most common problems for Windows users. Make sure you have installed the right version of the NetCDF library and updated the path as described on the wiki. If issues still persist, you might need to remove and reinstall the NetCDF library.



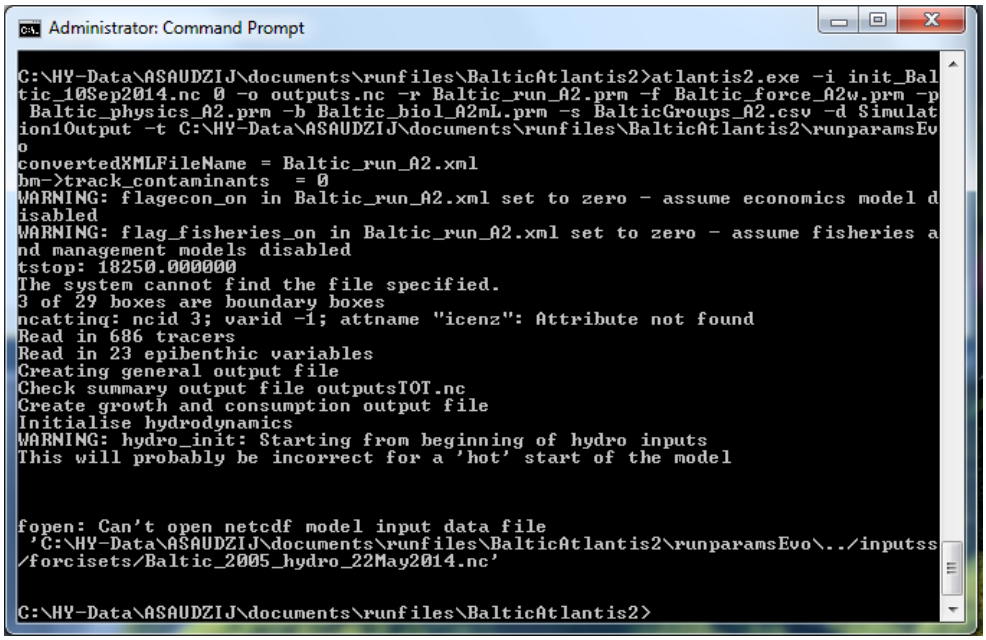
2. "Can't open input data file". Make sure the paths specified in the *force.prm* file are correct.

In the case below there was a typo in the path, the 'inputs' directory folder was misspelled

hd0.name ../input**s**/forcisets/Baltic_2005_hydro_22May2014.nc

Once this is fixed into
hd0.name ../inputs/forcisets/Baltic_2005_hydro_22May2014.nc

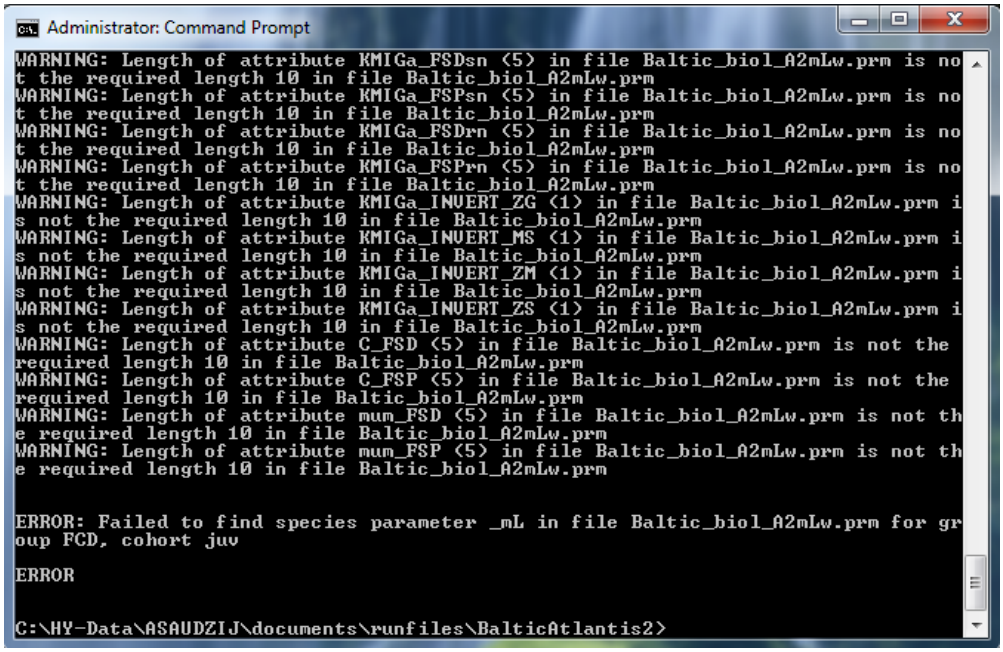
the run went smoothly



3. Failed to find a parameter. If some of the required parameters in the PRM files are missing Atlantis will quit and give an error message. In the case below Atlantis could not find a parameter FCD_mL. This happens when mistakes were made in formatting parameter files (e.g. if you used excel

to create your file check to see if the PRM file contains " " around a parameter name, if does remove them and just leave the name) or when new obligatory parameters were added during the code update to account for new functionality (and you have updated the code on your computer)

Keep an eye on the Atlantis wiki news blog for information on modifications to the code and any new required parameters. To find the missing parameters search the Atlantis wiki for the parameter name to see what is expected.



4. Atlantis crashes with a specific error message on physics, biology or other processes.

Atlantis code has inbuilt checks that will cause the program to quit if things go wrong. This can happen if biomasses of some lower trophic groups go to very low levels, or volumes of cells change too much. These messages cannot all be included here, but the best way to understand them is to search for keywords in the message in the code itself. This approach is also useful in understanding where different parameters are used in the code.

To search for a specific string in the entire code, select Quick Find (Ctrl+F) from the Edit menu, and enter the string you want to find. Make sure you search in the "Entire Solution" (selected in the "Look in:" menu) rather than the current document or routine. Also make sure you enter the text accurately.

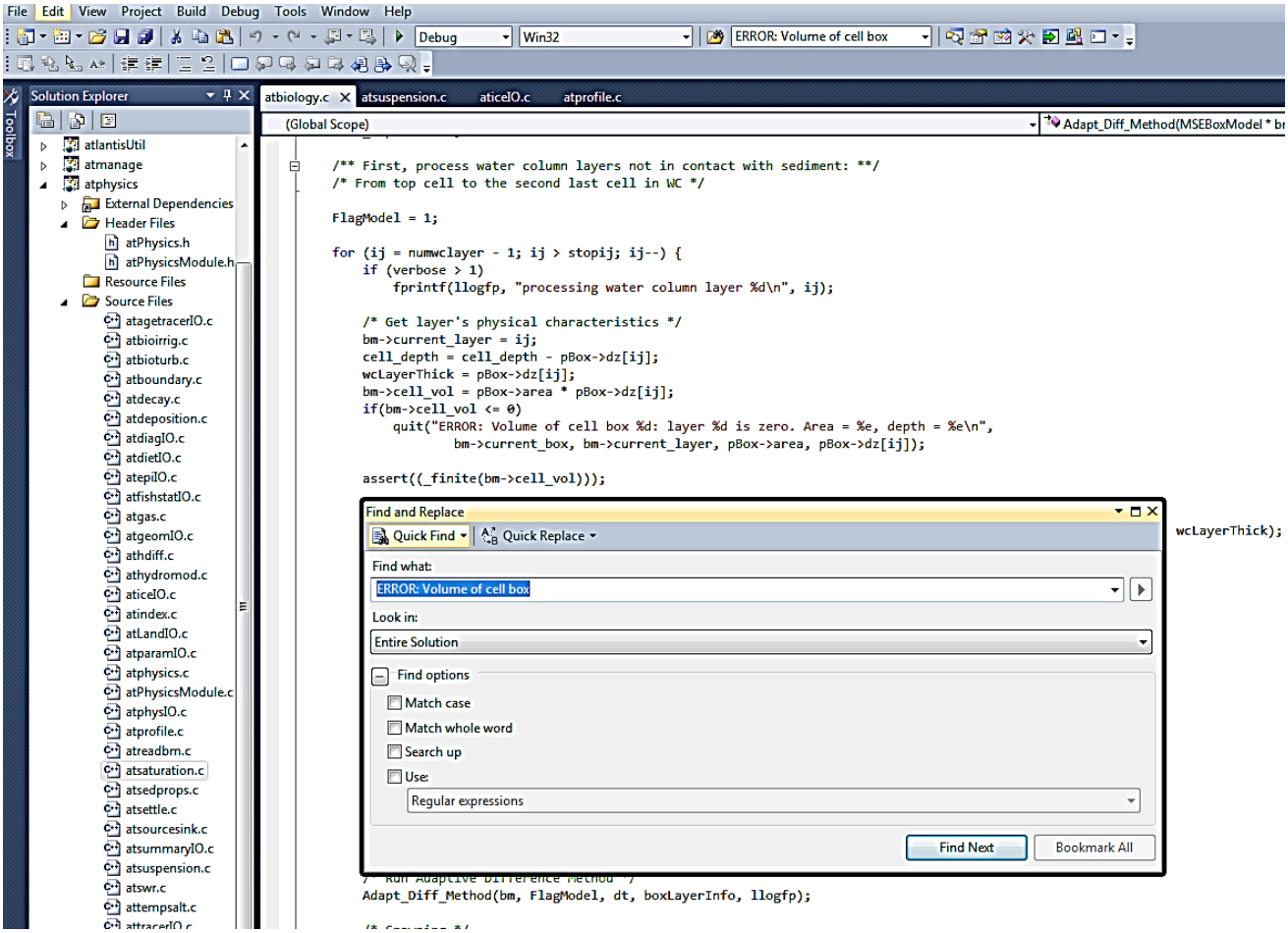
When you search for an error message only enter general text, e.g. "ERROR: Volume of cell box", as in the example shown above. Do not enter specific numbers from the error message, as they are not present in the code (they are specific to that one simulation only). For example, the crash message might say

ERROR: Volume of cell box 5: layer 1 is zero…"

However the message built in the code actually looks like this:

ERROR: Volume of cell box %d: layer %d is zero

where %d values will be replaced by specific numbers.



Finding the routine that leads to the error message may not immediately answer your questions, but it might give ideas about where things went wrong and which parameters might have to change (look at the parameters used in this routine). It is not necessarily straightforward and easy, but as you get used to searching through the code and understand it better, things will slowly start making sense.

A document compiling error messages is available on Google Doc file (available on March 20, 2016): https://docs.google.com/document/d/1B3X6_Y29gZxtKzjZY2oyylAJBxVtYXXu72pVbwv6mWw/edit

Please check this document for further information and to contribute to it by posting error messages you get.

3. GEOMETRY OF THE MODEL DOMAIN

3.1. Boxes and layers defining the geometry of the model

Atlantis is a spatially-explicit model, which means that it simulates spatial variation in biogeochemical and socio-economic processes. The spatial domain or geometry of the specific model consists of boxes or polygons subdivided into vertical sediment, water and optional ice depth layers. Land is also an optional inclusion, represented via sediment layers only. The number and shapes of boxes, as well as a number and depth of vertical layers is defined by the user. One layer of one box is called a cell. A cell is the main model spatial unit. Most biogeochemical, biological or fisheries processes are replicated in each water column cell and are considered uniform within a cell. Movement between the water column cells is by passive advective transfer (forced by hydrodynamic forcing files) or directed active movement. Only sediment relevant processes are executed in the sediments, similarly for ice or land. Direct movement between cells of these special types is not possible for ice or sediments, though some directed movement is possible on land. Movement between the special cell types and water column cells may happen due to processes like deposition and melting.

The advantage of using a unique model polygon and depth domain, rather than a standard grid, is that the user can match the model geometry to the geographical and bioregional features of the simulated marine system. Smaller, higher-resolution polygons can be defined in areas of particular interest (Johnson et al. 2011) while open water areas can be modelled as one or several large polygons saving a significant amount of computational time. This means that you should think carefully about defining the model geometry (and forcing file conversion, see Chapter 4). Models focused on coastal fisheries often use bathymetry to define polygons, picking boundary isobaths that are important predictors of species distribution or fisheries management zones - such as states, provinces or marine protected areas. Oceanographers may want to isolate areas with persistent currents, eddies, or other oceanographic features. Biogeographic breaks (headlands, capes) may also be important to consider. Finally, the spatial distribution of biological and fisheries data, and the spatial scale of interest to stakeholders and other users of your model also need to be considered.

In principal users can setup very complex shaped polygons, however, the number of polygons and faces between them will determine the speed of calculations. Remember that all biological and fisheries process calculations are repeated in each layer of each polygon!

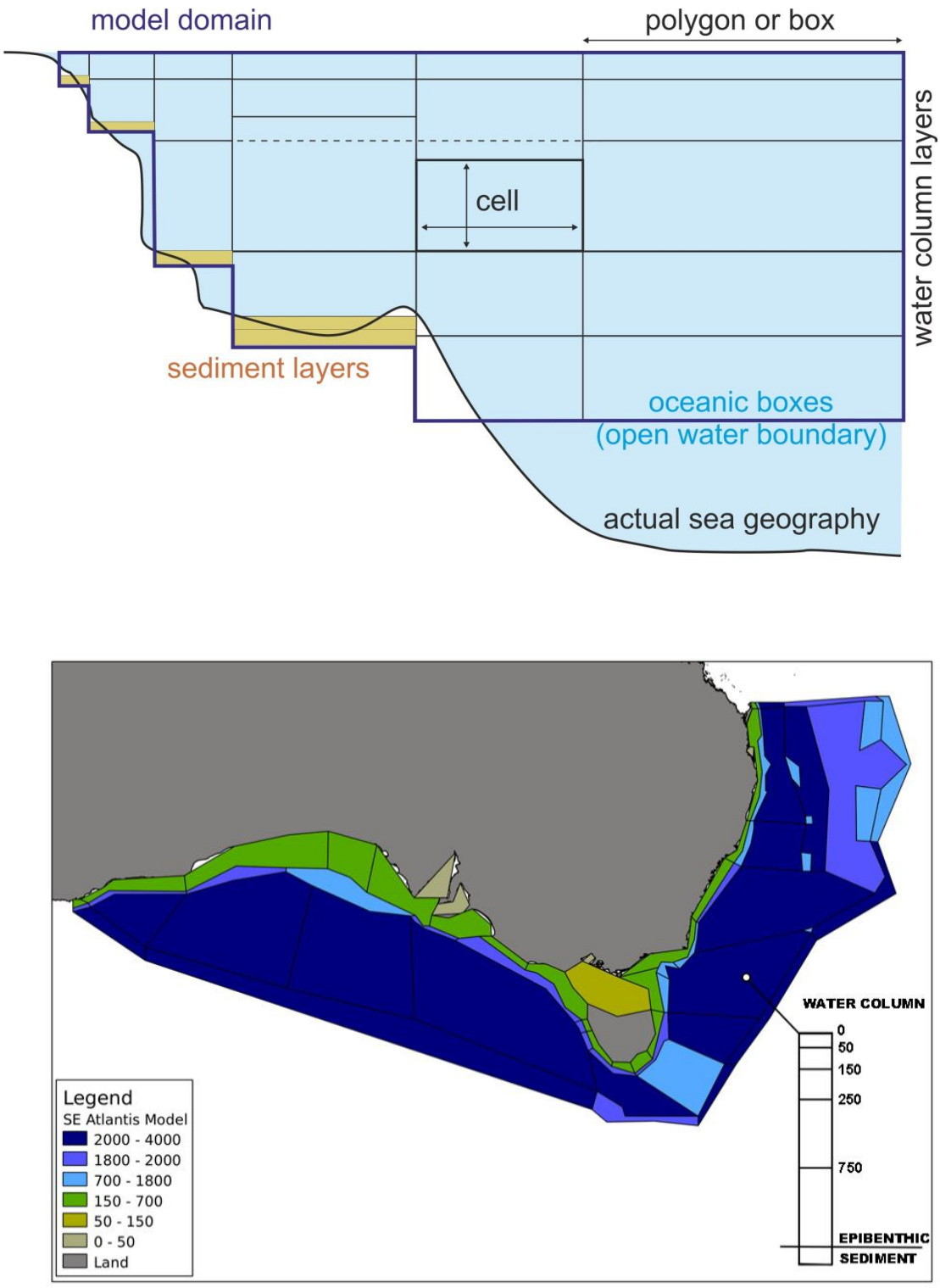


Figure 4. Top: An example cross section of an Atlantis model, showing water column and sediment layers. Oceanic boxes have an open water boundary at the bottom of the deepest water layer. The depths of vertical layers do not have to be the same in all boxes, although most models keep them identical for simplicity (shown with horizontal dashed line). The depth of the bottom water column layer varies depending on the geography of the marine system.
Bottom: An example of the horizontal Atlantis model domain (set in the BGM file)

3.2. How to define a geometry file for a new model?

The first step in developing a new model is to define the two-dimensional domain specified in the BGM file. Atlantis was created prior to the development of shapefiles that have now become a standard in GIS applications. Hence Atlantis uses its own BGM file format (see wiki BGM file page), which was a common format for many early box models. These days the developers of new Atlantis applications typically define model geometry and shapes of the polygons using GIS tools, but any other alternative approaches that can provide geographic representation can be used. The GIS tools can export shapefiles, which are converted into the Atlantis-specific BGM file using specially written packages. A new R tool (rbgm, see below) is also being developed to enable users to convert files from GIS or other spatial editing applications directly to BGM files, skipping the shapefile stage (see below). The BGM file is also used by some software packages, such as Dive or Olive, for spatial visualisation of Atlantis outputs.

Details of the model geometry are defined in the **BGM** and *intial_conditions.nc* files.

The **BGM** file defines the **two-dimensional** geometry of the model domain. It includes information on boxes, their shapes, midpoints, faces. It also specifies the maximum depth of the modelled ecosystem and maximum depth of each box.

The ***initial_condition.nc*** file includes further information on the **three-dimensional geometry**, specifying numbers and depth of the vertical layers in each box. The maximum number of water column and sediment layers is set in the global attribute section in the *initial_conditions.nc* file (wcnz and sednz parameters). The actual number of water column and the ID of the top active sediment layer (typically 0) in each box is then given in the numlayers and topk parameters given for each box.

Nominal or initial **depths of the water column and sediment layers** are set in the *initial_ condition.nc* nominal_dz parameter. Typically they are set to be identical among boxes (see Fig. 4 top panel), except for the bottom layer, but this is not strictly required. The actual depths of the water column and sediment layers can change dynamically through the simulation and are stored in the dz variable. The maximum change in water column depth is set in wc_dz_tol , whereas minimum and maximum depth of the sediments layer(s) is set in the minseddz and maxseddz parameters in the *physics.prm.*

The optional ice layers are included as a fourth dimension in the *initial_conditions.nc* file. Modelling ice in Atlantis is still under development, see chapter 5.5.2

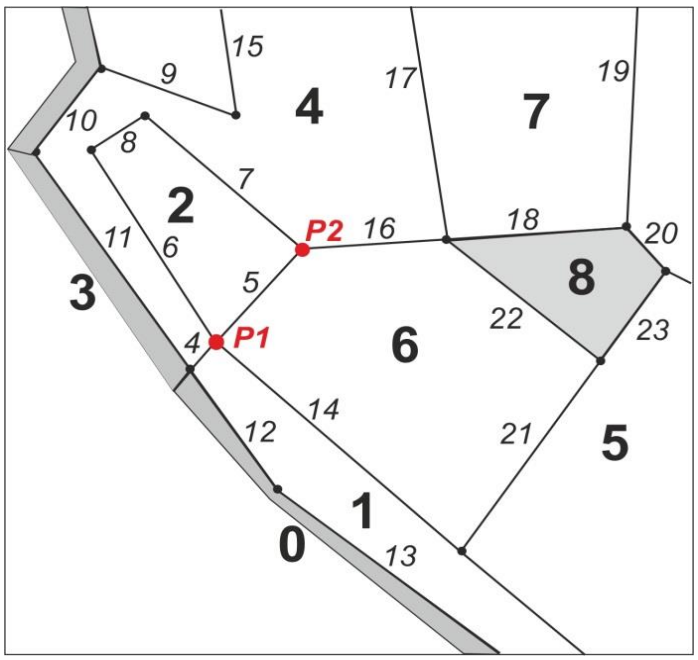


Figure 5. **Example of an Atlantis model geometry (part of a model domain). Boxes are numbered in** bold**, grey boxes indicate boundary or island (land) boxes. Faces are numbered in *italic* numbers. Vertices are shown as dots; vertices P1 and P2 describe face 5.**

There are **two main types of boxes: dynamic and boundary**. The corners of each polygon are defined by **vertices** and the lines connecting these corners are called **faces**.

Dynamic boxes make up the model domain; all biological and socio-economic processes are modelled in them. Dynamic boxes must satisfy the (approximate) requirement of mass onservation4, although their volume can change slightly due to currents and tides (just as seen in the real world). In dynamic boxes numeric fluxes of a tracer (e.g. NO3, phytoplankton) due to ecological processes (such as growth and mortality) cannot exceed a certain proportion of its standing biomass. This proportion is set by the RelTol parameter in the *biology.prm* file.

Boundary boxes are used only as sources and sinks for advective transport. They represent the "outer world" or the system beyond the model domain and don't have the requirement of mass conservation. The areas of boundary boxes play no role (they can be set as tiny slivers or large boxes) and there are no water or tracer movements between boundary boxes. Islands inside the model domain are modelled as boundary boxes (unless land has been activated in the model by setting flagAllowLand to 1 in the *run.prm* file). In models where land is not active islands and other land boxes have zero depth (box 8 in the Fig. 5). However, when land is active (only in the most recently development versions of Atlantis) explicit modelling of land is allowed (see chapter 5.5.1), these boxes use a positive value for total box depth (botz) rather than the negative values used for aquatic boxes.

In the vertical dimension the polygons can be either **non-oceanic** or **oceanic**. Sometimes the Atlantis model does not capture the full depth of the marine ecosystem. For example, the actual maximum depth in the modelled ocean area can be 3500m (shown in the maxbotz parameter in the BGM file), but the model domain only includes the top 2000m (box-specific botz parameters in the BGM file). The polygons for which botz is greater or equal to maxbotz are classified as oceanic (e.g. a maxbotz of -2000 versus a botz of -3500). This means that the bottom water layer in these polygons has an open water boundary with deeper (not explicitly modelled) waters rather than with a sediment layer. In oceanic boxes the sediment layer is still included in the setup (for simplicity) but sediment calculations

4 A small amount of mismatch (typically to cover rounding error) is allowed over the total fluxes in a cell in any one time step, but if it becomes too large (i.e. exceeds the flux tolerance set in the *biol.prm* file, with typical values being on the order of 0.2 mgN*m^{-3} per second) the model will quit and provide a warning message, as this indicates that there is a problem with the parameterisation

are not performed in these boxes when the model is run and the biomass of organisms in the sediments are not included in the total biomass calculations. Further, the deepest layers of the oceanic boxes receive nutrient input from "outside" of the model domain, designed to simulate nutrient mixing from deeper layers not included in the model domain.

Below is an example of a BGM file with explanations of parameters given after #

```
# box model geometry based on VMPA_setas_20051216.bgm

# conversion from lat/long space to x-y space of the model domain. Required for mapping and light level
# calculations (when lim_sum_hours is turned on in the biology.prm file). Definition is based on GIS standards
#  The parameters should be space separated

projection proj=aea lat_1=-18 lat_2=-36 lat_0=0 lon_0=134 x_0=3000000 y_0=6000000 ellps=GRS80 towgs84=0,0,0,0,0,0,0 units=m no_defs

# Number of boxes in horizontal plane
nbox 11

# Number of faces in horizontal plane
nface 22

# Maximum bottom depth (m) of the model domain (not the real ecosystem)
maxwcbotz     -5000

# vertices of the polygon defining the boundary of the dynamic model space throughout the total geographical
# extent of the model, creating one big polygon. They are ordered sequentially in either clockwise or
# anticlockwise direction (it doesn't matter which so long as it remains consistent)= in (x,y) format, where x,y is
# in projected (linear transformation) of longitude and latitude

bnd_vert 4227984.241 1449270.8
bnd_vert 4279136.436 1451037.995
bnd_vert 4684173.263 1362498.346
… and so on for all vertices

# Data for box number 0                    ## user defined box number identifier
box0.label      Box0                       ##users reference box name, e.g. Box0 or Sydney Harbour
box0.inside     4043667.571 1150676.493    ## midpoint of polygon
box0.nconn      5                          ## number of faces shared with other dynamic boxes
```

```
box0.iface      1 10 11 13 17                   ## id numbers of shared faces, each face has a
unique id
box0.ibox       0 2 2 2 2                       ## box that each shared face corresponds to
box0.botz       -370                            ## maximum depth of the polygon in the real
ecosystem
box0.area       3261982282                      ## area of the box, m^2
box0.vertmix    0.000001                        ## vertical mixing scalar for the polygon
box0.horizmix   1                               ## horizontal transport scalar for the polygon
box0.vert 4005591.597 1198429.099               ## List of vertices (x,y) in clockwise or counter
clockwise order
box0.vert 3998613.455 1174049.433
box0.vert 4065533.389 1102640.49
box0.vert 4095090.658 1128446.461
box0.vert 4095245.855 1128504.676
box0.vert 4005591.597 1198429.099
… and so on for all boxes

# This section describes each dynamic face (faces of all non-boundary boxes)
# Data for face number 0
face0.p1        4005591.597 1198429.099         ## one endpoint of face (x1,y1)
face0.p2        4095245.855 1128504.676         ## other endpoint of face (x2,y2)
face0.length    113698.3335                     ## length of face
face0.cs        -0.788527462 0.614999546        ## Cosine and sine of the slope of the face
(slope=sine/cosine)
face0.lr        2 0                             ## Number of the box to the left and to the right as
you look
                                                ## from point 1 to point 2. Correct orientation is
crucial! (see
                                                ##  below)
… and so on for all faces
```

Below are some strict rules and best practice suggestions on setting up the model geometry. These rules are provided here to give general insights into how the geometry file is setup. If you are developing a BGM file for a new model, check more detailed instructions on Atlantis wiki (links given below).

Rules:

1) Box 0 **must** be a **boundary** box and Box 1 **must** be a **dynamic** box.
This rule is a developmental legacy and determined by the way the Atlantis code and oceanographic file conversion was setup during the model development. Box 0 is special because it is set up to have water and nutrient fluxes to nearly all other model boxes. This is set to simulate supply of advective tracers to the oceanic boxes and accommodate any missing water fluxes due to conversion of oceanographic files into Atlantis input files (see chapter 4). While the model is running it also acts as a store for information on any migrating groups currently outside the model domain.

2) **Doughnut shaped boxes are not allowed**.

For example, you cannot have one box around an island. This is because such boxes have their midpoint on land, which would affect calculations of a number of processes relying on midpoint values.

3) The description of boxes, faces and vertices in the BGM file (iface, ibox, bnd_vert parameters) **must be done in the same orientation** – clockwise or counter clockwise. It doesn't matter which you chose, just make sure it is consistent throughout.
The orientation can be checked using the Checkwinding package developed by Cameron Ainsworth and available from the Atlantis wiki.

4) Islands **must** be represented as boxes with a depth of 0 (or >0 if land is active), not as empty regions. By default the land boxes are either treated as boundary boxes, but can also be modelled as land, if the land option is activated (see chapter 5.5.1).

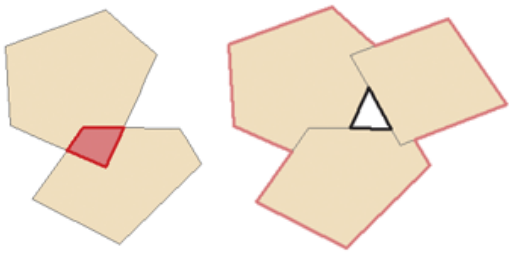
5) It is of utmost importance to **correctly identify the boxes to the left and to the right of a face** in the face#.lr parameter (where # stands for the box number). The points P1 and P2 in the Figure 3 describe face 5 between boxes 2 and 6. If face5.p1 is given as P1 and face5.p2 is P2, then left and right boxes are designated as you look from P1 to P2. This means the right box is 6 and the left box is 2 and face5.lr must be set as 2 6. Setting the direction incorrectly inverts the direction of the flow in the hydrodynamics forcing file! Checkwinding package will check the file for correct right/left designation.

6) The **midpoint** of a polygon is defined in the box#.inside parameter. It is used to: i) calculate the temperature in a box when temperature forcing data is not provided and we are in the southern hemisphere; ii) calculate the sunrise and sunset values based on lat/long when lim_sun_hours is set to 1; iii) calculate distance between boxes and invasion speed if the invasion option is active; iv) calculate distance from a box to a port in the dynamic fishing code. **If the midpoint of the box is misplaced and is not within the actual box, Atlantis will quit when trying to read time series that deposit something into that box or when running dynamic fishing routines.**

Suggestions:

7) Keep the number of boxes as small as possible (fewer than 90 is good), remember the fewer the boxes the faster the model will run (e.g. models with 20-30 boxes will run three to four times faster than one with 90 boxes!)

8) Keep the number of dynamic faces (those corresponding to non-boundary boxes) to less than 400, but 100-200 would be even better! It is not recommended to have boxes with very complex shapes (i.e. do not slavishly follow complex isobaths in the GIS when drawing up the box shapes). This is because complex box shapes are 1) is prone to error and 2) may give a false impression of reality. Simple shaped boxes serve as a good reminder that the spatial results presented are only model simulations and should be treated as such



9) When setting up box shapes in GIS, make sure the topology is perfect, which means no gaps or overlaps between polygons. If the shapefile has overlaps or gaps it won't convert to the BGM file

10) In some cases conversion of water fluxes from oceanographic files to Atlantis hydrodynamic forcing files leaves small coastal boxes isolated, i.e. no water fluxes enter or leave them. This will be covered in more detail in Chapter 5.4. The BGM file has two box-specific water mixing parameters that are used in some physics routines, including those used to correct for the lack of mixing. These parameters allow box-specific mixing scalars to be setup and help to tune vertical and horizontal water flows in the model domain:
box#.vertmix - vertical mixing scalar used only when vert_diffusion or vert_mix are turned on in the *physics.prm*.
box#.horizmix – horizontal mixing scalar that can be used to correct for hyperdifusion (if insufficient correction was done during the file conversion stage, see Chapter 5.4). It is also used as a box-specific scalar if fill_zero_exchange or horiz_diffusion flags are turned on (set to 1) in the *physics.prm* file (see Chapter 5 for further details).

3.3. Practical steps for building the BGM files from GIS shapefiles

1. Define polygons using QGIS or similar GIS package. This will produce a shapefile of the model domain. The shapefile is the master of the model geometry, any changes should be made to the shapefile and the BGM should be recreated from that master. Hand editing of the BGM should be avoided5, other than to manually set the box-specific mixing values, or to correct any issues resulting from BGM generation (like misplacement of the inside point). See the Atlantis wiki BGM file page for more information.

2. Run the BGMeriser package, to convert shapefiles into BGM files. Details on access and installation are given on the wiki (here) as are details on how to run it (here).

3. If the shapefiles do not pass through the BGMeriser, they probably have issues with gaps or overlapping polygons and should be cleaned. Check the instructions on the wiki for tips on cleaning up the shapefiles using QGIS.

4. Use Checkwinding to check the orientation of boxes and faces in the BGM file

5 Expert users may be able to tinker directly with the BGM file, but extreme care should be taken and it should be documented so any future users can understand the changes made (as the master shapefile file will no longer 100% match the model geometry).

NOTE!

Please keep in mind that **shapefiles do not preserve all information about the faces** among the boxes (see chapter 3.5). Shapefiles only enumerate faces, but they do not give information about their orientation. They also do not preserve topological information among boxes, i.e. their relationship to one another. Correctly created shapefiles include user defined extra attributes to contain the additional information required to generate a BGM file. However, when shapefiles are edited in a GIS, all face information may change, especially if some boxes are added or deleted. This is why BGMeriser must be run after each shapefile amendment. This is an important disadvantage of using shapefiles for Atlantis model geometry and a **new R package is being developed** to allow users create BGM files directly from GIS or other applications, without having to use shapefiles (see chapter 3.5).

3.4. How to view your model geometry?

The two-dimensional model geometry is now defined in your shapefiles and in BGM files. **The shapefiles can be viewed using QGIS or a similar GIS package.** If shapefiles are not available, BGM files can be converted into shapefiles using BGM2SHP package.

The BGM files are also read by various tools used to visualise Atlantis outputs in a spatial context, listed in chapter 2.9

3.5 Using R for BGM geometry

There is an R package ***rbgm*** for working with BGM files contributed by Michael Sumner. The aim of the package is to provide a common platform for reading and writing BGM files, which then also can be used by other Atlantis R applications that require information on the model geometry. Generally, shapefiles do not store all information required to completely re-create a BGM file. One needs at least a line shapefile and a polygon shapefile to store what is in the BGM and also some notes to go with it, which is why a shapefile is only a starting point for a BGM file. Note that BGMeriser **will** create a valid BGM file from a shapefile, but it has to be configured in the right way.

Using ***rgbm*** you can

- read the BGM format, maintaining all topology and attributes
- convert from BGM to GIS-like objects using the Spatial tools in R (**sp** package)

The "GIS-like" object is an R equivalent of a "shapefile", which consists of "layers" of geometric shapes (polygons, lines, or points) that are linked to a table of attributes. The BGM geometry is really a combination of information on lines (faces) and polygons (boxes), where each face has information about its orientation and its neighbouring faces and boxes. R uses Spatial layers for separate description of a SpatialPolygonsDataFrame (a polygon shapefile), SpatialLinesDataFrame (a line shapefile) and SpatialPointsDataFrame (a point shapefile).

Once the BGM is loaded in R, it can be used to visualise faces and boxes as lines and polygons, and use these objects to extract data from other sources, like doing spatial queries with other map data, extracting gridded data summaries, and using these for visualization or further analysis (see rbgm

description for further information). A goal for ***rbgm*** is to drive coupling of Atlantis with physical data and models via R. ***rbgm*** is able to write from Spatial forms to shapefiles and other formats, but since we can read directly from BGM or from a GIS or geodatabase, this also means that we don't necessarily need to create shapefiles at all (soon **rbgm** will be able to create BGM files after reading the input from QGIS or Manifold GIS projects).

A collection of publicly available BGM files from existing Atlantis models is also available. This collection can be viewed using an R package ***bgmfiles***.

4. HYDRODYNAMIC, SALINITY AND TEMPERATURE FORCING DATA

4.1. General introduction to forcing of Atlantis models

Like most other ecosystem models, Atlantis does not calculate water fluxes (current, circulation) itself, but uses outputs from specialised oceanographic models such as ROMS (Regional Ocean Modeling System) (Shchepetkin and McWilliams 2005; Haidvogel et al. 2008), BLUELINK (Brassington et al. 2007) or HYCOM (Chassignet et al. 2006). In other words, Atlantis models are forced with hydrodynamic data from oceanographic models. At the minimum level Atlantis requires only a hydrodynamic forcing input file defining water fluxes across boxes and layers. The model will then use simple routines to calculate temperature and salinity conditions in each cell at each simulation time step. However, most Atlantis models, and other ecosystem models (Fiechter et al. 2014; Rose et al. 2015), also use oceanographic model output to force temperature and salinity conditions (and in some cases also oxygen, pH or other tracers). This is because oceanographic models are specialised to model temperature and salinity conditions, and their calculation of these parameters are deemed to be more accurate than those used in Atlantis. Moreover, taking the physical properties from the oceanographic model means they are consistent with the exchanges (currents) being read in from the oceanographic model. In rare cases, if oceanographic models are not available, water flux information can be pieced together from observations or literature to generate hydrodynamic forcing files manually. Table 5 lists some of the spatial domains and forcing characteristics of example model applications.

Oceanographic models simulate processes on their native grid, which is typically of much higher spatial resolution than Atlantis. The temporal scale in oceanographic models is also often in the range of seconds rather than the 6h, 12h or 24h used in Atlantis. Further, most oceanographic models use the same number of water column layers regardless of total depth (called sigma layers) (Fig. 6). This means that the depth of a layer in oceanographic models is very different in shallow and deep waters, while in Atlantis, water layer depths remain relatively stable (with small variations due to tides). This means that water fluxes and values of all forced parameters must be interpolated onto Atlantis model polygons and cells.

Table 5: Hydrodynamic structure of some example Atlantis models. Advect = advection, temp = temperature, salt = salinity.

	SE Australia	Eastern Tasmania	Port Phillip Bay	Western-port	NEUS	California current	Gulf of California
Key oceanographic effects	Boundary currents	Boundary current, upwelling	River input and slow flushing	River input and flushing	Gulf Stream,	Boundary current, upwelling,	Colorado River input, tidal mixing

	SE Australia	Eastern Tasmania	Port Phillip Bay	Western-port	NEUS	California current	Gulf of California
					Boundary current, Georges Bank	El Nino	/ stratification
Number of boxes	71	11	60	27	30	62	66
Max number of water column layers	5	6	1	1	4	7	6
Ocean boundary depth	2400m	2400m	45m	30m	200m	1200m	2025m
Hydrodynamic model used to supply flows	OFAM-Bluelink[1]	OFAM-Bluelink	3D hyrdro model[2]	Empirically defined	HYCOM[3]	ROMS[4]	ROMS[5]
Hydrodynamic time series length (years)	10	10	5	1	1	7	23
Transport model variables	Advect, temp, salt	Advect, temp, salt	Advect	Advect	Advect, temp, salt	Advect, temp, salt	Advect, temp, salt
Environmental drivers	Inshore sediment change, temp, pH, nutrients, river flow		Nutrients	Nutrients	Nutrients	Temp, nutrients	Temp, nutrients, river flow
Climate change	Yes	No	No	No	No	Yes (impl.) [6]	No
Explicit larval advection	No	No	No	No	No	No	Yes

1. Oke et al 2005,
2. Walker 1997a, 1997b,
3. hycom.rsmas.miami.edu/las/,
4. ROMS Hermann – NOAA PMEL/JISAO, Seattle,
5. ROMS Pares-Sierra - Centro de Investigación Científica y de Educación Superior de Ensenada.
6 Implicit handling of climate change means that extra mortality terms (related to acidification) or forced changes (e.g. for temperature) are used rather than flows from down scaled global circulation models.

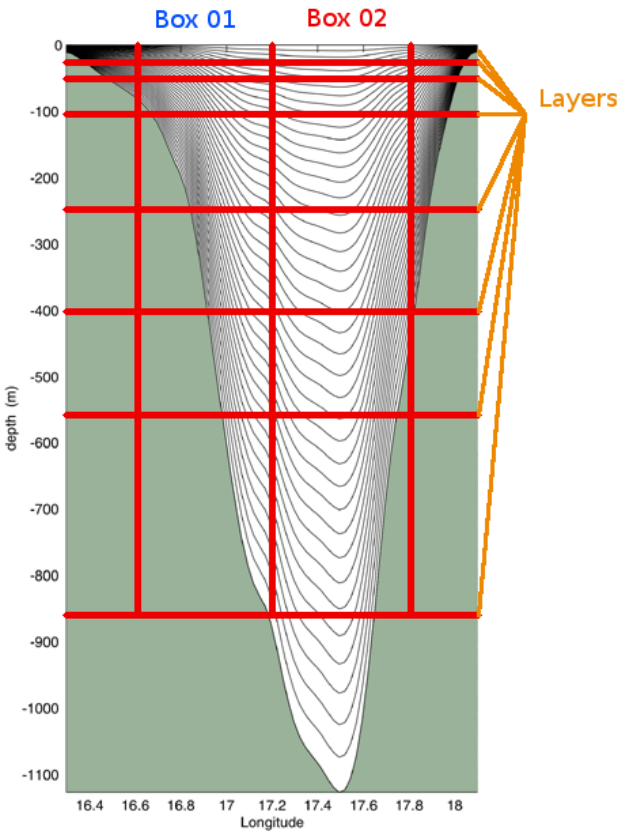


Figure 6. **An example of oceanographic sigma depth layers (grey lines) with an overlay of Atlantis spatial structure. Figure by Javier Porobic Garate**

4.2. Transforming oceanographic model output into Atlantis forcing files

The interpolation of the oceanographic output into Atlantis forcing files requires transformation across different shaped grids and depth layers. Water fluxes must be transformed from the oceanographic grid to Atlantis faces, whereas temperature and salinity values must be averaged onto Atlantis cells and time steps. The transformation must also account for hyperdiffusion, which is caused by the different spatial scales of oceanographic and Atlantis models. Because concentration of tracers within a layer of a box is assumed to be uniform, a tracer that enters a box is instantaneously assumed to be equally distributed throughout the entire box. This means that flow of tracers in large boxes would be overstated by orders of magnitude if this hyperdifussion phenomenon is not taken into account. The (crude) correction simply divides the east-west water flows by the width of the box (in metres) in the east-west orientation and north-south flows by the width of the box in the north-south orientation. This is a simple and a standard approach to correct for hyperdifusion and can be further refined using the horiz_mix scalar in the BGM file.

Fortunately, the transformation of oceanographic files is increasingly streamlined. Until recently most model developers used two codes contributed by CSIRO. We will describe them in a more detail below. However, new codes are currently being developed by the Atlantis community, i.e. an R script by Cecilia Hansen to construct hydrodynamic files from ROMS files. Please check the Atlantis wiki forcing files page and Atlantis google group for any new developments in this area.

The interpolation of oceanographic data onto the Atlantis model geometry requires three main conversions (Fig. 7):

1. Calculate **horizontal** water fluxes across Atlantis polygon faces, i.e. from one polygon to another
2. Calculate **vertical** water fluxes within Atlantis polygons.
3. Calculate **state variables** used for optional forcing, such as salinity and temperature for each cell and time step. For this step temperature and salinity values from high resolution oceanographic models are mapped onto Atlantis cells and a weighted average for each cell is calculated

These three steps constitute the main conversion and are conducted using the *Hydro* Matlab script (follow the link for further instructions on installation and use of the *Hydro* script and also check *Hydro* FAQ for more information). *Hydro* produces two NC files which are then used by *Hydroconstruct* to produce the final Atlantis forcing files. *Hydroconstruct* makes sure the order of water columns is in the right direction, transforms the water flux data into per layer per box fluxes and also corrects for hyperdifussion (using either area based scaling or area and box shape based scaling). The

Hydroconstruct running page and parameter file provides further information on the script and how to run it.

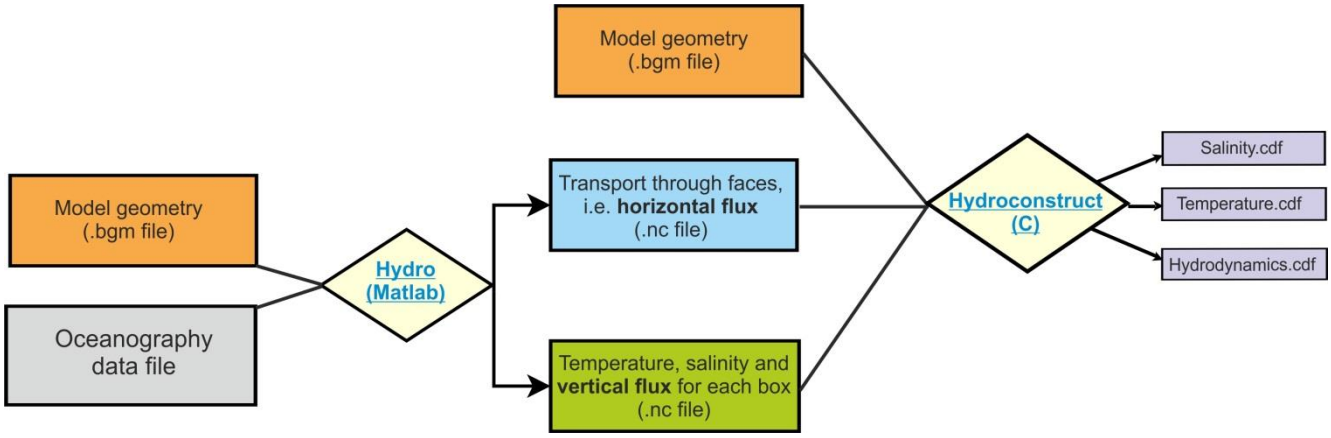


Figure 7. Main steps in converting oceanographic files into Atlantis input files, as done using two CSIRO contributed scripts *Hydro* and *Hydroconstruct*. Figure by Javier Porobic Garate

Good practice tip 3

Interpolation and interpretation of the oceanographic data is one of the main challenges for Atlantis models developers and users, because usually they are biologists and not oceanographers. It is therefore highly recommended to include an oceanographer in the Atlantis team, at least during the model development stage, but also if and when long term forcing of environmental trends is needed.

4.3. Things to keep in mind about hydrodynamic forcing files

Oceanographic modelling has made rapid progress in recent years and availability of high resolution data is increasing, but there are still a **few issued that should be kept in mind:**

1. **Absence of long term oceanographic data to study long term trends.** Oceanographers rarely conduct 50+ years of high resolution simulations, yet such time frames are often needed for Atlantis simulations. The length of oceanographic data time series usually more on the order of 1 to 25 years (see Table 4 for examples) and a common practice among Atlantis users is to 'loop' over the available oceanographic input years. One should be aware that if long term trends are present in the oceanographic data, they will be seen as jumps in average variable values every time the file is recycled. Also, such recycling alone is not suitable for studying long term environmental trends. Ideally long term forcing files should be obtained directly from long term oceanographic simulations.
As a crude approximation, **long term trends can be forced** onto hydrodynamic files using an R script contributed by Hem Nalini Morzaria Luna and used for the Gulf of California model. Alternatively, trends in temperature, salinity and pH can be added in the biology.prm file using several Tchange, Schange and PHchange parameters (see below)

2. **Insufficient resolution in coastal areas**. Most oceanographic global and publicly available models do not include coastal areas at sufficiently high resolution (in many old models they are not represented at all). Yet this is where a lot of important processes of nutrient input, recycling, spawning

and larval dispersal take place. It is therefore very important to assess the water fluxes in small or shallow coastal areas and make sure that they are sufficiently accurate (see below).

3. **Loss of resolution to represent eddies and upwelling.** Translating high resolution oceanographic models to the Atlantis polygons means that transformed forcing files may not capture the true magnitude of eddy strength or local upwelling, both of which are important for driving local productivity. To compensate for this loss of resolution it is recommended that local forcing of productivity be included using eddy parameters (see below) and/or forcing additional nutrient loading into coastal waters (to simulate upwelling; Brand et al. 2007).

4. **Oceanographic models can include 'drift'**, temporal artefact trends that bias long-term oceanographic outputs. If 'drift' is seen to be a problem, modellers can choose a limited set of years of oceanographic output and loop these years.

5. **Artificially inflated vertical fluxes.** Sometimes oceanographic models can inflate the vertical fluxes (they are used for tuning the horizontal fluxes), which can lead to too much vertical mixing. This is becoming less of a problem as the skill of oceanographic models improves with time, but it is still recommended to check the intensity of vertical fluxes. If vertical fluxes appear to be too strong they can be downscaled using vertical diffusion and vertical mixing routines (see below) or by using an R script contributed by Raphael Girardin (details and a link will be added when available, but also check the wiki). Similat 'jets' can appear in horizontal fluxes (particularly in older models and most often close to the boundary of the oceanographic model domain).

6. **Underestimated diffusion of tracers between boxes**. In real world there are lots of bi-directional flows between spatial domains that are modelled as distinct Atlantis boxes. However, when these flows are converted into hydrodynamic forcing files, all flows are averaged for each pair of cells, so that only one-directional flow occurs between two cells at each time step. This might under-represent diffusion of tracers between boxes, especially if boxes are large and if averages all go in the same direction in all layers of the two boxes.

NOTE!

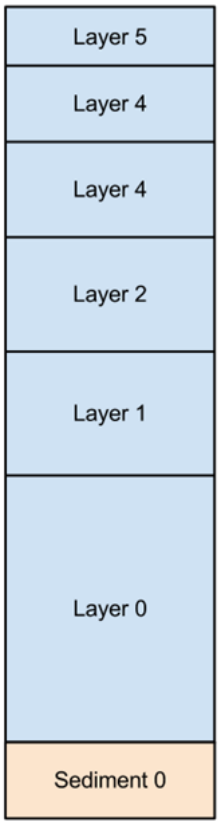
The **hydrodynamic forcing files are automatically** recycled or **looped** for the entire model simulation run. This is **not the case** with other forcing files, such as temperature, salinity or other optional forcing files. To rewind the temperature and salinity forcing files set the flags temp_rewind and salt_rewind to 1 in the force.prm file. If the flags are set to 0 then the simulations will use the last temperature and salinity value once the forcing file time series ends. To rewind any other optional forced tracers make sure you set TracerName_rewind (e.g. pH_rewind) parameter to 1.

NOTE!

The hydrodynamic forcing files **are not necessarily the final water transport** in your model. They provide the horizontal and vertical water fluxes, which are then used in the transport model described in the *(transportBM)* routine in the attransport.c file.

The transport model uses the water exchanges from the hydrodynamic forcing files, but can also:
1) apply water flow corrections to isolated boxes according to specified parameters

2) add local water inputs or sinks, such as river flows or outflows
3) apply water exchanges through the surface layer due to precipitation and evaporation
The final flows of water are given in the output.nc file listed in hdsource, hdsink, eflux and vflux (see description of the ouput file parameters in chapter 5.6). If some of the optional physics routines are used, then these final exhcanges may be different from the hydrodynamic forcing inputs.



4.4. Numbering of vertical layers in the input and output NC files

Atlantis allows users to define several water column and sediment layers, although most applications have only one sediment layer.

Layer counting starts from the sediment-water interface.

The water layer 0 is always closest to the sediment, **the next layer up is layer 1 and so on.** Sediment layer 0 is closest to the water, **the next layer down is 1 and so on.**

In the.nc files water column layers are listed first, followed by the sediment layers.

layer 0, layer 1, layer 2, layer 3, layer 4, layer 5, sediment 0

In the model setup shown here, the layer corresponding to the water column's surface layer can be anything from layer 0 to layer 5**, depending on the total depth**

of a box. In deep boxes the surface layer will be layer 5, while for the shallowest boxes the surface will have layer 0 as their surface layer.

NOTE!

The ordering of layers in output files is opposite to that in the input file. This is a development legacy aimed to improve the viewing of output files with NC file viewers. This also means that the position of missing water column layers in arrays describing boxes with fewer than maximum layers is reversed between the input and output files.

In the input.nc **file a shallow water box with three water layers will look like:**
layer 0, layer 1, layer 2, *0, 0, 0,* ***sediment 0***

In the output.nc **file the same box will look like:**
0, 0, 0, ***layer 0, layer 1, layer 2, sediment 0***

A good way to understand this is to look at temperature variables. Note that warmer water is on the surface.

Input.nc **file:**

Temp =
15.75, 18.54, 18.55, 0, 0, 0, **11.85,**
16.38, 17.64, 17.49, 0, 0, 0, **11.77,**
15.66, 17.34, 17.31, 0, 0, 0, **11.57,**
16.05, 16.92, 16.87, 0, 0, 0, **11.29,**
5.15, 11.85, 13.92, 17.02, 19.01, 19.83, 0.32,
5.29, 11.91, 13.85, 16.39, 18.50, 19.48, 0.93,
…

Output.nc **file:**

Temp =
0, 0, 0, **15.75, 18.54, 18.55, 11.85,**
0, 0, 0, **16.38, 17.64, 17.49, 11.77,**
0, 0, 0, **15.66, 17.34, 17.31, 11.57,**
0, 0, 0, **16.05, 16.92, 16.87, 11.29,**
5.15, 11.85, 13.92, 17.02, 19.01, 19.83, 0.32,
5.29, 11.91, 13.85, 16.39, 18.50, 19.48, 0.93,
…

4.5. Viewing and checking forcing input files with R

An R-package ***shinyrAtlantis*** is available on https://github.com/shanearichards/shinyrAtlantis and provides a number of shiny applications to help with viewing Atlantis forcing and parameter files.

For exploring hydrodynamic forcing files the package provides a function 'make.exchange.object' that reads in the BGM file containing box geometries and the NC file containing the exchanges. The function 'make.exchange.object' also needs to be provided with the cumulative layer depths. It then produces a data object that can be fed into the shiny application 'sh.exchange'. The 'sh.exchange' provides a visual description of the exchange data. It visualises connections and exchanges among boxes and layers and helps to identify if any boxes are isolated. A time-series of exchanges for each box can also be generated, as well as a two-dimensional visualisation of the horizontal flow throughout the domain. Development is under way to add visualisation of the salinity and temperature forcing files.

5. THE PHYSICS SUBMODEL

The Physics submodel deals with most physical and some biogeochemical processes in Atlantis, such as water and gas exchanges and mixing, nutrient inputs, deposition and resuspension of particulate matter, as well as some biological processes in the sediments, such as bioturbation and bioirrigation.

The central part of the Physics submodel is the **transport model** that reads in water exchanges from the hydrodynamic forcing files, applies a number of optional routines to correct for lack of horizontal and vertical mixing (see below), distributes water flows through the model domain, and calculates new concentrations of dissolved or passively advected tracers, such as salinity, temperature, oxygen, pH and pelagic planktonic organisms. These calculations simultaneously account for diffusion, advection and dispersion processes. The transport model is preceded by a number of optional routines executing sediment mixing, resuspension, gas exchange with the atmosphere, bioirrigation and others (see below).

Most of the details in the Atlantis Physics submodel are based on the Port Phillip Bay Integrated Model (PPBIM) described in Murray and Parslow (1997) and Walker (1997) and on the ERSEM model described in Ebenhöh et al. (1995) (see important references in Chapter 1.5). Atlantis users interested in understanding the physics processes deeper can learn more by consulting these references.

The parameters and flags used in the Physics submodel are described in:

1) *initial_conditions.nc*

2) *physics.prm, run.prm* and *force.prm* files

3) *hydrodynamics.nc* forcing input file

4) optional forcing NC and TS input files, such as temperature, salinity, pH, oxygen, contaminants, point source inputs, solar radiation, evaporation, precipitation and others.

In this chapter we will:

1) introduce physics and biogeochemical tracers and their properties,
2) describe the routines executed in the main *physics()* routine and parameters in the *physics.prm* file
3) explain why in some cases the user might need to correct for the lack of the vertical and horizontal mixing in hydrodynamic forcing files
4) give a brief introduction to the latest Atlantis developments allowing for the modelling of ice, land and contaminants
5) provide a small example demonstrating the importance of the optional physics routines

5.1. Physical and biogeochemical variables and their characteristics

The *initial_conditions.nc* file lists important characteristics of physical and biogeochemical tracers, used for various calculations in the *physics()* routine. Here is an example of a tracer description:

```
...
double Det_Si(t, b, z) ;
        Det_Si:bmtype = "tracer" ;
        Det_Si:units = "mg Si m-3" ;
        Det_Si:long_name = "Detrital Silica" ;
        Det_Si:dtype = 0 ;
        Det_Si:sumtype = 1 ;
        Det_Si:inwc = 1 ;
        Det_Si:insed = 1 ;
        Det_Si:decay = 0. ;
        Det_Si:dissol = 0 ;
        Det_Si:passive = 1 ;
        Det_Si:partic = 1 ;
        Det_Si:svel = -2.894e-005 ;
        Det_Si:xvel = 0. ;
        Det_Si:psize = 1.e-005 ;
        Det_Si:b_dens = 1000000000. ;
        Det_Si:i_conc = 200000000. ;
        Det_Si:f_conc = 200000000. ;
...
```

From this we can see that Det_Si is a three-dimentional variable (t,b,z), found in the water column (inwc=1) and the sediments (insed=1). It is not removed from the system through burial (decay=0), it is not dissolved in the water (dissol=0), but is present in a particulate form (partic=1). The sedimentation velocity of particles is svel=-2.894e-005, and its negative value shows that the particles sink (are not buoyant). The particle size is psize = 1.e-005 m

Table 6. Examples of currently used obligatory and optional physics and biogeochemical tracers and variables and their characteristics as defined in the *initial_conditions.nc* file (collected from different Atlantis applications and used for illustration purposes only). Further description of tracers and physics variables is given in the following chapters on sediment and biological processes.

Tracer	Name	In water	In sed	Dissol-ved	Parti-culate	Decay	Svel	Psize
***Obligatory tracers for dynamic properties**. These tracers participate in ecological processes and change as a result. It is very important to get the characteristics of these tracers correct in the initial_conditions.nc file as Atlantis will use them for dynamic calculations.*								
NH3	Ammonia	x	x	x		1.e-007	NA	NA
NO3	Nitrate	x	x	x		1.e-009	NA	NA
DON	Dissolved organic nitrogen	x	x	x		0	NA	NA
MicroNut	Micronutrients	x	x	x		0	NA	NA

Oxygen	Oxygen	x	x	x		0	NA	NA
Si	Dissolved silica	x	x	x		0	NA	NA
Det_Si	Detrital silica	x	x		x	0	-2.9e-005	1.0e-005

Optional tracers for dynamic properties. *Required only if*
[1] - trackAtomicRatio flag is set to 1 or [2] - flagIsEstuary flag is set to 1 in the run.prm file

Tracer	**Name**	**In water**	**In sed**	**Dissol-ved**	**Parti-culate**	**Decay**	**Svel**	**Psize**
P[1]	Phosphorus	x	x	x				
TOP[1]	Total organic phosphorus	x	x	x				
Tracer	**Name**	**In water**	**In sed**	**Dissol-ved**	**Parti-culate**	**Decay**	**Svel**	**Psize**
Carbon[1]	Carbon	x	x	x				
CO2[1]	Carbon dioxide	x	x	x				
SED[2]	Sediments	x	x		x	0	-2.0e-005	5.0e-006

Obligatory diagnostic tracers for physical properties and reporting of biogeochemical processes. *These tracers either describe pure physical properties (temperature, light, stress) or outcomes of other processes, e.g. chl_a concentration is a result of primary production and consumption. The properties of these tracers are hardwired in the Atlantis code and the values in the initial_condition.nc file are overwritten during simulations.*

Tracer	**Name**	**In water**	**In sed**	**Dissol-ved**	**Parti-culate**	**Decay**	**Svel**	**Psize**
Salt	Salinity	x	x	x		0	NA	NA
Temp	Temperature			x		0	NA	NA
Light	Light intensity on the surface of a cell			x		0	NA	NA
Stress	Surface Stress			x				
Denitrification	Denitrification			x				
Porosity	Sediment porosity							
Nitrification	Nitrification			x				
Chl_a	Chlorophyll			x				

Diagnostic tracers. *They are required in the initial_conditions.nc file and are used during debugging for reporting fine scale information about the group that is tagged by which_check parameter in the run.prm file*

Tracer	Name	In water	In sed	Dissol -ved	Parti- culate	Decay	Svel	Psiz e
DiagNGain	Gain in checked group							
DiagNLoss	Loss in checked group							
DiagNFlux	Flux in checked group							

5.2. The main *physics()* routine

The routine *physics()* defined in **atphysics.c** is the main physics routine, run at each timestep of the simulation. It is described below in the order of execution. The first column lists the routines, the flags to activate them, and shows which flags are active in the example SETas model. The second column describes routines and parameters. Remember that (as elsewhere in this manual) routines are listed in *italics() and* files containing these routines are bold and **underlined** and libraries are **bold**. Parameter names from *physics.prm* are given **in orange** and those from *force.prm* file are also marked with an asterisk *.

I. Execute explicit optional physics processes.

Add sources and sinks *sourceSink()* in **atsourcesink.c** injection 1	Applies forced inputs and outflows of water and nutrients from point sources, including evaporation and precipitation The number of point sources and sinks is given in npointss*. The inputs are read in as time series TS or NC files and they give details on inflow or outflow of water (m^3, temp, salinity) and nutrients (NH3, NO3, Labile Detritus, Si). They are also used to force nutrient inputs from river inflows, upwelling, waste disposal and other sources. Theoretically sinks of nutrients and water flows can also be included (negative values in the forcing files), but this has not typically been applied in practice. The *sourceSink()* routine also deals with water exchanges due to evaporation (Evaporation*) and precipitation (Precipitation*). If the injection flag is set to 0 then evaporation and precipitation will not be done at all. Note that evaporation and precipitation forcing files are provided as two-dimensional NC files, with the values given as millimetres of water input or lost on a xy coordinate grid. These inputs are automatically converted to the Atlantis box geometry by the model.

Add solar radiation *do_light()* in **light.c** solar_radiation 1	Add solar radiation energy to the surface of the model domain (Solar_radiation*). If forcing TS files of solar radiation are provided solar radiation intensity at each time step will be identical across the entire surface area of the model domain. This can be changed by setting lim_sun_hours to 1 in *biology.prm*, which will apply a set of NASA routines to calculate sun hours for each box given its longitude (using lat/long projections from the BGM file)
Exchange of gasses between the surface water layer and atmosphere *gasExchange()* in **atgas.c** atmospherics 1	Executes **temperature independent** gas exchange with the atmosphere across the water-air interface at the top of the surface layer. This calculates new concentrations of gases in the surface layer and it assumes an infinite source of gases in the atmosphere. Gas exchange with the surface water layer is governed by the diffusion coefficient, saturation value and stagnant layer thickness of the gas. These coefficients and gases to be used in the routine are hardwired in Atlantis code and defined in the **gaslist[]** data structure in the **atgas.c** library. Currently only O2 and CO2 are included. Note that *gasExchange()* is not the only routine dealing with input of gases from the atmosphere. There is now an option to include explicit deposition of nutrients and gases from the atmosphere, including O2 and CO2 (see description of include_atmosphere flag below). The include_atmosphere option deals with deposition of nutrients and gases trapped in particulate matter. It is handled in the specific routines defined in **atNutrient.c** library **atecology** - e.g. O2 is handed by *Oxygen_ROC()* routine. The deposition of O2 in *Oxygen_ROC()* is **temperature-specific**. **If both include_atmosphere and atmospherics flags are turned off there will be no O2 exchange with the atmosphere!**
Decay or removal of tracers from the water column and / or sediments. *decayBM()* in **atdecay.c** decay_wc decay_sed 0 0	Removes tracers from the system, simulating burial in the sediments. The exponent of the removal rate is given in the *initial_conditions.nc* file :decay parameter, so that removal rate is calculated as [tracer_concentration*exp(:decay*time)]. The removal rate in the sediments can be scaled to that in the water column with decay_sed_scale parameter to set as a multiplier on the standard :decay rate. If the :decay parameter of a tracer is set to 0 in the *initial_conditions.nc*, no removal of the tracer will occur even if the routine is activated.

	Note, the *decayBM()* routine implements a complete removal of the tracer from the system, not the actual decomposition decay. Even if the decay is deactivated a tracer can still break down through nitrification and denitrification processes. In this case however its nitrogen content may not be entirely removed from the system, but can be transformed into other tracers as specified by the nutrient cycling steps (e.g. organic N is transformed into NO3).
Resuspension of **particulate** tracers from the sediments into water column *resuspendBM()* in **atsuspension.c** resuspension 0	Calculates erosion of sediments due to shear (bottom) stress as a function of sediment type. Then calculates new concentrations of particulate tracers in the bottom water column layer after the resuspension. The rate of resuspension depends on the bottom stress, porosity of the sediment and background erosion rate. These parameters are given in the *initial_conditions.nc* file. The bottom stress can also be supplied as an external BottomStress* forcing file. To enable the routine, the user **must** provide non-zero stress values. This can be done either by giving non-zero values for the "Stress" tracer in the *initial_conditions.nc*; or, if NC Stress values are zero, Atlantis will look for an external forcing file. If no forcing file can be found, Atlantis will quit. The maximum rate for a 1cm sediment depth is set in the max_erosion (m in a single time step). The routine is particularly important when simulating wetlands (shallow ecosystems with many sediment layers).
Settling of **particulate** tracers from the water column *settleBMwc()* in **atsettle.c** settling 1	Calculates settling (**downward or upward!**) of **particulate** tracers from the water column depending on their settling velocity :svel parameter in initial_conditions.nc. **Only tracers with non-zero :svel and :psize will be acted upon**. This typically includes particulate biogeochemical tracers (Detrital Si) and some phyto- and zooplankton. If settling velocity is positive, particles are buoyant and rise up through the water column (done in the *advect_up()* routine). If settling velocity is negative, particles sink towards the bottom (done in the *advect_down()* routine). The flux of tracers from the lowest water column layer is added to the uppermost sediment layer if the tracer is allowed to exist in the sediments (:insed parameter in *initial_condition.nc*). The deposition to sediments is done in the *deposit()* routine in **atdeposition.c**.

	The deposition depends on the sediment's and tracer's porosity. The porosity of the tracers is calculated in the *calc_por()* routine in **atsedprops.c** using the :psize (particle size) and :b_dens (bulk density) parameters from the *initial_condition.nc* file. Sediment porosity is set as a "phys" variable in the *initial_conditions.nc* file. However, if the settling routine is turned on sediment porosity is calculated dynamically through the simulation in the *deposit()* routine. The sediment porosity is calculated as a ratio of the new water volume in the sediment, after deposition, and the total sediment volume, after deposition. If a particulate tracer is not allowed to exist in the sediment (:insed is 0 in *initial_conditions.nc* file), it is retained in the lowest water column layer. The *deposit()* routine uses the maxseddz and minseddz parameters to define the maximum and minimum depth of one sediment layer. If the maxseddz is exceeded everything extra will stay in the bottom water layer and no more deposition will occur.

II. Execute optional biological sediment processes – bioirrigation and bioturbation:

The bioirrigation and bioturbation routines simulate exchange of dissolved and particulate tracers respectively between sediment layers and between the bottom water column layer and the sediment. The baseline bioirrigation and bioturbation routines have been adopted from the PPBIM. However, Atlantis now also applies a multiplier to the baseline rate, which depends on the biomass of the living organisms in the sediment layers. The multiplier is calculated in the Biology submodel and will be described in the relevant chapter.

The bioturbation routine is only performed when more than one sediment layer is present! Because most models have only one sediment layer, the only way particulate tracers can be returned to the water column is through resuspension.

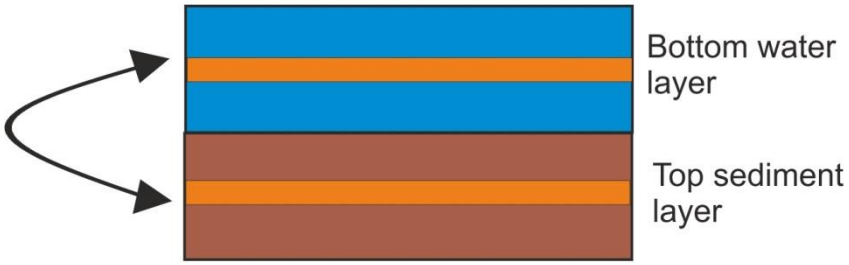
The full bioirrigation routine also requires more than one sediment layer, but it models simple exchange of dissolved particles even in models with only one sediment layer.

The processes used in Atlantis are similar to those in the PPBIM model, but are dynamically modified by the sediment fauna via an "enhancement" term (similar to that of ERSEM, Ebenhöh et al. 1995).

The "background density" of sediment organisms is set in the *initial_conditions.nc* file sedbiodens tracer and it can vary among boxes. Background density represents basic aerobic

and anaerobic diversity. This background amount is "enhanced" depending on the abundance of the dynamically modelled infauna and epifauna groups (identified with _INF and _EP names in the GroupType parameter of the *functional_group.csv* file). The abundance of dynamically modelled groups changes through time but the background density stays stable. The background biological activity remains present in sediments even if all dynamically modelled groups are wiped out (e.g. in anoxic sediments). Hence bioturbation, bioirrigation and other sediment processes can still occur (if activated).

Bioturbation and bioirrigation are simulated as exchange in thin water and sediment layers, as shown below. The routines take a small sliver of the water layer and sediment layer, swaps them and calculates new concentrations of dissolved and particulate tracers respectively. The thickness of the sliver to be exchanged reflects the intensity of the bioirrigation and bioturbation processes and depends on the amount of the biological activity in the sediment.



Bioirrigation: Inflow of water and **dissolved** tracers from the bottom water layer into sediments due to biological activity *bioirrig()* in **<u>atbioirrig.c</u>** bioirrigation 1	*bioirrig()* includes three subroutines (only the last two are done in models with one sediment layer): *dissol_diff()* calculates diffusion of dissolved tracers across the sediment layers, **if more than one sediment layer is present**. This is calculated as a product of: - bi_dissol_kz diffusion constant at zero depth (m^2s^{-1} per animal per m^2), where the amount of animals per m^2 is set in the **sedbiodens** tracer in the ***initial_conditions.nc*** file. - density of biological activity of _INF and _EP groups - scalar of biological activity as a function of depth (which depends on the profile described in the biosedprofile) - scalar of irrigation enhancement due to oxygen concentration *exchange()* applies the inflow of water and dissolved tracers between bottom water layer and sediments, depending on the concentration of the tracer. This can be seen as a passive exchange of dissolved tracers depending on their concentrations. It is calculated as a product of

	- bi_exchange constant (ms^{-1} per animal per m^2) - area of the sediment/water interface - density of biological activity - scalar of irrigation enhancement *injection()* does injection of water and dissolved tracers form bottom water layer into sediments, simulating active pumping of water by animals moving in burrows - bi_injection constant at zero depth (m^2s^{-1} per animal per m^2) - area of the sediment/water interface - density of biological activity - scalar of irrigation enhancement The profile of biological activity with depth is set by the biosedprofile parameter, which can take four functional forms: constant, linear, parabolic, Gaussian
Bioturbation: movement of **particulate** tracers throughout the sediment layers and into the water column due to biological activity. **This routine works only if more than one sediment layer is present** *bioturb()* in **atbioturb.c** bioturbation 1	Only **particulate** tracers that are not macrobenthos and are allowed in the sediments, i.e. sediment grains, settled phytoplankton, microphytobenthos, meiobenthos, detritus and sediment bacteria, can be acted upon by bioturbation. The *bioturb()* routine includes three subroutines (like the *bioirrig()* routine described above): *partic_diff()* implements particulate tracer diffusion within the sediments and is implemented in the same way as diffusion of dissolved tracers in the *dissol_diff()* subroutine described above. The diffusion rate is calculated as a product of: - the diffusion constant bt_partic_kz (m^2s^{-1} per animal per m^2) - density of biological activity - scalar of biological activity as a function of depth (which depends on the profile described in the biosedprofile) - scalar of bioturbation enhancement *expul()* applies expulsion of particulate tracers from sediments into the bottom water column. Like above, its constant is set by the bt_expulsion parameter (ms^{-1} per animal per m^2), which is modified by the biomass density, scaling by depth and the bioturbation enhancement scalar. *mix_in()* applies exchange of particulate matter across sediment layers. The exchange constant is set in bt_exchange (ms^{-1} per animal

	per m^2) which is modified by biomass density, scaling by depth, and bioturbation enhancement scalar

III. Execute optional routines to enhance horizontal and vertical diffusion and mixing (see chapter 5.4 for why these routines may be needed):

Horizontal diffusion *hdiffBMwc()* in **athdiff.c** horiz_diffusion 0	**Horizontal diffusion is done before the main transport model is run. This was mostly needed in the past, when oceanographic models and their conversion to Atlantis forcing files did not provide sufficient horizontal mixing.**
Filler horizontal diffusion **applied for boxes that have not had any water exchanges for a set period of time** *filler_hdiffBMwc()* in **athdiff.c** fill_zero_exchange 0	Hydrodynamic models don't always resolve coastlines well, which can lead to some small coastal boxes being left isolated (no water fluxes into or out of them). This optional routine is used to correct for this. It is activated if there is no flux for a set threshold of days, provided in the flush_threshold (often set to 1 day). The routine will force mixing between the isolated box and matching layers of the adjacent boxes. The routine will apply a constant water column mixing coefficient wc_kz, which indicates the proportion of volume to be mixed (a reasonable value is within a range of 1e-9) and scale it by the relative box volumes. The horizontal mixing can be further tuned using box-specific horizontal mixing parameter boxID:horizmix from the BGM file. To activate this box specific mixing set use_fill_horizmix to 1. Otherwise the baseline diffusion rate among boxes is used.
Vertical diffusion added to correct for the lack of vertical circulation (this is not the same as vertical mixing used to imitate eddies, see below) *atvdiffBMwc()* in **atvdiff.c**	Vertical diffusion executed throughout the water layers before the main transport model. Setting flag mix_deep to 1 activates the replenishment of nutrients in the deep layers of oceanic boxes (those that have open water boundary with deep waters) and set the depth value for deep ocean layers in mix_deep_depth parameter. Nutrient values in the bottom layer of boxes that are deeper than mix_deep_depth will be reset to the original value given in the *initial_conditions.nc*

vert_diffusion 0	The nutrients that will be reset are: NH3, NO3, Si, MicroNut and also Phosphorus and Carbon if they are tracked throughout the simulations (trackAtomicRatio flag is set to 1 in the *run.prm*)
Vertical mixing added to simulate upwelling and increased mixing by eddies *vertical_mixing()* in **atvmix.c** vert_mix 1	Because processes within one cell are considered uniform, explicit representation of eddies and upwelling would require a lot of small boxes, untenable for Atlantis models. Instead the supply of nutrients from deeper water levels by eddies and upwelling can be represented with this optional vertical mixing routine. The routine applies season and box specific vertical water column mixing, in addition to the vertical exchange forced through hydrodynamic files. The main difference between vertical diffusion routine above and vertical mixing is that diffusion is typically a slower process of movement from high to low concentration rates. Mixing in turn simulates advection (movement by water currents) and is usually much faster than diffusion. Also vertical mixing can be customised to vary across boxes and seasons. The total mixing rate (in m^3s^{-1}) is calculated from lower layer up as a product of - mix_injection coefficient (1e-3) - Box specific :vertmix scalar defined in the BGM file - seasonal scale - eddy scale - volume of the box Note that the sign of the global mix_injection and box-specific :vertmix parameters defines the **direction of vertical mixing**. Usually these coefficients are positive, which means the water will flow upwards, but in theory they can also be negative. Either way, **in vertical mixing routine water flows only in one vertical direction (up or down)**. The seasonal scale is modelled as a sine shaped wave (sin(2.0*3.1415*(day_of_the_year-31.0)/365.0) and it's fluctuations can be increased by setting the mix_season_kz multiplier (set to 10 in SETas model). The effect of eddies (eddy scale) is activated with eddy_vertmix flag. It can be tuned with two parameters. First, box-specific eddy strengths per each quarter of the year (the model linearly interpolates between these to get eddy strength on any one day) are set with eddy_S1 to eddy_S4. These eddy parameters are used also in scaling *Primary_Production()* in **atprocess.c** in the **atecology** library. Therefore, to allow for different effect on primary

	production and vertical mixing by eddies, the vertical mixing has specific parameter eddy_vertmix. Vertical mixing can further be further scaled with the global scaling coefficient eddy_mixscale As in the vertical diffusion routine above, activating the flag mix_deep will reset nutrient concentrations in the deepest layers of oceanic boxes

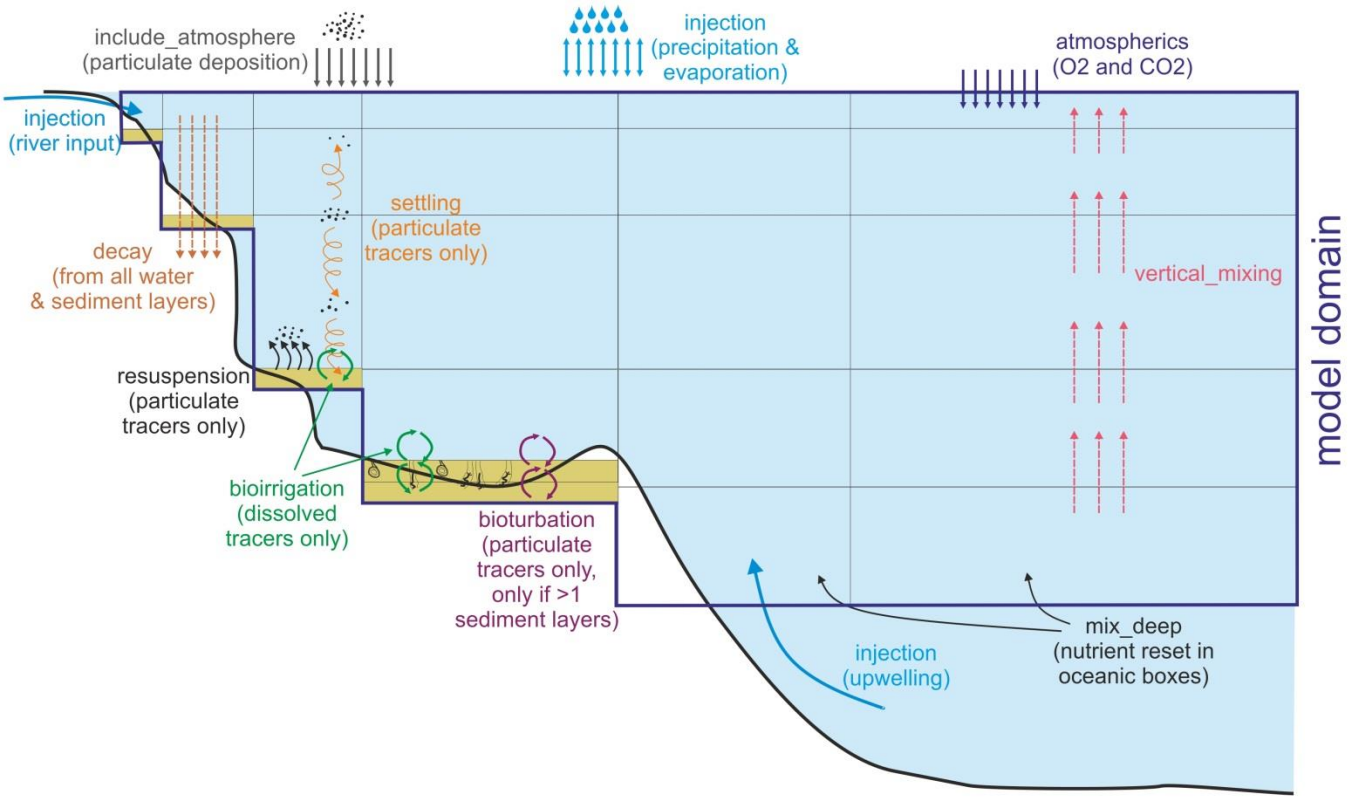


Figure 8. Illustration of optional physics routines described in detail above. Some boxes have two sediment layers. Vertical mixing can also be set as downwelling, but only one direction is possible in a given simulation.

IV. Execute the main transport model

Although there is an option to turn the transport model off (advect_diffusion 0) this should **only be done for debugging purposes** as it **stops all water fluxes** in the model domain. Also note that if you turn off the transport model this will stop the the t.hd timestep from incrementing. The timestep is the timestep from the hydro model. It is also used to read in data from the point source input files. So if you turn off the transport model the values read in from the point source files will always be the value corresponding to the first hydro timestep (the first time value in the first hydro forcing file).

The transport model has an option to scale all transport forced from hydrodynamic input file. This is set by setting scale_transport flag to 1 or 2.

If scale_transport is set to 1, the transport will be scaled by prcnt_exchange parameter. For example, setting prcnt_exchange to 0.5 will half all the horizontal and vertical transport, in some older models this was needed to help correct for hyperdifussion.

If scale_transport flag is set to 2, all transport is scaled by the area of the box and ka_exchange multiplier. The area corrected exchanges are multiplied by ka_exchange before they are used (transport_scalar = ka_exchange / box_area). This helps to ensure the transport is not scaled down too much.

NOTE! Do not scale transport by box area if hyperdifussion correction by box area has already been done during the forcing file construction stage, such as in the Hydroconstruct package (see above).

Main (advective) transport model *transportBM()* in **attransport.c** advect_diffusion 1	This routine executes all forced water fluxes (moves water across cells) and calculates new concentrations of tracers. Transport model also has a cascade_flows flag which executes transport among boxes with large differences in vertical layers. If cascade_flows is set to 0 the loop calculating transport will only go through the number of water columns in the existing box, if it is set to 1 it will go through the number of water columns in both of the exchanging boxes. The *transportBM()* routine calculates new concentrations of tracers, including temperature, salinity and pH. These calculations are done separately in the *Properties_At_Depth()* routine in **atbiophysics.c** in the **atecology** library. The calculated values are then overwritten if temperature, salinity, pH or other forcing files are provided (see below). The temperature calculations use baseline_temp and temp_ampltiude parameter and will be described in chapters dealing with Biology submodel.

V. Read in temperature, salinity and other tracer forcing values and overwrite values calculated in the transport model

Read in forcing files for temperature, salinity, pH or other forced	The calculations of tracer values done in the *transportBM()* routine are overwritten if forced values are provided and appropriate flags activated in the *force.prm* file

tracers and overwrite calculated values *tracerForcingBM()* in **attempsalt.c** use_tempfiles* 1 use_saltfiles* 1 use_pHfiles* 0 use_force_tracers* 0	Typically, Atlantis models include temperature and salinity forcing (Table 5), but also pH, oxygen and other tracers set in the *initial_conditions.nc* can be forced. Such forcing is done if calculations of tracer concentrations done in Atlantis are deemed to be of insufficient accuracy and better information is available from more specialised models or observations (see tracer forcing on the wiki) The number and names of all tracer forcing files must be listed in the *force.prm* parameters nforceTracers* and tracerNames*, eg: nforceTracers 2 tracerNames 2 Oxygen Light This is followed by the actual files and parameter indicating whether the tracer values should be looped (rewound) or not. Oxygen_nFiles 1 Oxygen_File0.name inputs/forcisets/oxygen.nc Oxygen_rewind 1 (see instructions also on the wiki). Note, that the names of tracers provided in the tracerNames MUST exactly match the names in the *initial_conditions.nc* file, so that Atlantis knows which tracers to overwrite. **NOTE! Forcing of tracers, especially those containing nitrogen, should be done with care, as it can easily affect the conservation of mass in the model**

VI. Do final corrections and scaling

Apply temperature, salinity and pH scalars to the biological processes *Ecology_Apply_Environ_Scalars()* from **atbiophyscis.c** in **atecology**	Runs optional scaling of biological processes by environmental conditions, such as temperature-, oxygen-, salinity-, pH-dependent feeding rates, mortality or other biological processes. This will be described further in the Biology routines
Update eddy values	

Eddy_Strength_Update() routine in **atphysics.c**	Conducts linear interpolation of eddy strengths for the current day of the year from the quarterly box specific values given in eddy_S1, eddy_S2, eddy_S3 and eddy_S4.
Checks saturation values *Saturation_Check()* in **atsaturation.c**	Checks that gases and nutrients haven't surpassed saturation values. If saturation values are exceeded, then concentrations are reset to the maximum saturation concentrations.
Get ice related information from input forcing files *Get_Ice()* in **aticeIO.c**	Reads in ice properties for the current time step from TS or NC forcing files Only executed if the *initial_conditions.nc* file has the global attribute icenz (see below and details on wiki), which activates the ice model.
Update time for writing outputs	Update the next time to write the values into *inputs.ts* and *exports.ts* files (see Table 3 on the contents of these files)

5.3. Other parameters in *physics.prm* not included in the *physics()* routine

Below are the remaining parameters and physics processes that were not described in the main *physics()* routine.

I. Resetting water column depths

wc_dz_tol: This parameter is used by *check_wc_dz()* routine in the **atvertgeom.c** file of the **atphysics** library. The routine checks the depth of water layers. Transport of water fluxes can in some cases cause too much flux into certain layers. This can be controlled by setting the maximum change in depth allowed in the wc_dz_tol parameter. The model will only allow this much fractional change before dz is reset to nominal value given in the *initial_conditions.nc* file by the nominal_dz variable. The wc_dz_tol parameter in many models is set to 0.2.

constrain_wc: This is a flag that is called in the *box_bio_process()* routine in **atbiology.c** and it resets the minimum water column depth to 1m if it becomes any shallower due to water fluxes. Turning this flag on is recommended in models that have a lot of shallow water areas and do not model sediment processes through multiple sediment layers explicitly. This avoids difficulties in modelling nutrient fluxes in very shallow waters.

II. Checks for the depth of sediment layer(s)

Same checks are done for the depth of sediment layers, which are controlled by maxseddz and minseddz parameters. The depth of the sediment layer(s) is controlled by the *check_sed_dz()* routine in

atvertgeom.c. If the sediment layer depth gets too small the layer is merged with an adjacent sediment layer. If the sediment layer depth becomes too large it is split in two (made up of the original layer and a new inserted sediment layer). **This is only used in models that have >1 sediment layer!** In models with only one sediment layer, there are no constraints on the thickness of the sediment layer.

III. Atmospheric deposition of gases and nutrients trapped in particulate matter

Atlantis has always allowed for atmospheric deposition of NH and NO, but this new functionality makes this explicit and allows users to modify atmospheric concentrations (mg per m^3) of nutrients and gases trapped in particulate matter. Note, **this is not the diffusion of gases through the water's surface**, which is dealt with in the *gasExchange()* routine. The atmospheric deposition is handled in respective nutrient routines in the **atNutrient.c** file in **atecology** library. To apply the atmospheric deposition, set include_atmosphere flag to 1 in the *physics.prm*. If the flag is set to 0 there will be no implicit deposition from the atmosphere.

If include_atmosphere is active then atmospheric concentrations (mg per m^3) must be provided for the following seven nutrients and gases (example values given here are taken from the SETas model)

atmospheric_NH 0.017
atmospheric_NO 0.025
atmospheric_F 0.0
atmospheric_P 0.0
atmospheric_Si 0.0
atmospheric_O2 198.9
atmospheric_CO2 0.0

IV. Scaling of point sources

It is possible to scale the input of nutrients from point sources given in the TS files. This scaling might be useful if simulating increase or decrease of nutrient input through time. The scaling routines are activated with the flag nutrientchange. If the flag is set to 1, then a number of additional parameters should be provided, telling Atlantis how to scale the nutrients. The parameters required for scaling allow detailed scaling of each nutrient in each point source and are explained further on the wiki.

V. Other parameters from old functionality

vdiffwt_wc: sets the scalar of water column diffusion and is used in the optional and currently rarely activated vertical and horizontal diffusion routines.

vdiffwt_sed: sets the scalar of diffusion in the sediments. It is used in the bioirrigation routine.

edge_type: a string of box-specific parameters describing the way box edges deal with water fluxes (0=standard, 1=absorptive, 2=reflective, 3=assymetrical scaling). This was used in the early days of Atlantis, when oceanographic files were not as sophisticated as they currently are and transport of tracers had to be manually adjusted. Setting different edge types for different boxes scaled the

transport of tracers into and from the boxes. For current models, it is recommended to set the edge_type to 0 (standard) for all boxes.

biooxprofile: This was used in earlier Atlantis versions when the oxygen profile in the sediments was set and not dynamic. Currently the amount of oxygen in the sediments dynamically depends on the amount of biological activity and the parameter is not used.

5.4. When to use vertical and horizontal transport and mixing?

The horizontal and vertical diffusion routines *hdiffBMwc()* and *atvdiffBMwc()* were created and used most often at a time when oceanographic models were still quite coarse and did not capture the water fluxes with sufficient accuracy. They are rarely used in current models. However, a number of issues still remain due to the loss of spatial resolution when converting files into the Atlantis model domain. This is where activating optional vertical mixing and filler horizontal diffusion routines might be useful:

1) **Eddies**. The spatial scale of most Atlantis models is too coarse for explicitly resolving oceanic eddies. Yet, eddies have a strong influence on productivity and oxygen content and ideally should be included in some form. The effect of eddies can be captured in two non-exclusive ways:

a) Eddy parameters (eddy_S1 to eddy_S4 and eddy_mixscale) parameters are used in the *Primary_Production()* routine to scale production. If some boxes are known to have strong eddies that amplify local production, the user can modify box- and season-specific eddy parameters to enhance production in certain boxes and certain seasons. The presence of eddies will scale the growth rate of primary producers, so that for the same nutrient concentration more growth is allowed (but of course, no growth is possible if no nutrients are available).

b) Eddy effect on productivity can be simulated by activating the *vertical_mixing()* routine, where mixing will be enhanced with box- and season-specific eddy scalars. Increased vertical mixing will bring more nutrients to upper water levels and will increase productivity in boxes identified through seasonal eddy_S1 to eddy_S4 values (which can differ box to box, showing relative eddy strength). Differences in vertical mixing intensity among boxes can be further tuned using the box-specific :vertmix parameter in the BGM file.

Good practice tip 4

Eddies are not currently included in most models, but exploring their potential effect on the model outcomes is recommended. They provide a useful approach for adding spatial realism in primary productivity dynamics and may be particularly important in deeper waters and in systems where eddy effects are known to be strong.

2) **Upwelling**. The importance of upwelling for primary productivity is well recognised, but it is often difficult to capture the intensity of vertical mixing and nutrient input through the hydrodynamic forcing inputs alone. The latest oceanographic models are getting better in capturing upwelling through vertical fluxes, but in some cases Atlantis modellers might want to add an additional forced time series of point nutrient source into boxes where nutrient supply due to upwelling is not

accurately captured. Upwelling would normally only be added into oceanic boxes, i.e. those that do not end with the sediment but with the open deep water boundary. The mixing of the forced nutrient input into the water column might have to be increased using the *vertical_mixing()* routine and by modifying box specific :vertmix parameter in the BGM file (or by using eddy_S1 to eddy_S4 parameters), to ensure the additional nutrients get properly mixed into the water columns.

Note, that the replenishment of nutrients in the deepest water column layers of the oceanic boxes may not require forced nutrient inputs, as it can be set with the mix_deep flag (see the description of the vertical diffusion routine above) and by normal vertical mixing. Yet, activating the mix_deep will only reset the nutrient concentrations in the deepest water layer to the nominal (initial) values given in the *initial_conditions.nc*. Including a separate input forcing file allows the user to specify time variable nutrient input throughout the year.

3) **Absence of fluxes in small coastal boxes**. Many oceanographic models, unless they were specifically built, still do not capture coastal processes with sufficient detail. This, combined with the loss of spatial resolution, can lead to isolated coastal boxes that receive no water fluxes (especially if the boxes are small or particularly shallow). This would naturally affect the model outcomes in such boxes, leading to high accumulation or depletion of nutrients and biomass.

Ideally, such issues should be resolved in cooperation with oceanographers who developed the models. If this is not possible, the filler horizontal diffusion routine will provide a very crude approximation of mixing for such boxes. Mixing can be further tuned with box specific mixing scalars (:vertmix and :hozirmix) in the BGM file.

Good practice tip 5

The exploration of vertical and horizontal water fluxes is strongly recommended, especially during the model development stage. Currently there are two complementary ways to do it:

1. Use a ***shinyrAtlantis*** R package (see chapter 4.4) to plot fluxes through time and connections among boxes. The tool helps to identify isolated boxes, visualise intensity and direction of fluxes among boxes and also plot the magnitude of vertical fluxes in each box. The user can then compare these fluxes with the existing oceanographic knowledge and decide whether the intensity of vertical mixing should be increased or decreased using box-specific parameters.

2. Use passive inert tracers, such as sediments (SED in the *initial_conditions.nc*) to explore dispersal of particles through the model domain. Sediments are not acted upon by any organisms, so they won't be eaten or decay (make sure :decay is 0). The initial concentrations of the tracer can be set in specific locations in the *initial_conditions.nc* file and their dispersal through the model domain followed in the model output. Alternatively, sediment inputs can be given in the TS files as forced point sources; and their dispersal from the point input locations can then be followed through time.

5.5. Latest developments

New functionality is added to Atlantis almost every month. At the time of writing of this manual (April 2016) three significant new features were being tested: land, ice and contaminants. They are described

here briefly, but the users are advised to check Atlantis wiki for further developments and instructions.

5.5.1 Land

It is now possible to include land in Atlantis. The key reason to include this functionality is to allow explicit modelling of the movement of nutrients, organisms and human activity between land and water (groups can only make this transition if they are marked as being able to live on land and in the water in the *functional_group.csv* file). It also allows for the future development of Atlantis into an integrated water-land model (so you can model land organisms). Currently this may be useful for shallow water areas, harbours and estuaries that have a lot of islands.

The optional land model is turned on by activating the flag flagAllowLand in the *run.prm* file. Land boxes are set up in the BGM file as having positive botz value. Land occupies water column layer 0 (bottom layer) and processes on land boxes are modelled in two-dimensional space, just like those for epibenthos. To obtain the total biomass, tracer values on land boxes are not multiplied by depth. It is also possible to include specific terrestrial tracers, by setting their bmtype = "terrestrial" in the *initial_conditions.nc* file. The values for these tracers would be stored in a different place than those for the water column tracers.

5.5.2. Ice

Atlantis applications for the northernmost and southernmost domains (Barents Sea, Baltic Sea, Antarctica and others) have prompted the development of an explicit ice model.

The optional ice model is activated when the *initial_conditions.nc* file has the global icenz variable included. This sets up the maximum number of ice layers. The ice model requires ice forcing files and their paths need to be identified in the *force.prm* (nInceFiles parameter in *force.prm*)

The key characteristics of ice are described in the wiki along with the additional parameters added to the *physics.prm* - these include maximum and minimum depth of an ice layer (maxicedz and minicedz), the number of ice classes, presence of slush, and specification of the ice model to use. The biological groups that are allowed in the ice include bacteria, diatoms, mixotrophs and small zooplankton. Activating the ice model requires a number of new parameters in the *biology.prm* file too. These describe how to handle albedo, light attenuation, the depth consumers can dig into ice (the ice penetration depth of predators).

5.5.3. Contaminants

It is now possible to track an unlimited number of contaminants through the food chain. However, each contaminant requires specific new tracers tracking contaminant concentrations in each functional group (and each age group of fully age structured functional groups), which means significant changes to the input files and longer simulation run times. Each contaminant in the *initial_conditions.nc* is setup as a separate tracer (eg, Arsenic). Because contaminants are tracked through all functional groups, a new contaminant tracer must be added for all tracked functional groups, e.g. Sed_Bact_Arsenic, Zoo_Arsenic, Cod1_Arsenic, Cod2_Arsenic, Cod3_Arsenic and so on.

The functional groups take up contaminants either through contact, general uptake or through consumption. For each functional group the user will have to provide a number of new parameters in the *biology.prm* file: the uptake shape (linear or sigmoidal), uptake rate (mgm-^{3} per second), 100% and 50% lethal concentration, and time to reach 50% mortality. It is also possible to include an additional contaminant effect on growth.

To set up contaminant tracking set the track_contaminants flag to 1 in the *run.prm* file and provide contaminant names and forcing files (see instructions on the wiki).

5.6. Outputs of physics tracers in the *output.nc* file

Values of physical variables throughout the simulation are reported in the main *output.nc* file. Although not usually analysed in standard simulations, it might be useful to explore these values during the model development stage.

Table 7. List and characteristics of physics variables given in the *output.nc* files (the same variables are given in the *initial_conditions.nc* file but without the time dimension).

Name	Description, units	Comments
volume (t, b, z)	Volume of each cell at each time step, m^3	The volume changes due to water fluxes that simulate currents and tides. The change in the box's volume is seen only as change in the depth and not surface area of the water column. The maximum fractional change in the depth is set by wc_dz_tol, if the depth exceeds this the depth is reset to the nominal_dz values given in the *initial_conditions.nc*. At present big tides will cause this reset, a flag to disable this is under development so please check the wiki for any updates.
hdsource (t, b, z)	Hydrodynamic sources, 1	A number of fluxes (number connections) into a given box – from other boxes and from itself
hdsink (t, b, z)	Hydrodynamic sinks, 1	A number of fluxes (number connections) from a given box – to other boxes and to itself
eflux (t, b, z)	Hydrodynamic net horizontal exchanges, 1	Total inflow or outflow (positive/negative value) of water into the given cell in the horizontal direction
vflux (t, b, z)		Total inflow or outflow (positive/negative value) of water into the given cell in the vertical direction

Name	Description, units	Comments
	Hydrodynamic net vertical exchanges, 1	
porosity (t, b, z)	Porosity of the sediment layer, 1	Fractional porosity of the sediment (from 0 to 1). Initial values are set in the *initial_conditions.nc*, but the property is dynamically calculated after (optional) settling of particulate tracers
nominal_dz, (b, z)	Thickness of a water layer, m	Initial or nominal depth of water column and sediment layers in each box. This is **not a dynamic variable**, as it does not have the time dimension. For all but the bottom water column layers the nominal depth is usually a round number (1 to a few hundred meters). The depth of the bottom water column layer varies depending on the terrain of the actual ecosystem (see Fig. 4).
numlayers (t, b)	Number of layers	Number of layers in each box as set by the user. The number of layers is set in the *initial_conditions.nc* and currently remains stable through time. New functionality in the future might allow the number of water column layers to change due to large tides.
dz (t, b, z)	Thickness of a water layer, m	Dynamic thickness of the water layer, reflecting changes in its volume due to water fluxes (see volume variable above)
topk (t, b)	Sediment top index	The sediment layer that has the interface with the water. In current models this is always 0, but the variable was used in the PPBIM when multiple dynamic sediment layers were modelled
sedbiodepth (t, b)	Depth of biological activity, m	Maximum depth of biological activity in the sediments. The shape of the profile of biological activity with depth is set in the biosedprofile in *physics.prm*. The actual depth of the activity will change depending on the amount of activity present and the oxygen levels
seddetdepth (t, b)	Max depth of detritus, m	Depth of detritus in the sediment layer. It determines the depth of the habitat suitable for the infauna and

Name	Description, units	Comments
		changes through time due to sediment processes. It is dynamically calculated in *Bio_Box_Process()* routine in **atbiology.c**.
sedoxdepth (t, b)	Depth of oxygen horizon, m	Depth of the oxygenated sediment layer. The initial values are set in the *intial_condition.nc*, but the actual oxygen layer depth is dynamic and depends on the amount of oxygen in the water column and sediments, and amount of biological activity.
sedbiodens (t, b)	Biological activity, Animals m^2	The "background density" of biological activity that is always present in the sediment. It is set by the user and can vary among boxes to reflect different quality of the sediment (see description of bioirrigation and bioturbation in chapter 5.2). Even though this variable has time dimension, it does not currently change through time
sedirrigenh (t, b)	Bioirrigation enhancement, 1	Scalar of bioirrigation enhancement due to action of infauna species (marked with _INF in the GroupType of the *functional_group.csv* file). It is calculated in the *Irrig_and_Turb()* routine in **atprocess.c** in **atecology** library
sedturbenh (t, b)	Bioturbation enhancement, 1	Scalar of bioturbation enhancement due to action of infauna species (marked with _INF in the GroupType of the *functional_group.csv* file). It is calculated in the *Irrig_and_Turb()* routine in **atprocess.c** in **atecology** library
erosion_rate (t, b)	Erosion rate m/s	Dynamic erosion rate variable, which depends on the supplied bottom stress values and the topk variable
eddy (t, b)	Eddy strength, 1	Eddy strength in each box at each time step, depending on the eddy seasonal values and scalars (see vertical mixing routine description in chapter 5.2)

5.7. Importance of optional physics routines

During the model development stage many people focus their attention on the parameters defining biological routines, while the *physics.prm* file is rarely touched. However, a number of optional routines included in the Physics submodel may have a large effect on the nutrient dynamics – bioirrigation, bioturbation, settling of particulate matter. This in turn is likely to have a large effect on biological groups and especially lower trophic levels. The example below is a quick analysis on the effect of turning some Physics routines off on total biomasses of functional groups in a Baltic Sea model. The model is still under development and is not presented in detail here.

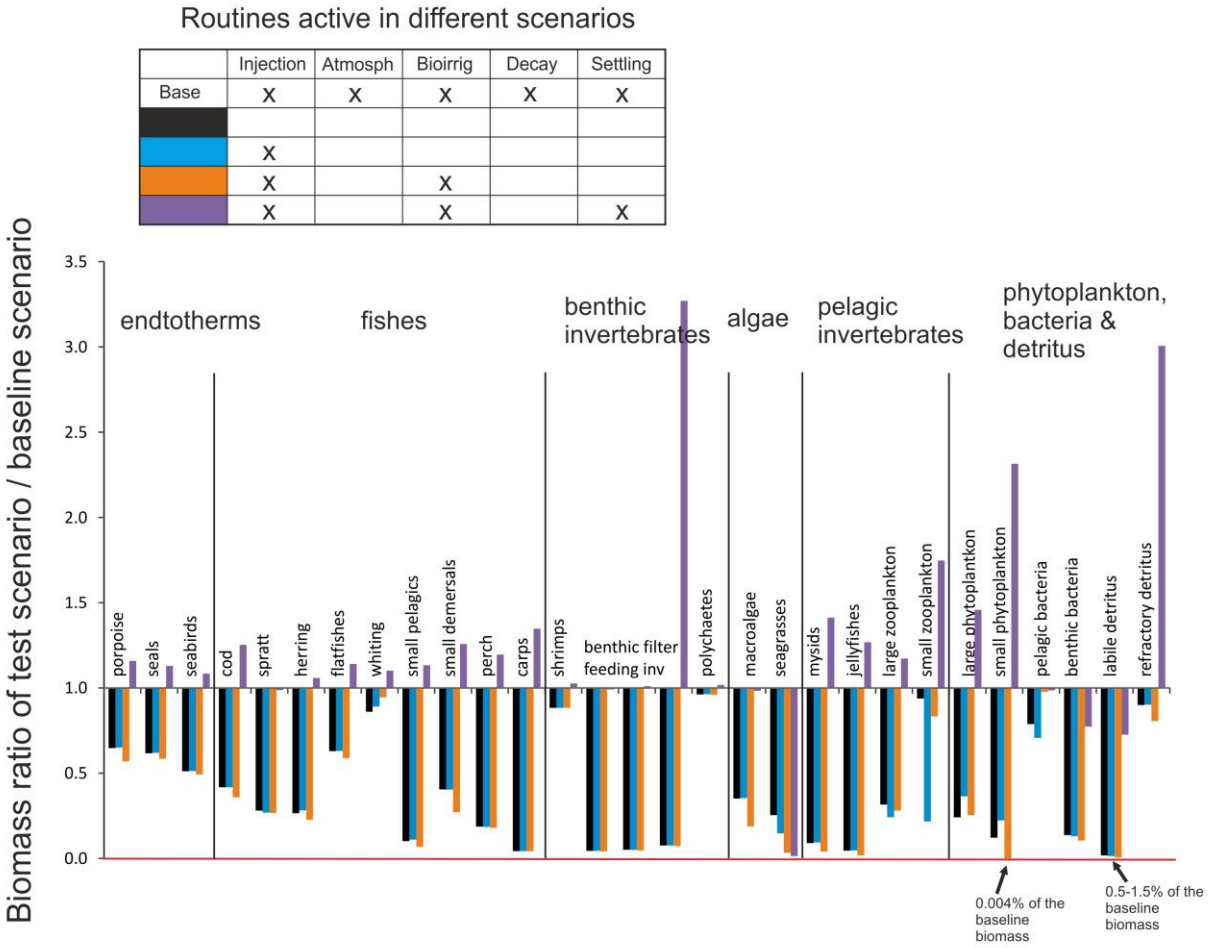


Figure 9. Total final biomass ratios of all Baltic Sea model functional groups in scenarios exploring the effects of optional physics routines. The simulations were run for 20 years.

Separate analysis was also done where six parameters in the *physics.prm* file defining bioturbation and bioirrigation intensity (typically in the range of 1e-7 to 1e-8) were increased 1000 times. The result is not shown but the effect was very minor, for most groups it made less than 5% difference in the final biomass compared to the scenario using same the physics routines but with default parameter values. Note, that the bioturbation routine is not included here, because the Baltic Sea model has only one

sediment layer and hence the routine cannot be performed (setting bioturbation flag to 1 or 0 did not have any effect on biomasses).

Good practice tip 6

It is recommended that while developing the model, users conduct a range of simulations with some of the optional routines turned on or off and explore different ways that physical and biological process might affect nutrient turnover.

6. DEFINITIONS OF FUNCTIONAL GROUPS

6.1. Introduction to functional groups

Types of functional groups

Biological components are represented as **biomass pools** (most non-vertebrates), **age-structured biomass pools** (some invertebrates) or **age-structured groups** (vertebrates). These three group types are assigned according to the GroupType and NumCohorts parameters given in the *functional_group.csv* file (Table 6.1). Group types FISH, FISH_INVERT, SHARK, MAMMAL, BIRD, REPTILE are assigned to the fully age-structured groups with the number of age groups given by NumCohorts. Other groups are assigned to biomass pools. If NumCohorts for a biomass pool is >1 they will be assigned to an age-structured biomass pool. The age-structured biomass pools are similar to stanzas used in, for example, Ecosim and are used to allow for different juvenile and adult behaviour and parameters. All non-vertebrates that have NumCohorts=1 are assigned to simple biomass pools.

NOTE!
Can a non-vertebrate be modelled as an age-structured group?

Six GroupTypes (FISH, FISH_INVERT, MAMMAL, SHARK, BIRD and REPTILE) are currently modelled as age-structured groups. If full age-structure is needed for a non-vertebrate, it should be set as one of these groups, most likely as FISH_INVERT or FISH. FISH_INVERT currently uses the same code as FISH, but the intent is to use it to represent a size rather than age-based model, which is a more typical representation for key invertebrates (such as exploited species). Note however that setting a group as FISH_INVERT or FISH would not call typical non-vertebrate routines (see below).

Atlantis does not explicitly model sex, but represents an average individual. This means that all individuals in a biomass pool and an age cohort (or a genotype in a cohort, if multiple genotypes are modelled) are considered identical in terms of their reproductive output. This may require some caution in parameterisation, e.g. number of offspring per mammal individual.

State variables tracked

Main model

In models that are based on N only (i.e. do not explicitly track P and C) the only state variable tracked for most biomass pools is N, but both N and Si fluxes are tracked for Si limited biomass pools, such as diatoms. The Si limited groups are identified from the *functional_group.csv* file parameter IsSiliconDep (see below). In age-structured groups three variables are tracked for each age cohort: reserve N (RN), structural N (SN) and numbers (Num). In addition, fluxes in the water column and sediments are tracked through eight **inanimate pools**: ammonia (NH), nitrate (NO), dissolved silica (Si), detrital silica (DSi), dissolved oxygen (O2), dissolved organic nitrogen (DON), labile detritus (DL), and refractory detritus (DR). An additional inanimate pool, carrion (DC), is also required, though it is only dynamically used if the fisheries submodel is active; DC is then used to represent fisheries discards. State variables tracked for each group defined with the GroupType parameter in the *functional_group.csv* file are listed in Table 6.1.

Extended models

If the model also includes tracking of P and C (trackAtomicRatio is set to 1 in *run.prm*), then P and C pools are tracked for each functional group, together with N pools. Additional inanimate pools are for C and P are also required.
If the model is of an estuary (FlagIsEstuary=1 in *run.prm* file) a sediment (SED) inanimate pool is added for the water column. If the model tracks contaminants (track_contaminants=1 in *run.prm*) a separate pool is added for each contaminant for each nitrogen pool tracked, so that concentration of contaminants can be tracked throughout the entire food web.

Allocation of pools throughout the vertical dimension: water column, epibenthos and sediment

Atlantis models 3-5 explicit distribution types of organisms and inanimate pools through the model domain – **water column**, **epibenthic layer** and **sediments** are tracked in all models and **ice** and **land** may also optionally be included. The water column and **sediment** are modelled as three dimensional spaces (units of variables are in m^3), whereas the epibenthic layer is two-dimensional (units in m^2). Most organisms are restricted to one type, such as sediments (infaunal invertebrates identified with INF in GroupType name), epibenthic layer (epibenthic organisms, identified with EP in the GroupType name) or water column (fully age-structured groups). Some of the smaller groups (e.g. microflora primary producers) and the inanimate pools can exist in both the sediments and water column (Table 6.1).
In the Atlantis code the water column, epibenthic, sediment, ice and land distributionns are referred to as "habitatType". However, remember that they are not the same as the ecological epibenthic habitats modelled in Atlantis (e.g. seagrass, corals etc.).

Table 6.1. State variables tracked in the main Atlantis model (no ice or land included). * indicates optional Si tracking in primary producers that are limited by Si. Primary producer Si limitation is set in *functional_group.csv* file parameter IsSiliconDep (see Table 6.2). In models that include C and P, both C and P are tracked like N (so the number of variable tracked is three times larger).

Group	**State variables tracked**	**GroupType in CSV file or *initial_conditions.nc***
Biomass pools		

pelagic **bacteria**	N in water column	PL_BACT
sediment **bacteria**	N in sediments	SED_BACT
(Age-structured) biomass pools		
These groups can be age structured if NumCohorts set > 1		
non epibenthic **primary producers**	N in water column N in sediments *Si in water column *Si in sediments NOTE: Age-structure not recommended	SM_PHY LG_PHY MICROPHTYBENTHOS DINOFLAG
epibenthic **primary producers**	N in epibenthic layer for each age group defined by NumCohorts parameter. If age-structure is selected for SEAGRASS it is not really age structure, but different parts of the plant – with "age 0" assumed to be the pool representing leaves, "age 1" the roots, and "age 2" the epiphytes on the leaves, see details here	PHYTOBEN SEAGRASS
infaunal **invertebrates**	N in sediments for each age group defined by the NumCohorts parameter. NOTE: Not typically age-structured	SM_INF LG_INF
epibenthic **invertebrates**	N in epibenthic layer for each age group defined by the NumCohorts parameter	SED_EP_FF SED_EP_OTHER MOB_EP_OTHER CORAL
water column **invertebrates**	N in water column for each age group defined by the NumCohorts parameter	SM_ZOO MED_ZOO LG_ZOO
land **plants**	N on land for each age group defined by the NumCohorts parameter	MARSH MANGROVE
large invertebrates	N in water column for each age group defined by NumCohorts parameter	CEP PWN
Age-structured groups		
vertebrates	RN in water column SN in water column Num in water column for each cohort defined in NumCohorts parameter	FISH SHARK MAMMAL BIRD REPTILE

size/age structured invertebrates of particular interest	RN in water column SN in water column Num in water column for each cohort defined in NumCohorts parameter	FISH_INVERT

Inanimate pools

Ammonia	NH3 in water column NH3 in sediments	NH
Nitrate	NO3 in water column NO3 in sediments	NO
Oxygen	O2 in water column O2 in sediments	O2
Dissolved organic nitrogen	DON in water column DON in sediments	DON
Dissolved silica	Si in water column	Si
Detrital silica	DSi in water column	DSi
Labile detritus	N in water column N in sediments	LAB_DET
Refractory detritus	N in water column N in sediments	REF_DET
Carrion (only dynamically tracked if fisheries are active)	N in water column N in sediments	CARRION
Sediment (only if flagIsEstuary=1)	SED in water column	SED

Timestep of the model

The timestep at which most processes are repeated is typically an *adaptive* daily (24h) or diurnal (12h) timestep, though tidal models with shorter timesteps (3h or 6h) also exist. This timestep is specified as dt in the *run.prm* file. Reproduction, recruitment and migration occur less frequently, typically once per year.

The *adaptive timestep* means that for most of the groups the processes are repeated once per timestep, but for small organisms, such as bacteria and phytoplankton the timestep is reduced to ensure that fluxes into and out of the component with the fastest turnover rates remain stable and numerical artifacts are not introduced into the mathematical solution scheme. The length of the adaptive timestep for such groups is determined to ensure that the relative change (flux) in any variable does not exceed a pre-specified proportion of the standing stock – this tolerance is set in the *biology.prm* file. The adaptive timestep changes through time, depending on the standing stock of these groups and the temperature, nutrient availability and other factors that affect the turnover rates in these "fast" variables.

Biological processes modelled

The modelled biological processes include primary production, bacterial processes, feeding, assimilation, waste production, explicit respiration (optional and not used in most models), growth,

movement and migration, non-predation mortality, maturation, reproduction and recruitment. In biomass pools all processes are modelled through biomass (nitrogen) turnover. In age-structured groups, the state variables are numbers-at-age and weight-at-age (tracked as reserve and structural nitrogen) which can be converted to biomass.

6.2. Defining functional groups

The functional group input file sets all the biological groups in the model. It can be set in CSV or XML format (see examples here). The number of groups is entirely flexible, but each model must include three obligatory detritus groups: carrion (CARRION), labile detritus (LAB_DET) and refractory detritus (REF_DET).

> The *functional_group.csv* file defines the biological groups used in the model and some of their key characteristics. These characteristics are used throughout the code to apply different appropriate routines for the group. The number of groups in the *functional_group.csv* file must match the number given in the K_num_tot_sp parameter in the *run.prm* file. The detritus groups must be defined as the last three groups in the file.

Currently each group must have the following parameters (characteristics). For new updates, check the Atlantis wiki functional group page

Table 6.2. Parameters in the *functional_group.csv* file.

Parameter name	Description
GroupCode	2 or 3 letter code of upper case letters used to identify each functional group. Atlantis will display an error message and quit if the group codes are not the correct format. This code is very important and applied to all functional group specific parameters
Index	An index of the group. Atlantis doesn't actually read this in but it is helpful for modellers to see as it defines the order of read-in for species vectors and the group ID number if you are trying to activate some of the debug/calibration fprintf statements using whichcheck in the *run.prm* file. The index must start form 0, as this is how the C programing language counts
IsTurnedOn	A boolean value (either 0 (false) or 1 (true)) to indicate if the group is active in the current model run. If the group is not active it will be excluded from the model. This option allows the user to exclude some groups from the model without having to setup new initial conditions and parameter files.
Name	Name of a group. This must match the tracer names in the *initial_conditions.nc* file as it used to find the tracers in the *initial_conditions.nc* input file.

LongName	The long name written to the NetCDF output files. This name can be a few words long and ideally should be informative. This is especially important when sharing the model with others.
NumCohorts	The number of cohorts in a functional group. If the value is > 1 the groups is assumed to be age-structured (either an age-structured biomass pool or age-structured group). In the earlier Atlantis versions age structured groups all had to have 10 cohorts. This is now relaxed and there is no upper limit on the number of age-groups to model (but remember, the more age groups the slower the model will run).
NumGeneTypes	Number of discrete phenotypes in each age group. This option is used when simulating evolution or multiple size-at-age representations and should only be set to >1 if flag_do_evolution is set to 1 in *biology.prm* file. If more than one phenotype is present, initial conditions will be have to be set up for each phenotype.
NumStages	This parameter describes the number of separate stages in a functional group, which are set to 1 or 2 if juvenile and adult stages have separate parameters for distribution, mortality, migration, feeding and others. In principle, more than 2 stages are possible, although this has not been tested. Typically all age-structured groups have 2 stages, but age-structured biomass pools can also have separate parameters for juvenile and adult stages and thus have 2 stages.
NumSpawns	Number of spawns in the year. Typically one per year, but more than one is possible (especially for ages structured biomass pool groups). The timing of spawn is given in Time_Age_XXX parameter in *biology.prm* file (if NumSpawns is 1 it expects a single value, otherwise it expects a vector of length NumSpawns).
NumAgeClassSize	Number of calendar years in one age group. This can be set to be anything between a few months to many years and allows simulation of different longevity organisms without having to setup too many age groups (which reduces simulation speed). Note, all individuals in an age group are considered identical (unless different phenotypes are set in NumGeneTypes, in which case all individuals in one phenotype of one age group are identical). However, the proportion of an age group per annual cohort is tracked so that recruitment variation is represented through time and not homogenized away (see chapter 10.9.6).
NumStocks	This sets the number of geographically distinct stocks for age-structured groups and age-structured biomass pools. The stocks must occupy different boxes or layers, i.e. they do not spatially overlap. Some biological parameters, like vulnerability to prey can be set as stock specific, which allows for better tuning. It has been applied to some groups that are known to include ecologically distinct stocks in different geographic

	areas. If NumStocks >1 then the corresponding number of stock specific parameters will be required in the *biology.prm* input file
VerticallyMigrates	If true then the group can move vertically through the water column. This should be set to 1 except for debugging purposes. If this is set to 0, no vertical movement will happen even if it is specified in the *biology.prm* file and the vertical distribution of individuals in the *initial_conditions.nc* file will be retained throughout the simulation.
HorizontallyMigrates	If true then the group can move horizontally throughout the model as well as migrate out of the model. No migration will happen if this is set to 0. Biomass pool groups which are marked as passive in the *initial_conditions.nc* will still be advected even if HorizontallyMigrates is set to 0.
IsFished	If set to 1 the group can be fished (targeted) by commercial or recreational fisheries and the catch will be reported in the output files, and is analysed in the assessment model. Note that if set to 0, dynamic fishing will still occur if parameters are given in the *harvest.prm* file.
IsImpacted	If true then the group can be impacted by fishing, bycatch or incidental fishing related mortality (e.g. destruction of living habitats). This parameter differentiates between targeted and accidental interactions; it is not reported in the landings. This means there must be values in the harvest files for this group. Note the group can be impacted by bycatch/incidental mortality even if no explicit targeted fishing is setup. This means the bycatch/incidental mortality parameters should be assessed carefully in the *harvest.prm* file. If both this value and IsFished are set to 0 then all fisheries related impacts on the group are disabled.
isTAC	If true, the group will be under total allowable catch (TAC) management. It will activate specific TAC related code in the Harvest submodel.
GroupType	The type of species. This is one of the most important parameters, as it tells Atlantis what kind of organism the functional group is representing and what kind of processes should be applied to it. Six group types are allowed for age-structured groups. A lot more group types are available to define invertebrate biomass pool types and these group types are used to dictate the adaptive timestep for simulating proceses, and all of them have some specialized ecological routines applied (see some details about their characteristics here). The main Atlantis model groups include: *AGE STRUCTURED GROUPS* 1) Standard fin-fish or generic vertebrate FISH 2) Size/age-structured invertebrate FISH_INVERT 3) Chondrichthyans SHARK 4) Birds BIRD 5) Marine mammanls (potentially terrestrial mammals in the future) MAMMAL

6) Reptiles (can be present in the water and on land) REPTILE.

BIOMASS POOL GROUPS

Phytoplankton

7) Small Phytoplankton SM_PHY
8) Large Phytoplankton LG_PHY (used in combination with IsSiliconDep to define diatoms)
9) Trichodesmium and Cyanobacteria TRICHO (N fixers)

Other primary producers

10) Microphtybenthos MICROPHTYBENTHOS
11) Dinoflagellates DINOFLAG
12) Phytoben PHYTOBEN (typically used to represent macroalgae)
13) Seagrass SEAGRASS
14) Turf TURF

Zooplankton

15) Small Zooplankton SM_ZOO
16) Medium Zooplankton MED_ZOO
17) Large Zooplankton LG_ZOO
18) Jellyfish JELLIES (in the past represented as LG_ZOO)

Large pelagic invetebrates

19) Cephalopod CEP
20) Prawns PWN

Infauna

21) Small Infauna SM_INF
22) Large Infauna LG_INF

Epibenthic organisms

23) Sediment epibenthic filter feeders SED_EP_FF (often used for bivalves, sponges)
24) Benthic grazers SED_EP_OTHER
25) Mobile epibenthos MOB_EP_OTHER (often used for crabs, lobster, octopus)
26) Corals CORAL

Bacteria

27) Pelagic Bacteria PL_BACT
28) Sediment Bacteria SED_BACT

Obligatory detritus groups

29) Carrion CARRION
30) Labile detritus LAB_DET
31) Refractory detritus REF_DET
32) Additional ice and land based groups are also available.

Ice dwelling

33) ICE_BACT
34) ICE_MIXOTROPHS
35) ICE_DIATOMS
36) ICE_ZOOBIOTA

	Land dwelling (vegetation only for now) 37) MARSH 38) MANGROVE
IsPredator	If true then the group is assumed to eat other groups by the assessment code (for the purposes of defining network indices etc), it does not affect the ecological model.
IsCover	If true the group is classified as a habitat. This means that the group becomes a unique habitat type, that can be used by other groups in the model for shelter etc, in addition to the general four geological habitat types included by default (**reef, flat, soft, canyon).** For example, if IsCover is set to 1 for three functional groups (e.g. BFS, BFF, and MA), the model has a total of seven habitat types. In this case the habitat dependency parameter habitat_XXX in the *biology.prm* file must have 7 values for each functional group. The living habitats are always the first entries and the geological classes always the last 4 entries. So in this example, the first three entries will refer to the IsCover groups in the order they are listed in the *functional_group.csv* file and the last four entries will refer to reef, flat, soft, canyon habitats.
IsSiliconDep	If true, the group is assumed to be Si limited and its energy flows are tracked both through nitrogen and silicon. All Si limited groups must have separate Si tracers set in the *initial_conditions.nc*, in addition to the usual Nitrogen tracers. At present most models have this set to true for Diatoms and MicroPhytopbenthos.
IsAssessed	If true then this group will be assessed in the Assessment submodel. This submodel will generate a number of assessment indices.
IsCatchGrazer	If true, the group grazes on catch, for instance pinnipeds that eat fish from nets. If set to 1 for a functional group XXX, then availability of catch of each functional group should be given for this group in the pFCXXX parameter in the *biology.prm* file. The pFC parameters are not required for groups that are not catch grazers.
OverWinters	If true, this group will go into hiatus over winter. The overwintering can be triggered and ended by time of the year or temperature. This is set using overwinterStartTofY_XXX, overwinterEndTofY_XXX, overwinterStartTemp_XXX, overwinterEndTemp_XXX parameters in the *biology.prm* file. These parameters only have to be set for overwintering groups.
isCultured	This identifies aquaculture groups, and applies aquaculture processes to them. Parameters required for aquaculture groups are described here.
isHabDepend	Identifies groups that are habitat dependent. Note, this value takes precedence over habitat preference parameters in the *biology.prm* file. If isHabDepend is

	set to 0 then the group is not affected by habitats, even if appropriate parameters are given in the *biology.prm*
numMoveEntries	The number of movement time periods you would like the group to have. Historically this has been fixed at 4 to allow for seasonal movement, but currently any number of moves is allowed. See further details on parameters in *biology.prm* here

NOTE!

What is the role of detritus in an ecosystem model? An excerpt from Murray and Parslow (1997) PPBIM report:

"The detrital pool acts as a store of fixed nutrient, which introduces a delay into nutrient recycling. Particulate detritus sinks and thereby transports nutrients out of the surface layer into deep water or to the sediment. Dissolved organic matter (DOM) does not sink, and very large pools of refractory DOM are found in the open ocean and in coastal waters. Horizontal transport of particulate detritus and DOM from regions of high production to more oligotrophic areas can also be important"

"Three forms of non-living organic matter are included in the model: labile particulate detritus, refractory particulate detritus and dissolved organic matter [NOTE – these same pools, plus Carrion, are included in Atlantis, which is based on the PPBIM]. These are represented by the symbols DL, DR and DON respectively. The particulate detrital pool [in nature] contains a wide variety of particles, with diverse sources, ages, sizes and chemical composition. The model includes two functional components: fresh labile detritus breaking down on time scales of a few days, and semi-refractory detritus turning over on time scales of a year. This allows the model to treat perturbations to the nutrient cycle on time scales of days to years."

"[In nature] The pool of dissolved organic matter also contains a variety of compounds with diverse turnover rates. In oceanic waters there is a large pool of highly refractory DOM, whose average age has been estimated to be as much as 3400 years (Williams 1975). The bulk of this material is essentially irrelevant for the cycling of nutrients on the time scales of [model] interest. However the DOM pool also includes much more labile components. Highly labile sugars and amino acids released through cell lysis or messy feeding may be consumed by bacteria or break down on time scales of hours (Fuhrman 1987) to days (Holibaugh and Azam 1983). This material is not modelled explicitly, but is treated as though it was remineralised immediately to form dissolved inorganic nutrient. The model includes a single DOM pool with an assumed turnover time of the order of a hundred days. … Because it turns over on time scales, which are long compared with both phytoplankton uptake of inorganic N, and transport within the Bay, it plays a role in reducing N sequestration in coastal waters."

6.3. Defining the GroupType correctly

The definition of a group is very important as it will determine the ecological routines and the adaptive timestep applied for the group (see timestep description in chapter 6.1). Table 6.3 shows routines that

are called for different group types. The routines are further modified depending on whether consumers live in the water column or sediment.

Table 6.3. Routines called for different non-vertebrate GroupTypes in Atlantis code. The *Call_Group_Process_Function()* routine in **atecology.c** determines the specific routine used from the **atGroupProcess.c** file non vertebrate groups. Note, that the greatest diversity ofroutines are called for different benthic invertebrate types (for clarity benthic invertebrates are shown in ***bold italics***).

Routine name in Atlantis code	**GroupType for which the routine is applied**
Dinoflag_Process()	DINOFLAG ICE_MIXOTROPHS
Phytoplankton_Process()	LG_PHY SM_PHY PHYTOBEN SEAGRASS MICROPHTYBENTHOS TURF ICE_DIATOMS
Invert_Consumers_Process()	***SM_INF*** ***LG_INF*** SM_ZOO MED_ZOO LG_ZOO CEP PWN ICE_ZOOBIOTA
Epibenthic_Invert_Process()	***SED_EP_FF*** ***MOB_EP_OTHER***
Sediment_Epi_Other_Process()	***SED_EP_OTHER***
Coral_Process()	CORAL
Pelagic_Bacteria_Process()	PL_BACT
Ice_Bacteria_Process()	ICE_BACT
Sediment_Bacterica_Process()	SED_BACT
Labile_Detritus_Process()	LAB_DET
Refractory_Detritus_Process()	REF_DET
Carrion_Process()	CARRION

6.4. Modifying functional groups in parameterised models

Details on how to add a new functional group into an existing model are given on the wiki here. This will require a significant effort and reparameterisation of the model.

Alternatively, if changes to functional groups are needed, it might be easier to use an existing group and change it into something new. You may not want to change the three letter code, as that would involve changing all the parameter names for the group.

7. RUN PARAMETERS

The *run.prm* provides all the key information required to control the model run, load optional submodels and also parameters controlling the debugging and detailed output. The table below lists the main parameters used in the *run.prm*, but please remember to follow the news on the Atlantis wiki as new parameters are sometimes added.

Table 7.1 List and description of parameters in the *run.prm* file

Parameter	Meaning
General parameters controlling the run	
title	Title of the run. Ideally this should be an informative title (e.g. SE Tasmania model with constant forced fishing and base level fishing pressure on FMM increased 2000 fold) as it will written at the start of the *log.txt* file and into the output.nc file and will be displayed on the top of plots in visualization tools such as OLIVE or DIVE
dt	Timestep of the model. The timestep is typically given in hours (e.g. "12 hour"), however Atlantis reads the units together with the numerical value so other units "minute" or "day" may also be used. Typically the timestep is set to 12 or 24 hours, but it can be set to other intervals as well. It is not typically recommended to make it >24h and one should think carefully whether shorter than 12h is really needed, as the model will run slower. Some coastal, very dynamic, areas have 6h or even 3h timesteps, but that will require high resolution hydrodynamic forcing data.
tstop	Day on which the model run will end. If you want to run the model for 10 years, set this to 3651 (technically 3650, however given most output is done annually it is best to add an extra day to make sure the final output is printed before the model stops).
tburnday	Number of days that define the burn-in period. At the end of this period the "virgin" state of the model is stored. This means that every comparison of species biomass to the original or "virgin" state will use values stored at the end of the burn-in period and not those set in the initial conditions. This is useful for models that have unstable dynamics at the start of the model run.

Output files and reporting	
toutstart	Day on which to start writing the output. This is normally set to 0, but can be set to a larger value for long runs if there is a need to save memory and there is no interest in the burn-in period dynamics.
toutinc	Periodicity in days for the Ecology submodel snap shot output to the NC files, in days. In longer runs, this is often set to monthly (30) or yearly (365) so file sizes do not become very large. In poorly behaved runs (that are crashing), or ones with extreme events which condition subsequent behaviour, fine scale output (e.g. at the same level as the timestep) may be required. However, remember that a lot of output is written at each reporting period, so make sure you set reasonable values for long well-behaved runs (quarterly or even yearly). Note that in Atlantis all years are 365 days long. If monthly output is required and it is set to 30 days then after one year the output would be shifting by 5 days every year.
toutfinc	Periodicity of the cumulative output from the Harvest submodel (catches, quotas, discards, etc) to the NC files (CATCH.nc, TOTCATCH.nc). These values are the sum of all effort, catch (etc) since the last output dump. Unlike the ecology output, they are not snap shot values.
tsumout	Periodicity of the output to the TXT files.
flagannual_Mest	If 1, estimates of mortality per predator are written out annually, otherwise they are written at the tsumout frequency
fishout	If 1, it switches on the fisheries output (it is recommended that this is on if you have fisheries in your model, but is unnecessary if it is an ecology model only).
flagreusefile	Whether to replace or append ouputs to the existing files, when starting a new run. If it is set to 0 Atlantis will quit with a message that a file with such name already exists. If it is set to 1 results will be appended to the existing file, if set to 2 it will overwrite (replace) the existing file without warning. NOTE: This is old code. If you really want it to append please run a test first to see if it worked. If not, let the developers know, as it likely needs updating. Setting flagreusefile to 0 or 2 is recommended.
flag_age_output	If set to 1, an additional output file *agebiomindex.txt* will be created and the total biomass output will be given for each cohort. This can bea very useful output file, because without it you will need to run a specially written script over the *output.nc* file to determine the biomass per age group per functional group.

Parameters loading different submodel or options

flagecon_on	If 0 the Economics submodel is not loaded
flag_fisheries_on	If 0 the Harvest submodel is not loaded
flag_skip_biol	If 1 the Ecology submodel is not loaded – only for debugging!
flag_skip_phys	If 1 the Physics submodel is not loaded. This will actually produce an error message and Atlantis will quit. **So this value should be NEVER set to 1.**
fishmove	If 0 all age structured group movements (vertical and horizontal) are turned off

Parameters controlling detailed outputs for a specific box and diagnostic reporting
The outputs are written into the *log.txt* file for model tuning and debugging purposes.

debug_it	If 1 - give detailed outputs of all processes for a selected box and time period into the *log.txt* file
checkbox	Box number for which to give detailed outputs into *log.txt* file
checkstart	Day on which the detailed output in the *log.txt* file for the box above should start
checkstop	Day on which the detailed output in the *log.txt* file for the box above should end **Note!** Detailed output will likely produce a huge amount of data, so make sure the time period of detailed reporting is not too long or the run is short
debug	This parameter has a lot of options about what topic area or code path to give detailed outputs for (for debugging). These settings trigger fprintf lines in the code and print information to the *log.txt* file. Each setting focuses on different processes. Setting it to 0 turns off detailed outputs. Other parameter values include debug = 1 (for fisheries catch fprintf) debug = 2 (for discards) debug = 3 (for forced fish catches) debug = 4 (for the assessment model processes) debug = 5 (for spatial management)

	debug = 6 (for basic effort options) debug = 7 (for economically driven effort allocation model) debug = 8 (for economic calculations) debug = 9 (for quota setting and handling) debug = 10 (for aging in age structured groups) debug = 11 (for recruitment in age structured groups) debug = 12 (for spawning in age structured groups) debug = 13 (for migration in/out of model domain) debug = 14 (for movement within the model domain) debug = 15 (for stock-based allocations within a functional group) debug = 16 (for basic overall biomass calculations) debug = 17 (for feeding) debug = 18 (for all ecological processes) debug = 20 (print biology process function parameters) debug = 21 (print prey function parameter) debug = 22 (debug clam - atlantis linkage) debug = 23 (for deposition in the physics sub-model) debug = 24 (for the forced mortality scalar) debug = 25 (for the forced growth scalar) debug = 26 (for FSPB additions) debug = 27 (for debugging the C and P code) debug = 28 (for debugging the SS3 tiered code) debug = 29 (for external scaling) This list changes as new debugging issues arise. So if you want to know the latest list, the entire list of currently available debugging options can be seen in the **atlantisboxmodel.h** file (around line 750) in the Header files subfolder of the **atlantismain** library. There is also a list on the Atlantis wiki
fishtest	If 1, count up total population of age structured groups after each main subroutine
flaggape	Periodically list prey vs gape statistics in *log.txt*
flagchecksize	Periodically list relative size in *log.txt*. This is useful during model calibration for seeing the current size versus size given in the initial conditions.
flagagecheck	Periodically list age structure per cohort in *log.txt*
flagdietcheck	Periodically list realised diet matchups per functional group in *log.txt*
checkNH	Give detailed logged output for NH fluxes in the box selected with checkbox. Useful understanding what groups may be at fault if you get an "unstable variable" crashing the model before completion of the run.

checkDL	Give detailed logged output for DL fluxes in checkbox. Useful understanding what groups may be at fault if you get an "unstable variable" crashing the model before completion of the run.
checkDR	Give detailed logged output for DR fluxes in checkbox. Useful understanding what groups may be at fault if you get an "unstable variable" crashing the model before completion of the run.
checkbiom	Give detailed logged output for biomasses of which_check group in the checkbox (see above)
which_fleet	ID number of fleet to track (if don't want to track anything set this to a value greater than the number of fisheries in your model). The ID is the same as in the *fisheries.csv* file
which_check	ID number of group to track (if don't want to track anything set this to a value greater than the number of functional groups in your model). The ID is the same as in the *functional_group.csv* file
habitat_check	Which distribution type (0-water column, 1- sediment, 2- epibenthic, 3-land, 4-ice) to report for the group marked with which_check. Some groups are ony tracked in one distribution type, others are tracked in both water column and sediments.
move_check	ID number of group for which movements should be tracked

Key parameters determining the model runs and inputs

flaghemisphere	0=southern, 1=northern hemisphere. This flag is important in calculating winds and circulation (affects coriolos and seasons).
flagAllowLand	If set to 1 if the model includes dynamic land components (this requires a large number of additional inputs). If set to 0 any land boxes are treated as boundary boxes and are not executed.
flagIsEstuary	Flag to indicate if the model is estuary. If true a SED sediment tracer is required in the initial conditions input file and tidal emptying of shallow boxes is allowed (with fish moved to the nearest water channel).
trackAtomicRatio	If set to 1, then not only N, but also P and C are tracked. This will require many additional tracers in the input files.
track_rugosity_arag	If set to 1, rugosity and aragonite saturation will be tracked and additional parameters required. Important for modelling corals.
track_pH	If set to 1, pH values will be tracked – important for ocean acidification simulations.

track_contaminants	If set to 1, contaminants will be tracked. This will require information on the number and name of the contaminants to be tracked, as well as additional tracers and initial conditions (so as to allow for the tracking of contaminants through each of the functional groups in the model).

Other parameters

mirror_invalid	If this flag set to 1 and the hydrodynamics model attempts to send water from one cell into an invalid cell (e.g. a layer in a box that does not exist) the water flow will instead will be sent into the layer closest to the intended destination layer. This option was needed because historically some hydrodynamic models tried to send water into layers below what Atlantis was modelling, so water had to be redirected to the extant bottom layer instead. Set this flag to 0 if your hydrodynamic forcing is working well or you want Atlantis to quit with a warning that something is wrong with the hydrodynamic forcing files.
flag_replicated_old	This flag is used to maintain legacy code options. While we fix true bugs sometimes we also update the code to better represent a process where we had made a rough first attempt that can now be done in a better way. However, such changes can change model dynamics and so for legacy models we need to keep the legacy options. When set to 1, the new code changes are not applied and the old results can be reproduced. However, for all new models it is recommended that this flag be set to 0, as the new code is more rigorous and defensible.
check_dups	If set to 1, Atlantis will check for duplicated parameters in the parameter files before starting the run. This can be used to ensure that same parameter is not entered twice with different values. However, it takes a LONG time to execute (especially for big models). If it is set to 1 the model run will not start for quite a while until all possible duplicates are checked. If you are concerned about duplicate entries, go through the file once with this until all duplicates are found and then turn it off.

8. FORCE PARAMETERS

Atlantis can take a lot of externally forced tracer and parameter values. Details on such forcing files are given in the *force.prm* file.

The minimum *force.prm* file will contain the pathname and load information for a hydrodynamics forcing file and nothing else. However, in most models a lot more of physical, ecological and socio-

economic processes are forced and pathnames for all these files will be required in *force.prm*. See table 8.1 below for the list and application examples of different forcing options.

NOTE!

Absolute time in Atlantis: model time, hydro time and TS time

The time (t) entry in the *initial_conditions.nc* file indicates the actual absolute time in calendar years that marks the start data and time of the model simulation. For example:

t:units = "seconds since 2000-01-01 00:00:00 +10" ;

This indicates that this model starts at midnight on the first of January 2002. This is the main Atlantis time (called **model time**). The hydrodynamics forcing file also has its own absolute start time, which is called **hydro time**; this does not have not to match with the model time. For example, the hydrodynamics data might have five calendar years and start from 2002 July. Atlantis will try to match the model time to the hydro time. In the example, above Atlantis will start the run by loading in the hydrodynamics beginning in January not July and run from there, matching the month of the year when rewinding the hydro data.

The time series forcing files also have a parameter indicating the absolute time of the start (**TS time**). For example, the forcing data may start from 1990 July. On read in Atlantis will by default convert these times into model time, skipping that part of the time series that occurs before model time begins. In the example here, the first 10 years of TS data will be skipped, and only the forcing information from year 11 will be used. Alternatively, if TS time starts later than the model time, Atlantis will run the simulation withouth any forcing until the model time matches the TS time.

Users have an option to match the TS time not to the model time but to the hydro time. This is done by setting ts_on_hydro_time=1. If ts_on_hydro_time=0 then the default will be used (matching to model time). However, keep in mind that if TS time is matched to hydro time any rewinding of hydrodynamics data will also mean the rewinding of the forcing data. For example, if 50 years of forcing data is available and the model runs for 50 years all forcing will be used if TS and model times match. However, if TS is matched to hydro time and the hydrodynamics is rewound every 5 years, then only 5 years of TS values will be used before the TS file is rewound along with the hydrodynamics data.

Note that the rewinding of TS filed described above (done when ts_on_hydro_time=1) is not the same as specific rewind options associated with most TS files. The specific TS rewind parameters apply when the time series of the run (either on model or hydro time) exceeds the time series given the TS file. See details on the TS file format below

It is recommended that users download the sample South East Tasmanian (SETas) model input files that give details on the structure of the forcing and other parameter files. The table below lists currently available forcing options

Format of the time series (TS) forcing files

The TS forcing files have a standard input format read by Atlantis. At the top of the file the user can provide a comment on what the file represents (marked by a single # at the start of each line). Then the user needs to provide a description of variables in each column of the TS data file (Atlantis needs this to know what the variable name is, its units and missing value marker). Atlantis recognises these entries as column description with ## at the start of each line. If a single # is given Atlantis will assume it is a comment and ignore it, so if you are having read-in issues make sure you have all column decriptions beginning with ##. Underneath this header section is the actual data table in columns. The number of columns in the data should be the same as the number of column descriptions.

The first column always indicates time and is given in the units defined (typically days). The data in the TS file can be given at longer time intervals, such as every 30 or every 365 days. If forcing data is not given for every timestep of the model Atlantis will either use the the last valid value (i.e. the last value before the gap in the data) or interpolates between the last valid value and the next (future) value in the time series file. The handling of patchy data is determined by the typeXXXXts parameter, such as typeCatchts or typeDiscardts. If typeXXXXts is 1 the same value is used throughout the period between two forcing time steps, if typeXXXXts is 0 then Atlantis will interpolate the value. Not all forcing files have the typeXXXXts parameter and if this parameter is not available it means that **Atlantis will interpolate values**. For example, if the value on day 1 was 10 and the value on day 5 is 50, setting typeXXXXts to 1 means that on day 2, 3 and 4 the value returned will be 10, while on day 5 value it will be 50. Setting typeXXXXts to 0 means that the value returned on day 2 is 20, day 3 value is 30 and day 4 value is 40. See further details about the interpolation here

Make sure you chose the correct value for the rewind parameter, as that will determine what happens if forcing file time series is shorter than the model run: rewind=0 means that the last value will be used for the remaining of the run, rewind=1 means that the time series will be reused from the start. More details on the structure of different TS files can be found here.

 # THIS FIRST LINES ARE A COMMENT ON THE SOURCE OF THE DATA
GDP time series for Australia 1972 to 2008 based on treasury projections in 2006 Budget
Fuel cost time series from autoregressive cost model based on Australian TACC diesel cost data
#
 # - EVERYTHING BELOW IS READ BY ATLANTIS AND MUST BE IN A STANDARD FORM
COLUMNS 3
##
COLUMN1.name Time
COLUMN1.long_name Time
COLUMN1.units days since 1972-01-01 00:00:00 +10
COLUMN1.missing_value 0
##
COLUMN2.name GDP
COLUMN2.long_name GDP
COLUMN2.units dollars
COLUMN2.missing_value 0

```
##
## COLUMN3.name FuelCost
## COLUMN3.long_name FuelCost
## COLUMN3.units dollars
## COLUMN3.missing_value 0
##

1 0.206 0.64
2 0.207 0.68
3 0.208 0.69
4 0.209 0.65
5 0.204 0.64
6 0.202 0.63
7 0.210 0.62
```

NOTE: If you are having issues reading a TS file

1) Make sure there are no extra spaces or tabs after the name of the column (this can sometimes happen if you have created your TS file in excel and then pasted on to a text editor). Atlantis should now ignore those spaces, but in the past it was an issue.
2) Make sure the model is space not tab delimited. Again Atlantis should now be able to read either option, but just to be careful.
3) If a collaborator created the file for you and they use a different operating system, make sure the file is in the correct file type for your machine (in text-pad, text wrangler etc check that it is a windows type if you are on a windows machine and linux if you do not use windows). See chapter 2.2 for file conversion details.

Table 8.1. List and description of external forcing options available in Atlantis

Forcing	**Parameters needed in *force.prm***
Forcing of physical parameters	
Hydrodynamics	This is the minimum forcing required to run Atlantis. See chapter 4 for details on hydrodynamic forcing files. nhdfiles gives the number of hydrodynamics files that will be supplied (read in). It is possible to put all the forcing time series into one file, but these files can get very big. Therefore, it is often easier to split the series into a number of smaller files. Atlantis will read these files in the order given, using forcing data from the first file, then from the second one and so on. Once the data from the last file is used, Atlantis will return to the first file and start reading from there again.

	The hydrodynamics files are automatically looped. For other forcing input files an additional parameter (_rewind) indicates whether the forcing data should be looped when the time series finishes (=1) or stay on the last value (=0) Note that the pathway to the file is tagged for Atlantis by the parameter name hdN.name where N indictes the number of the file, remembering that numbering in C (and thus Atlantis) starts at 0. So hd0.name, hd1.name, hd2.name and so on. For example, the following would indicate there are 2 hydrodynamic files and that they are located in the directory the model is being run from. nhdfiles 2 hd0.name hydrofile1.nc hd1.name hydrofile2.nc The following indicate the files are stored in the inputs sub-directory (a directory one level down from the run directory) hd0.name inputs/hydrofile1.nc hd1.name inputs/hydrofile2.nc Finally if the inputs folder is in the directory that is at the same level than the run directory then the path to the hydrodynamics files should read as hd0.name ../inputs/hydrofile1.nc hd1.name ../inputs/hydrofile2.nc For more information on how to indicate a path see, for example, this Wikipedia site
Temperature	Atlantis can calculate the temperature dynamically, but most models use temperature forcing from more accurate hydrodynamic modes. See chapter 4 for details of temperature forcing files. The lay out of the list of files is as for the hydro file – with as many files listed as given by the ntempfiles parameter. use_tempfiles 1 if temperature forcing (NC files) are being used ntempfiles N where N is the number of files being provided Temperature0.name temp1.nc the path and filename of the first file Temperature1.name temp2.nc the path and filename of the second file Temperature2.name temp3.nc the path and filename of the third file … temp_rewind 0 flag indicating whether to rewind (loop over) the temperature forcing data (1) or not (0)
Salinity	Atlantis can also load in salinity forcing from more accurate hydrodynamic modes. See chapter 4 for details of salinity forcing files. The lay out of the salinity parameters and file name list is as for temperature.

	use_saltfiles 1 nsaltfiles N Salinity0.name salt1.nc Salinity1.name salt2.nc … salt_rewind 0 (or 1)
pH forcing	If pH is tracked, it is possible to calculate it dynamically or provide external forcing values from specific models. The pH forcing is provided through spatial NC files in the same way as for temperature and salinity. use_phFiles 0 npHfiles N pH0.name pHforceA.nc pH1.name pHforceB.nc pH_rewind 0
forcing of other tracers	It is possible to force any other tracers with external data. See details on the wiki here. When new data is read in at the end of the timestep any existing value is overwritten with the value from the NC forcing file. This has been used for contaminants and for oxygen (e.g. when internal Atlantis calculations were not reproducing oxygen dynamics sufficiently well). This is a useful option, but should be used with care for any tracers that contains N (and P or C if tracking those tracers), as it can violate the assumption of conservation of matter. use_force_tracers 1 (this flag indicates whether additional tracer forcing files should be read it (1) or not (0)) nforceTracers N (where N is the number of tracers that will be forced) tracerNames nameA nameB … The list of tracer names (matching the names used in the *initial_condition.nc* file) that will be forced – with each name on a new line. For example, nforceTracers 2 tracerNames 2 Oxygen Arsenic For each tracer listed you must then give the filenames using the format TRACERNAME_nFiles N (where TRACERNAME is the name of the tracer and N is the number of files to be loaded) TRACERNAME_File0.name TRACER_1.nc (path and filename for the first forcing file for the tracer) TRACERNAME_File0.name TRACER_2.nc (path and filename for second file etc) TRACERNAME_rewind 0 (whether to rewind the forcing files (1) or not (0))

	for example Arsenic_nFiles 1 Arsenic_File0.name Arsenic.nc Arsenic_rewind 0
Point sourcing of nutrients	Even though Atlantis calculates internal nutrient dynamics, it is relatively common to force additional nutrient inputs (e.g. from sewage outfalls, river mouth, upwellings etc). For example, the California Current, Guam, Gulf of Mexico and Chesapeake Bay models all use nutrient loading -forcing with nutrient time series (Brand *et al.* 2007; Ainsworth *et al.* 2015; Weijerman *et al.* 2015). While such inputs are mostly used to imitate nutrient inputs from sources clearly outside the model domain (e.g. rivers), they can also be used as a proxy for multi-decadal trends in ocean productivity, even if full multi-decadal oceanographic data is not available and must be 'looped'. Externally forced nutrients sources (or sinks if negative values are given) are provided as TS files npointss N (where N is the number of externally supplied point source/sink files) For each TS file the following details must be given pss0.name XXXX (where XXX is a meaningful name for the point, such as a river name, used so the modeller knows what this point represents; this name is NOT used by Atlantis) pss0.location X Y Z (e.g. 3951091.3 1789202.7 **-1)** pss0.data PSSTS1.ts (path and filename of the forcing TS file) pss0.rewind 1 (whether to rewind the forcing file (1) or not (0)) **Note!** The location values (X Y Z) has three values – coordinates to input the point source (3951091.3 1789202.7) and the depth **(-1).** To make sure the point source is deposited in the desired box it is a good idea to ensure the X Y values match the "inside" point of the box as given in the BGM file (see chapter 3). The **last number in the location is very important** as it will determine the correct reading of the forcing data. For point source data the depth must be a negative number for it to be deposited in the water. Other time series forcing files (e.g. for catch) follow a similar format to the PSS files – requiring a name, location, data (filename) and rewind flag. However, **for these forcing files the third number gives the box ID number (from BGM file, see chapter 3) not a depth.**
Bottom stress	Bottom stress values are used to calculate resuspension and additional macrophyte mortality due to wave action. Atlantis can calculate bottom stress values dynamically, but if processes affected by bottom stress are a focus of the study it is recommended to force stress externally from more precice hydrodynamic models. At present there is no flag to indicate whether this file

	will be loaded (future releases are likely to switch to a flag as for temperature and salinity, e.g. use_stressfiles). At present if the BottomStress tag is in the *force.prm* then the listed file will be read automatically. BottomStress stress.nc (the path and filename of the stress NC input file). If you do not wish to use bottom stress simply omit the BottomStress entry from the *force.prm* file.
Precipiation and evaporation	See chapter 5.2 on how the precipitation forcing data is used. As in case of bottom stress, precipitation and evaporation are forced through spatial NC files. They will be loaded if the tag name is present in the *fore.prm* file (if not provided the model will not look to load the files). Precipitation rain.nc (path and filename for the precipitation) Evaporation evap.nc (path and filename for the evaporation) **These files will not rewind**.
Solar radiation	Solar radiation can be calculated dynamically by Atlantis or can be forced through a time series TS file. In the later case (TS file used) the same value is used for the entire model domain. If you do not wish to load a solar TS file do not provide a Solar_radiation entry in *force.prm* and set lim_sun_hours to 1 in the *biology.prm* file. As with bottom stress the solar radiation file will be loaded if the tag name is present in the *fore.prm* file Solar_radiation solar.ts (path and filename for the solar forcing file) Solar_radiation_rewind 0 (a flag indicating whether to rewind the data file (1) or not (0))
CO2 concentrations	Time series of atmospheric CO2 concentrations. This is used to dynamically calculate pH values and is required if trac_pH=1 in *run.prm*. See further details on the wiki here
Ice	If ice is modelled dynamically, the user must provide ice forcing data. See details on the wiki here

Forcing of ecological parameters

Forcing of recruitment	Recruitment can be forced in two ways, see details on the wiki here The first is to use a single recruitment time series across the entire model domain. To do this the tag Recruitment_time_series must be added to the *force.prm* file Recruitment_time_series filename.ts (path and filename for the time series) This time series provides a TS file of recruits entering the entire model domain. Box specific forcing of recruitment is also allowed as described in the link above.

	The second way is to allow for time series of recruitment to be read in for specific boxes. To do this set nMultRects N (the number of boxes where recruitment time series will be provide) typeMultRects 1 (the type of time series – whether to use the last value (0) or interpolate between values (1)) The for each of the N boxes you need to provide the following: MultRects0.name Meaningful_name MultRects0.location X Y BOXID (e.g. 4924432.61 3298510.07 94) MultRects0.data filename1.ts (path and filename of the TS file for box BOXID) MultRects0.rewind 0 (whether to rewind the file (1) or not (0)) MultRects1.name Meaningful_nameB MultRects1.location X Y BOXID2 MultRects0.data filename2.ts MultRects0.rewind 0 …
Recruitment scaling	Recruitment can also be scaled by a time series of species-specific scalars provided in a time series (TS) file. To use this option include the tag Recruitment_enviro_forcing in the *force.prm* file Recruitment_enviro_forcing scalarfile.ts (path and filename for the time series) More details are given on the wiki
Larval connectivity matrix	To activate set use_larvalfiles 1 in the *force.prm* and then provide the filename of the connectivity matrix per species using larval dispersal Larval0.name connectfile.nc (path and filename for the connectivity matrix NC file) Larval_rewind 0 (a flag indicating whether to rewind the file (1) or not (0)) The NC file must contain an entry XXX_Connnectivity (where XXX is the group code from the *functional_group.csv* file). These entries are annual connectivity matrices. If there are les entries than there are years in the run then the rewind flag will dictate whether the file is rewound or not. This is recent functionality that has not been widely used. It allows to enter larval dispersal matrix. See further details here
Linear mortality scaling	External scalar on linear mortality (*mortsc* in chapters on Ecology routines) – can also be supplied as an additional site specific mortality rate rather than a scalar. This can be provided as a TS file or spatially with box-specific values as an NC file. See details here on how the external mortality scaling is set and calculated.

Scaling of size, growth rate or maturation proportion	The scaling of growth and maturation has been used to explore ecological consequences of long-term trends in life-history traits (e.g. Audzijonyte et al. 2013; Audzijonyte & Kuparinen 2016). The scaling is forced through external forcing scalars on size (SN and RN values), growth rate (mum parameter) or maturation (FSPB parameter). The scalars are applied by scaling the existing values. See further details on size change forcing here and on mum and FSPB scalar here

Forcing of fisheries and socio-economic parameters

It is possible to provide forced catches, discards, effort and other socio-economic parameters. Forced catches and discards, in particular, have been used to parameterise the model and emergent biomasses given the known catch history. See, for example, details in Fulton et al. 2007 and Link et al. 2011. The catch and discard forcing files are provided as box-specific species-specific time series. At each timestep Atlantis will then take the forced catch from the species biomass. See further details in the description of the Harvest submodel.

Catch	To force catches use the following parameters nCatchts N (where N is the number of catch force time series (TS) files) typeCatchts 1 (whether to use the last value (1) or interpolate (0) between the two forcing time steps) This is followed by the actual box-specific forcing file information. Atlantis will attempt to take the forced catch from the specified box. If not enough biomass is available Altnaits will first try to take the biomass form younger age classes, then may look to neighbouring boxes (depending on the flagimposecatch setting in *harvest.prm*) and if it still can't fill it will hold it over ("carry over") and add the remaining amount to the biomass to take the following day. If at the end of the year there still remains catch that cannot be taken it is noted in the *log.txt* file and the "carry over catch" is zeroed to start again in the new year. Catchts0.name meaningful_name (e.g. box0catch; this is NOT used by Atlantis) Catchts0.location X Y BOXID (e.g. e.g. 370889.3287 -717418.8373 0. This is the x, y and box ID indicating the box where the catch will be extracted. It is safest to use the "inside" point of the box BOXID as given in the BGM file - see chapter 3 on details about model geometry) Catchts0.data catchfile1.ts (path and filename of the catch TS file) Catchts0.rewind 0 (whether to rewind the data file (1) or not (0)) **Note!** Remember that the third number of the location is the box ID number (from BGM file, see chapter 3). **It is very important to give this number correctly or else Atlantis will not read or apply the forcing inputs correctly.**
Discard	The entry format is as for catches above, but instead of catches provide discard values for the specified boxes

	nDiscardts N (where N is the number of time series (TS) files) typeDiscardts 1 (whether to use the last value (1) or interpolate (0) between the two forcing time steps) Discardts0.name meaningful_name Discardts0.location X Y BOXID Discardts0.data disfile1.ts (path and filename of the TS file) Discardts0.rewind 0 (whether to rewind the data file (1) or not (0))
MPA	This forcing file "closes" a proportion of the box to fishing (with the columns in the TS file being the closure per fishery), simulating marine protected areas (MPA). This is used in place of the MPA vectors in *harvest.prm* and is useful where MPAs vary through time. Further details are provided in the Harvest submodel description nMPAts N (where N is the number of time series (TS) files) typeMPAts 1 (whether to use the last value (1) or interpolate (0) between the two forcing time steps) MPAts0.name meaningful_name MPAts0.location X Y BOXID MPAts0.data mapfile1.ts (path and filename of the TS file) MPAts0.rewind 0 (whether to rewind the data file (1) or not (0))
Effort	This forcing file provides the effort time series to apply in a box (with the columns of the TS file giving the effort per fishery) Further details are provided in the Harvest submodel description nEffortts N (where N is the number of time series (TS) files) typeEffortts 1 (whether to use the last value (1) or interpolate (0) between the two forcing time steps) Effortts0.name meaningful_name Effortts0.location X Y BOXID Effortts0.data effortfile1.ts (path and filename of the TS file) Effortts0.rewind 0 (whether to rewind the data file (1) or not (0))
Residuals	These time series provide the price residuals per market if using the economics model and forcing historical prices. One time series is required per market, with the columns providing the residual values per species. nResidualts N (where N is the number of time series (TS) files) typeResidualts 1 (whether to use the last value (1) or interpolate (0) between the two forcing time steps) Residualts0.name market_name (NOT used by Atlantis) Residualts0.location X Y BOXID (x,y location and box ID for the port which supplies the market) Residualts0.data resid_file1.ts (path and filename of the TS file) Residualts0.rewind 0 (whether to rewind the data file (1) or not (0))

Economic statistics	Economic statistics include GDP and fuel costs. They are used in the economically driven dynamic fishing calculations. They are not box specific, but just time series values that apply across the entire model. nEconts 1 (typically 1 or 0; only non-zero if using the economic effort model) typeEconts 0 (whether to use the last value (1) or interpolate (0) between the two forcing time steps) Econts0.name meaningful_name (NOT used by Atlantis) Econts0.location X Y BOXID (x, y and box ID of any dynamic box in the model; by convention set to the values for the box with the larget market port) Econts0.data econ_file1.ts (path and filename of the TS file) Econts0.rewind 0 (whether to rewind the data file (1) or not (0))

9. PRIMARY PRODUCER PROCESSES

The main Atlantis model has six different types of primary producers, including two sizes of phytoplankton (large phytoplankton LG_PHY and small phytoplankton SM_PHY), mixotrohic dinoflagellates (DINOFLAG), microphytobenthos (MICROPHTYBENTHOS) and two epibenthic groups (seagrasses SEAGRASS and other phytobenthos PHYTOBEN). In models including ice, specific ice primary producers are modelled. Simialrly in models that dynamically represent land, the user can define terrestrial vegetation (primary producer) groups, which are restricted to land (e.g. MANGROVE, MARSH, with crops and agriculture also available as in industry if the land use model option is used). The description below will focus on the main model only.

9.1. Fluxes in primary producers

In primary producer pools the flux is determined by growth (G_{PP}), natural mortality (M_{PP}) and grazing (predation) ($Gr_{PP,i}$). In addition, there is an optional encystment at certain times of the year or when temperatures (Ec_{out}) reach a certain trigger level. Transfer back from cysts occurs when conditions are suitable (Ec_{in}).

Four types of primary producer groups are tracked separately in both the water column and sediments (SM_PHY, LG_PHY, DINOFLAG, MICROPHTYBENTHOS). Two groups are restricted to the epibenthic layer (PHYTOBEN, SEAGRASS). The general flux equation is

$$\frac{d(PP)}{dt} = G_{PP} - M_{PP} - \sum_{i=predators} Gr_{PP,i} - Ec_{out} + Ec_{in}$$

NOTE! Not all processes are executed in all layers.

Primary producer GroupType processes in water column, sediments and epibenthos

For SM_PHY, LG_PHY, DINOFLAG:
In the water column: the processes executed are growth (primary production) and mortality due

to lysis and grazing.
In the sediments: there is no primary production (only mixotrophic production in DINOFLAG), no lysis mortality and no grazing, only linear mortality.

For MICROPHTYBENTHOS:
In the water column and sediments: growth (primary production) and mortality due to grazing are carried out in both locations, with mortality due to lysis in water column or linear mortality in sediments.

For SEAGRASS and PHYTOBEN:
In epibenthic layer: growth (primary production), linear and extra mortality and mortality due to grazing are all executed.

NOTE!

Role and origin of microphytobenthos in Atlantis

Most ecosystem models do not explicitly model microphytobenthos, which consists of small epibenthic primary producers. The Port Philip Bay Integrated Model (PPBIM) (Muray and Parslow 1997), which is used as a basis for simulating lower trophic level dynamics in Atlantis, included microphytobenthos, because it was well studied in the Port Phillip Bay and its production was shown to be significant. Other estuarine models also use microphytobenthos given its important role in those kinds of ecosystems. However, for the majority of coastal Atlantis models the group is not used due to the limited amount of data available for parameterisation and the deeper water depths, where it plays a less important role.

9.2. Primary producer growth

The equations and processes governing primary production in Atlantis have been adopted from the PPBIM model of Murray and Parslow (1997). Primary production processes are executed by *Primary_Production()* routine in **atprocess.c**

Growth of a primary producer *pp* is determined by **multiplying** the biomass of *pp* by the maximum specific growth rate (*mum*) and by three additional potentially limiting factors:

1) nutrient limitation factor
2) light limitation factor
3) space limitation factor for benthic macrophytes
4) optional eddy scalar, representing enhancement of primary production by eddies
5) optional pH scalar, accounting for possible effects of acidification on primary production

$$G_{pp} = mum \cdot B_{PP} \cdot \delta_{light} \cdot \delta_{nutrient} \cdot \delta_{space} \cdot \delta_{eddy} \cdot pHscalar$$

The *mum* parameter defines the maximum growth rate per day (mgN day^{-1}).

NOTE!
Multiplicative model for nutrient and light co-limitation (from Murray and Parslow, 1997, page 26):

"Multiplicative models for nutrient and light co-limitation have been widely used (Fasham et al. 1990). A potential problem with multiplicative interactions involving several factors is that predictive growth rates become very low when each factor is moderately limiting. An alternative, Liebig's law of the minimum (ie. multiply mum by the smallest factor) is more often applied to interactions among multiple nutrients."

Atlantis has different options to design limitation by different nutrients (N, Si and P if phosphorus is tracked), set by the flagnut parameter, with choices of multiplicative or Leibig (minimum nutrient) limitation. However, this does not apply to combined limitation by nutrient, light and space, where simple multiplication is used. The quote above suggests caution when applying multiple limitations on growth, and careful assessment of whether the multiple impacts do not lead to overly constraining limitation.

9.2.1. Primary producer nutrient limitation

The **nutrient limitation** $\delta_{nutrient}$ scalar in case of limitation based on N only is calculated as

$$\delta_{nutrient} = \frac{DIN}{KN + DIN}$$

where **DIN = NH3+NO3** (ammonia and nitrate concentration) and *KN* is the half saturation constant of nutrient uptake (KN_XXX) in mgN m^{-3}

Nutrient limitation is calculated by the *Nutrient_Lim()* routine in **atprocess.c**

When phytoplankton is limited by multiple nutrients, such as N and Si in the case of diatoms, the choice of limitation equation is set by the flagnut parameter. This parameter calls three types of nutrient limitation options – Leibig, multicative, and the ERSEM WQI based option. Only the first two options have been more widely used, whereas the third version has been implemented only for testing purposes. The nutrient limitation scalar for the three options is calculated as:

$$\delta_{nutrient} = \min\left(\frac{DIN}{KN + DIN}, \frac{Si}{KS + Si}\right) \qquad \text{if Leibig limitation is used } (flagnut = 0)$$

$$\delta_{nutrient} = \sqrt{\frac{DIN}{KN + DIN} \cdot \frac{Si}{KS + Si}} \qquad \text{if multicative limitation is used } (flagnut = 1)$$

$$\delta_{nutrient} = {}^{2}\Big/\left(\frac{DIN}{KN+DIN}+\frac{Si}{KS+Si}\right) \qquad \text{if WQI limitation is used (flagnut} = 2)$$

For illustration, assuming a hypothetical case of DIN and Si concentrations of 100 mg m^{-3} and half saturation values for N and Si at 50 mg m^{-3} and 20 mg m^{-3} the three limitation cases would results in scalars of 0.67, 0.75 and 1.41.

Atlantis has an optional nutrient limitation for one additional tracer of micronutrients (currently used for iron). This option is set when flagmicro=1. If micronutrient limitation is used, half-saturation constants (KF_XXX) should be provided for all primary producer groups. Limitation by additional nutrients will be done according to the flagnut option described above.

In models that track P and C (TrackAtomicRatio =1 in *run.prm* file) primary production will also be limited by P and C. Further details are available on the wiki.

9.2.2. Primary producer light limitation

Photosynthesis in the model is potentially light-limited and the available suitable light (W m^{-2}) – the photosytenthically active radiation (PAR) - in the water column and at the sediment surface is modelled explicitly. The **light limitation** δ_{light} factor is calculated in *Light_Lim()* in **atproces.c** as:

$$\delta_{light} = \min\left(\frac{IRR}{KI}, 1\right) \qquad \text{if light adaptation is not used (flaglight} = 0)$$

where *IRR* is the amount of light available for the photosynthesis and *KI* is the light saturation coefficient (KI_XXX) in W m^{-2}.

Atlantis allows for **light adaptation** of primary producers to be represented, which is intended to capture their ability to rapidly adapt to different light conditions. It is used when flaglight = 1 and requires four additional parameters (KIOP_min, KIOP_shift, KI_avail, K_addepth), used to calculate *optIRR* in *Box_Light_Process()* in **atbiophysics.c**

If light adaptation is used the light limitation is calculated as

$$\delta_{light} = \min\left(\frac{IRR}{KI+optIRR}, 1\right) \qquad \text{if light adaptation is used (flaglight} = 1)$$

The amount of light falling on the surface of the top layer is either provided through external time series or calculated in Atlantis depending on the lim_sun_hours setting (described in chapter 5, Physics).

The **light attenuation** *Kd* through the water column (described in Fulton 2001) defines the availability of light to primary producers at different depths and is calculated as

$$Kd = k_w + k_{IS} + k_{PN} \cdot \sum_{i=phytopl} B_i + k_{DON} \cdot B_{DON} + k_{DL} \cdot (B_{DL} + B_{DR}) + k_{SED} \cdot B_{SED}$$

The equation shows that light attenuation is simply defined by the background blue water light absorption coefficient (k_w), additional light absorption due to tanins and other non-biological particulate matter that is important in coastal areas (set in k_{IS}), and concentration or biomass of light attenuating organisms (plankton) or substances (DON, detrituts, sediments) multiplied by the corresponding light attenuation coefficient. The attenuation is caused by:

1) background light absorption by water itself (k_w) – used in the PPBIM model.
2) additional background light absorption by water due to tanins and other non-biological particulate matter that is important in coastal areas (set in k_{IS}). Different coefficients (k_w_shallow and k_w_deep) are used for coastal and oceanic water. This second term of light absorption was not used in PPBIM but was taken from ERSEM
3) phytoplankton, with k_PN defining coefficient of light absorption by particulate matter, such as phytoplankton
4) dissolved organic nitrogen and the relevant coefficient k_DON
5) labile and refractory detritus and the coefficient k_DL
6) light absorption by suspended sediment k_SED, used only if the model is estuary (flagIsEstuary=1)

9.2.3. Primary macrophyte producer space limitation

Space limitation is only applied to epibenthic primary producers, defined by PHYTOBEN or SEAGRASS

The **space limitation** δ_{space} factor is calculated as

$$\delta_{space} = \frac{B_{PP}}{SPmax \cdot P_{area}}$$

where *SPmax* is the coefficient setting maximum biomass of the epibenthic primary producer (XXmax) in mgN m^{-2} and P_{area} is the proportion of the cell covered with suitable habitat scaled by the habitat degradation scalar (if used).

9.2.4 Effect of eddies on primary production

The effect of eddies, or the **eddy scalar** δ_{eddy}, on primary production is modelled by simple scaling (multiplication) using the user defined eddy_scale parameter. This dimensionless parameters is multiplied by the eddy strength in a given box to obtain δ_{eddy}

$$\delta_{eddy} = \text{eddy}_{\text{scale}} \cdot \text{eddy}_{\text{strength}}$$

Eddy strengths for each box and each quarter are set in the *physics.prm* file (see Chapter 5.2, section

III).

NOTE!

Effect of eddies on primary production

There is good evidence to suggest that the presence of eddies significantly enhances primary production through vertical water column mixing creating spatial and seasonal variation. There are two ways to add this variation in Atlantis and users are encouraged to make use of these parameters.

First, the effects of eddies on vertical mixing alone can be included through the parameters in the *physics.prm* (eddy_vertmix, eddy_S1 to eddy_S2, eddy_mixscale – see chapter 5.2, section III). This should increase vertical mixing in the water column and bring extra nutrients to the surface.

Second, the user can add an extra scalar δ_{eddy} on primary production through the eddy_scale parameter in the *biology.prm* file. This scalar will take the eddy values and re-scale the primary production calculated for the available amount of light and nutrients based on eddy strength. The second approach assumes that additional nutrients are implicitly supplied through eddies, increasing primary production (these nutrients do not appear in the tracer values but represent just an assumed increase). This *ad hoc* approach can be taken if the model fails to accurately reflect spatial and seasonal variation in primary production, but the explicit dynamics of nutrient turnover is not resolved accurately enough to enhance primary production sufficiently through nutrient mixing alone. It is not recommended for models that focus on nutrient dynamics and lower trophic levels, but might be useful for models that focus on the dynamics at higher trophic levels and socio-economic aspects.

9.3. Primary producer mortality

NOTE!

Primary producers in water column, sediments and epibenthos

Primary producer processes are handled differently depending on whether they are in the water column (WC), sediments (SED) or epibenthic layer (EPI).

Four primary producer groups are tracked both in the WC and SED (SM_PHY, LG_PHY, DINOFLAG, MICROPHTYBENTHOS). Two groups are restricted to the epibenthic layer (PHYTOBEN, SEAGRASS)

In WC: phytoplankton groups (SM_PHY, LG_PHY) can only grow (photosynthesise) when they are in the WC. MICROPHTYBENTHOS can grow both in the WC and in SED; for photosynthesis in the SED they use light conditions at the top of the sediment. DINOFLAG do not photosynthesis in the SED, but they can have mixotrophic feeding. Mortality expressed depends on whether it is in the WC and SED: primary producers are affected by linear mortality (mL_XXX) and optional acidification mortality in both locations, but phytoplankton (SM_PHY and LG_PHY) only suffer lysis (KLYS_XXX) in the WC. DINOFLAG and MICROPHYTOBENTHOS have all mortality terms in both locations. Both lysis and other mortality terms are separately scaled by the ***external forced optional mortality scalar***. All mortality products are transferred into the WC or SED labile detritus pools respectively.

In EPI: epibenthic primary producers are affected by linear mortality (mL_XXX), extra mortality due to waves or fouling (mS_XXX) and optional acidification mortality. Mortality products are transferred to the SED labile and refractory detritus pools; the split between labile and refractory detritus is determined by the FDL_benthos parameter.

Primary producer mortality in the **water column and sediments** is modelled as

$$M_{PP} = \left(\left(\frac{KLYS \cdot B_{PP}}{\delta_{nutrient} + 0.1}\right) + (mL \cdot B_{PP} + mA \cdot B_{PP})\right) \cdot mortsc$$

Primary producer mortality in the **epibenthic layer** is modelled as

$$M_{PP,EPI} = (mL \cdot B_{PP} + mA \cdot B_{PP} + mS \cdot B_{PP}) \cdot mortsc + F_{PP}$$

where mL (mL_XXX, day^{-1}) is the linear mortality rate, $KLYS$ is the lysis rate (KLYS_XXX, day^{-1}), mA is the optional acidification mortality, mS is the extra mortality of large epibenthic primary producers calculated below, B_{PP} is the biomass of the primary producer, $mortsc$ is the external mortality scalar (from forcing files) and F_{PP} is an optional fishing mortality that is sometimes applied to macroalgae.

The **extra mortality *mS*** in **large primary producers** includes mortality due to waves and fouling:

$$mS_{MA} = STRESS \cdot mS\_MA$$

$$mS_{SG} = DIN \cdot mS\_SG$$

where *mS_MA* is the algal mortality rate (day^{-1}) due to abrasion by bottom stress or waves (*STRESS*), provided through external forcing or calculated in the model (see Chapter 5, Physics) and *mS_SG* is the seagrass mortality rate (day^{-1}) due to fouling by epiphytes in water of high nutrient concentration, and *DIN* is the concentration of NH3 and NO3.

NOTE!
What is the difference between lysis and linear mortality?

Lysis represents cell leakage and is inversely related to nutrient concentration, whereas linear mortality is just a constant mortality term. PPBIM only had the linear mortality, but ERSEM also included lysis, and both options are included in Atlantis. The users have an option to use either or both, depending on the mortality parameter values set.

Finally, the extra mortality will also include any **optional acidification induced mortality**, described in details here and here and in Chapter 13. The acidification mortality is only included if the model explicitly includes pH effects (flagmodelpH=1) and primary producers are identified as pH sensitive (flagpHsensitive_XXX=1).

Primary producer mortality is handled by *Primary_Production()* in **atprocess.c** and *Phytoplankton_Process()* in **atGroupProcess.c** routines.

9.4. Encystment of primary producers

It is possible to include the encystment of primary producers, triggered by time of the year or temperature. This option is only modelled for groups that have isOverWinter set to 1 in the *functional_group.csv* file and is described in further detail on the wiki here.

9.5. Effect of temperature and salinity on primary production

Both light saturation (KI_XXX_T15) and maximum growth rate (mum_XXX_T15) are dependent on **temperature** and will be scaled by *Tcorr* calculated based on the current water temperature as described in chapter 13.

The **salinity** effect on light saturation and maximum growth rate are optional, where the *Scorr* is applied only when:

1) an organism is identified as sensitive to salinity, with flagSaltSensitive_XXX
2) an organism an outside the salinity limits defined with XXX_min_move_salt and XXX_max_ move_salt

If the two conditions are fulfilled, the *Scorr* scalar provided by the user as salt_correction_XXX parameter, is applied to KI_XXX_T15 and mum_XXX_T15 (along with the *Tcorr* described above).

9.6. Growth of mixotrophic primary producers

Atlantis includes a mixotrophic dinaflagelate group DINOFLAG, the growth of which is determined by both primary production and heterotrophic consumption.

Phtosynthetic growth of DINOFLAG is as for other primary producers, although there is some increase in efficiency at low light levels, represented here by increasing effective light available via the following equation (which is only used if light levels drop below 10% of KI_XXX):

$$IRR_{sp} = KI_{XXX} \cdot \ (0.01 \cdot IRR + 0.018)$$

Mixotrophic growth due to consumption is calculated by first determining potential grazing (using the standard feeding routine applied to consumers). Phagotrophic growth then tops-up photosynthetic growth to the maximum possible rate, as if nutrients were non-limiting, unless the maximum growth rate is higher than what could be physically grazed. In this way grazing leads to additional growth compared to photosynthesis alone. Unlike in other consumers mixotrophic groups have no excretion (production of NH) as the nutrients are used in the photosynthetic growth pathway.

Nutrient stress related lysis of DINOFLAG is also corrected to account for the mitigating effects of phagotrophy), with the corection dictated by the proportion of photosynthetic nutrient made up by the grazing.

Table 9.1. Parameters used for primary producer processes

Parameter	Description
Key parameters defining primary production and primary producer mortality	
KN_XXX	Half-saturation constant for XXX growth on DIN (mgN m^{-3})
KS_XXX	Half-saturation constant for XXX growth on Si (mgN m^{-3}) – only important for Si limited groups, as defined in the *functional_group.csv* file
KI_XXX_T15	Light saturation of XXX at 15C (W m^{-2}) - this parameter is temperature dependent (and potentially salinity dependent), and a scalar is applied based on current temperature (and salinity) conditions.
mum_XXX_T15	Maximum primary producer XXX growth rate at 15C (mgN mgN^{-1} day^{-1}) – this parameter is temperature dependent (and potentially salinity dependent), and a scalar is applied based on current temperature (and salinity) conditions.
XXmax	Maximum biomass of large epibenthic primary producers (phytobenthos and macroalgae) (mgN m^{-2})
mL_XXX	Linear mortality of primary producers (day^{-1})
KLYS_XXX	Lysis rate of primary producers (day^{-1})
mS_XXX	Extra mortality of macroalgae (PHYTOBEN) due to wave action (day^{-1})
mS_XXX	Extra mortality of seagrasses (SEAGRASS) due to fouling (day^{-1})
Further modification of primary production	

flaglight	Flag indication whether light adaptation in primary producers is activated (0=no, 1=yes), It set to 1 (i.e. activated) it will increase production at low light levels, but it will also require four additional parameters: KIOP_min, KIOP_shift, KI_avail, K_addepth (see SETas model example of *biology.prm* file)
flag_macro_model	Flag indicated whetehr a more detailed seagrass model is activated, see details here (0=no, 1=yes). If the flag is set to 1 then three separate pools will be initiated and tracked for each SEAGRASS group type: main biomass, epiphyte biomass and below ground biomass
flagnut	Nutrient limitation form used if primary production is limited by several nutrients (0=Leibig's law where limitation is set by the most limiting nutrient; 1= multiplicative; 2=WQI)
flagmicronut	Flag indications whetehr there is miconutrient limitation (0=no, 1=yes) If set to 1, half-saturation constant for gthe micronutrient (typically Fe) is required (KF_XXX)
eddy_scale	Coefficient scaling eddy strength impact on phytoplankton growth
swr_scalar	Proportion of shortwave radiation available to photosynthesis – usually set to 0.45
MB_wc	Scalar for microphytobenthos growth in the water column – growth should be strongly reduced as it is not on the substrate (which is its primary habitat). Only important when microphytobenthos is being modelled.

If TrackAtomicRatio is set to 1 in *run.prm* file and the model tracks N:P:C many additional parameters are required to describe P, and C effect on primary production. See details here and sample SETas model *biology.prm* file

---

10. CONSUMER PROCESSES

Consumers in Atlantis are modelled as biomass pools, age-structured biomass pools or age-structured groups. The age-structured groups are typically used for vertebrates and in the text below the terms age-structured groups and vertebrates may be used interchangeably. Non-vertebrates are largely modelled as biomass pools, which in age-structured biomass pools can be explicitly subdivided into ontogenetic stages (similar to stanzas in Ecosim).

The group types FISH, MAMMAL, SHARK, BIRD and FISH_INVERT are automatically assigned to age-structured groups. Other group types that have NumCohort=1 in the *functional_group.csv* file are modelled as biomass pools, and if NumCohort>1 they are modelled as age-structured biomass pools, where NumCohort parameter determines how many biomass pools to model.

10.1. Fluxes in consumer biomass pools

In **biomass pool or age-structured biomass pool consumers** (CP in the following equations) the only variable tracked is N (i.e. not Si or the micronutrient; although when P and C tracking is enabled

then N, P and C are all tracked for consumers). The flux through a consumer biomass pool is determined by growth (G_{CP}), natural mortality (M_{CP}), grazing (predation) for *i* predators ($Gr_{CP,i}$) and optional fishing mortality (F_{CP}). In addition, there is an optional encystment out of the system at certain times of the year or temperatures (Ec_{out}) and transfer back from cysts into the system when conditions are suitable (Ec_{in}).

Biomass pool consumers maybe in the sediment, epibenthic layer or water column (depending on group type). A single N (P, C) pool is tracked per biomass pool age group (see Table 6.1):

$$\frac{d(CP)}{dt} = G_{PP} - M_{PP} - \sum_{i=predators} Gr_{PP,i} - F_{CP} - Ec_{out} + Ec_{in}$$

In **age-structured consumers** (abbreviated as CX) the nitrogen pool is partitioned into the reserve (RN) and structural (SN) nitrogen of an average individual of each age group. Atlantis also tracks the numbers (Num) of individuals per age group. As a result for each age group three state variables are tracked – RN, SN and Num.

Fluxes through the RN and SN pool are determined by growth (G_{CX}), proportion of G_{CX} allocated to SN versus RN (λ) and spawn (Sp_{CX}), which is taken out of the RN pool on spawning (with timing defined in the *biology.prm* file, but typically once per year). Details determining λ are given in chapter 10.5.1. Note, that if respiration is included, then growth (G_{CX}) can also be negative, and all energy required to cover respiration (or maintenance) costs will be taken out from the RN pool (see chapter 10.4)

$$\frac{d(RN_{CX})}{dt} = G_{CX} \cdot \lambda - Sp_{CX}$$

$$\frac{d(SN_{CX})}{dt} = G_{CX} \cdot (1 - \lambda)$$

The equations above show that RN pool can change through the year as it will suddenly decrease after the spawn. The SN pool cannot decrease.

The flux of Num is tracked through individuals. The number of individuals in an age group depends on recruitment or ageing into the age group ($Nm_{CX\text{-}1,\,ageup}$), numbers ageing up into the next group ($Nm_{CX,\,ageup}$), natural mortality (M_{CX}), predation from *i* predators ($Gr_{CX,i}$), optional migration out (T_{CXout}) and into (T_{CXin}) the model domain and optional fishing mortality (F_{CX}).

$$\frac{d(Nm_{CX})}{dt} = Nm_{CX-1,ageup} - Nm_{CX,ageup} - M_{CX} - \sum_{i=pred} Gr_{CX,i} - F_{CX} - T_{CXout} + T_{CXin}$$

The numbers are converted to biomass as (SN + RN) · Nums.

10.2. Consumer feeding

NOTE!

What determines growth in consumers?

Growth in biomass pool consumers is determined by the intake or grazing (Gr) and assimilation efficiency. The consumed food or grazing term (Gr) is then assimilated according to the assimilation efficiencies for different food types (live, plant, labile detritus and refractory detritus). The unassimilated food is sent to the labile detritus, refractory detritus and ammonia pools according to the parameter determining detritus allocation (chapter 10.7), whereas all assimilated food is sent to growth.

Growth in age-structured consumers is determined by the intake or grazing (Gr), assimilation efficiency and optional maintenance or respiration costs (Rs). The consumed food or grazing term (Gr) is then assimilated according to the assimilation efficiencies for different food types (live, plant, labile detritus and refractory detritus). The unassimilated food is sent to the labile detritus, refractory detritus and ammonia pools according to the parameter determining detritus allocation (chapter 10.7). The assimilated food represents a temporary energy pool (A). This A pool is used to meet the optional maintenance or respiration costs (Rs) and all remaining energy is allocated to SN and RN, i.e. to growth. The proportion of energy going into SN and RN is determined by λ, which is calculated at each time step dynamically depending on the individual's condition and parameters determining energy allocation rules. The SN pool cannot decrease, whereas the RN pool can decrease, as it is used for reproduction and for meeting optional maintenance needs if assimilated food is insufficient.

The following chapters will describe these key processes: grazing (Gr), assimilation (A), maintenance (Rs) and energy allocation to SN and RN.

10.2.1 Grazing

Grazing or predatory interactions are modelled in a similar way for both biomass pools and age-structured groups, except that age-structured and biomass pool groups have different options for refuge from predation. Feeding interactions are modelled through biomass, which in age-structured groups is then converted to numbers to track individual mortality. In Atlantis predatory interactions are determined by:

1) Physical overlap – prey and predator must be in the same cell at the same time (determined by vertical and horizontal distribution parameters), and if prey is in the sediment the predator must be able to reach it.
2) Diet connection matrix (pPREY matrix or detailed ontogenetic diet preferences) that indicate maximum availability of prey to a predator. The actual realised consumption will depend on refuge factors, but if the value in the pPREY matrix is set to 0, no predation will occur.

3) Gape limitation for age-structured prey – prey that is too small or too big for the predator (either age-structured or biomass pool) will not be consumed.
4) Habitat refuge.
5) Environmental factors (temperature, salinity, pH) that can modify predator's feeding rates, prey's availability and nutritional content.

Throughout this chapter for brevity we will refer to both biomass pool (CP) and age-structured (CX) consumers as CX, as in many cases the processes are modelled in the same way for both groups; processes that are modelled differently will be identified throughout.

At the time of writing, Atlantis has twelve different options for modelling a predator's feeding functional response to the prey's biomass. There is a lot of theory and debate on what kind of functional response most accurately depicts predator's behaviour (e.g. Hunsicker et al. 2011) and this topic is not covered here.

Currently most Atlantis models use the modified Holling type II response, described in Murray & Parslow (1997) for the Port Phillip Bay Integrated Model. The grazing term (*Gr*), or the amount of biomass of a specific prey (*prey*) consumed by a consumer CX (note, this now includes both CP and CX consumers), in the modified Holling type II response is calculated as

$$Gr_{prey} = \frac{B \cdot C \cdot B^*_{prey}}{1 + \frac{C \cdot \sum_i \left(E_i \cdot B^*_{prey,i}\right)}{mum}}$$

where

$$B^*_{prey} = p_{prey,CX} \cdot \delta_{overlap} \cdot \delta_{habitat} \cdot \delta_{size} \cdot B_{prey}$$

is the available biomass of *prey* in the cell after taking into account all refuge (δ) options: $\delta_{overlap}$, $\delta_{habitat}$ and δ_{size}. The refuge options are described in detail in the next chapter.

B is the feeding biomass of a consumer (predator) CX in a given cell (mgN m^{-3}) – a proportion of the consumer biomass may not be feeding if it is not active at that time of day or if it is spawning and has been identified as a group that does not feeding while spawning (feed_while_spawnXXX=0). B_{prey} is the biomass of prey (or fisheries catch available for consumption) in that cell (mgN m^{-3}), $p_{prey,CX}$ is the maximum availability (ranging from 0 to 1) of the *prey* to the predator CX defined in the pPREY matrix (or optional ontogenetic diet matrices), *C* is the clearance rate, similar to a search volume of the consumer CX, and *mum* is the maximum consumption rate, sometimes referred to as maximum growth rate, of the consumer CX.

The units of *C* and *mum* in CP consumers are m^3 mgN^{-1} day^{-1} and mgN mgN^{-1} day^{-1}. In CX consumers the units depend on the flagfishrates parameter, which defines whether rates are absolute or per unit of consumer biomass. If rates are absolute (flagfishrates=0) then *C* units are m^3 ind^{-1} day^{-1} and *mum* units are mgN ind^{-1} day^{-1}. If flagfishrates=1 then *C* units are m^3 mgN^{-1} day^{-1} and *mum* units are mgN mgN^{-1} day^{-1}.

The E_i is the assimilation efficiency of consumer CX on prey group category *i,* which represents the four prey group categories defined in Atlantis: living prey and fisheries catches, plant prey, labile detritus and carrion (ie. fisheries discards) and refractory detritus. **Note that carrion is included in the labile detritus food category**, and the EDL assimilation efficiency for species that feed on carrion should be higher than what it would be if the species was feeding on labile detritus alone.

NOTE!
Use of assimilation efficiency E_i in the grazing term

Even though assimilation efficiency is included in the grazing term *Gr* it **does not** define the final assimilated food, but is only used **in modified Holling feeding** responses (see Chapter 10.3) to scale the maximum consumption rate of a consumer. **Higher assimilation efficiencies lead to lower Gr value**, as the predator need not intake as much food as it can extract more energy from what it does consume.

NOTE!

Should I use absolute or mass-specific* C *and* mum *rates (*flagfishrates)*?

The flagfishrates parameter defines whether *mum* and *C* parameters are given per individual or per unit of body weight. It is important to remember that in Atlantis the realised size-at-age is dynamic and will change through time, depending on food availability. Setting flagfishrates to 0 or using absolute *C* and *mum* values means that regardless of the realised size-at-age *C* and *mum* values of an age group will stay the same throughout the simulation and Atlantis will behave more like an age-structured model. Using mass specific values (flagfishrates=1) means that *C* and *mum* will vary from year to year depending on the size of individuals in an age group, and behaviour of Atlantis will be closer to the size-structured models.

Regardless of whether absolute or mass-specific rates are used, separate *C* and *mum* parameters are required for each age group, so that C_XXX and mum_XXX strings in the *biology.prm* file contain as many values as there are groups in an age-structured functional group XXX. This means that Atlantis allows users to assume that the relationship between *C* and *mum* and body weight can change with ontogeny.

10.2.2 What are the C *and* mum *parameters?*

Some Atlantis users have been confused about the meaning of *C* and *mum* parameters in the modified Holling representations. The *mum* parameter was initially adopted by Murray & Parslow (1997), who replaced the standard Holling Type II maximum ingestion rate parameter (inverse of handling time) with a "maximum growth rate" *mum* by scaling it by assimilation efficiency:

Origin of *mum* from Murray and Paslow 1997 (page 29):

"In a classical paper, Holling (1966) discussed the theoretical and empirical basis of the functional response relating ingestion per consumer (G) to food density (P). He identified three functional forms:
I) G = min (C*P, Gmax) (bilinear),
II) G = C*P / (1 + C*P/Gmax) (rectangular hyperbola),
III) G = $C*P^2$ / (1 + $C*P^2$/Gmax) (sigmoid or switching).

The parameter Gmax represents a maximum (food saturated) ingestion rate at high food densities, while C is a measure of grazing efficiency at low food densities. In Type I and II, the parameter C has units of volume cleared per grazer per unit time, and represents a maximum clearance rate.
…
In this model, we have used the rectangular hyperbola without thresholds, Type II, as the standard formulation. However, rather than specify the maximum ingestion rate Gmax, we have specified the maximum growth rate, "mum". This is related to Gmax through growth efficiency E, so that mum = Gmax*E, or Gmax = mum/E."

The growth efficiency described above is the same as assimilation efficiency in Atlantis.

NOTE!

What is *C*?

C is often referred to as a clearance rate and it is defined as the volume of water searched, but it is different from an often used gut clearance rate that reflects food assimilation efficiency. The *C* parameter in Atlantis has units of m^3 ind^{-1} day^{-1} (or m^3 mgN^{-1} day^{-1}) and reflects the search or "cleared" volume. Note, however, that *C* is treated differently from true search volume (vl_XXX, vla_XXX, vlb_XXX) parameters used in predcase=5.

The true search volume parameters vl_XXX (for biomass pools) and vla_XXX and vlb_XXX (for age structured groups) are by default mass-specific and the relationship between body weight and search volume does not change. This means that only one entry is required, rather than a separate entry for each age group.
Further, for demersal species (flagdemXXX =1 in the *biology.prm* file) the search volume is multiplied by 0.5, but this scaling is not applied to the *C* parameter.

And what is *mum*?!

The *mum* parameter is only used in the modified Holling feeding responses (modified after Muray and Parslow, see above). Even though *mum* is called a maximum growth rate, it is included in the grazing (feeding) and not the growth equation, so it does not define the final growth. The consumed food will first be assimilated and the actual growth will then depend on other factors, such as respiration, reproduction and energy allocation to SN and RN.

As described in Muray and Parslow (1997) mum relates to the maximum ingestion rate (*Gmax*) as *mum = Gmax*E*, where *E* is the assimilation efficiency. The maximum ingestion rate is in turn the inverse of a more familiar "handling time" parameter (*ht*). Hence *mum* can also be seen as an inverse of the handling time as *mum=E/ht*

There is no clear consensus among currently used models as to the exact relationship between *C* and *mum* parameters. Theoretically the *mum* value should always be larger than *C*, but in some models and for some species the opposite is true. This is explored in Figure 10.1, which shows values of Gr_{prey} for different prey biomass, and different *C* and *mum* rates.

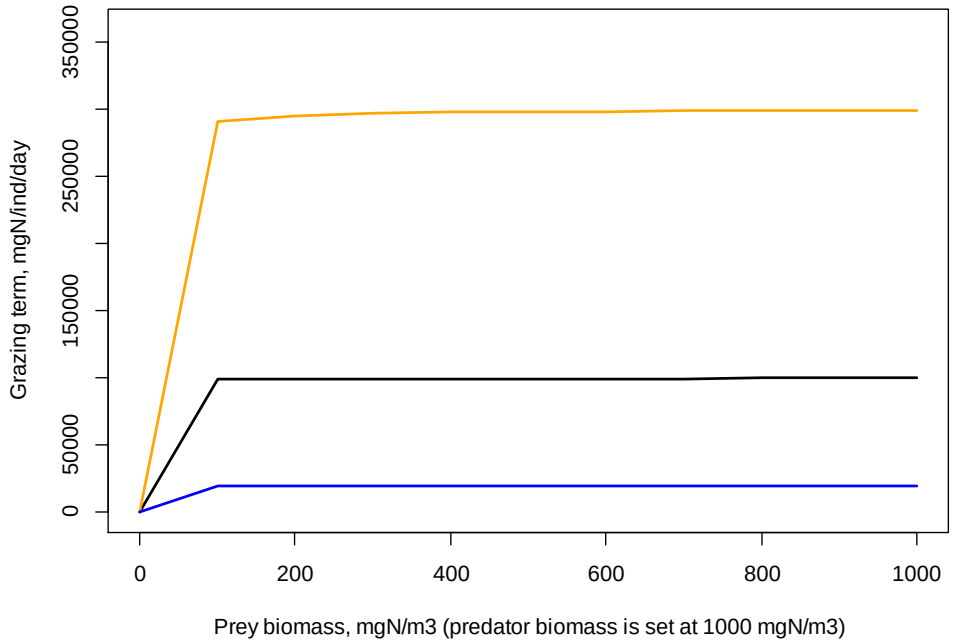


Figure. 10.1. The value of grazing term (*Gr*) as a function of prey biomass (ranging from 0 to 1x of the predator biomass) and different assumptions on the ratio between *C* and *mum*. Predators biomass is 1000mgN, *C* = 100 m^3 ind^{-1} day^{-1}. The calculations assume only one prey, no refuge (δ_{refuge}=1), maximum availability $p_{prey,CX}$ of 1, and assimilation efficiencies $E_{prey,i}$ of 1.
Black: *mum* = *C*, **orange**: *mum* = *C**3, **blue**: *mum* =*C*/5

NOTE!

Ensuring meaningful prey choice and diets

Atlantis **does not** implement optimal prey choice. The clearance rate *C* is specific to each predator but identical for all prey. Once all the available prey is assessed (based on overlap, size and habitat refuge) the consumption is applied **uniformly across all available prey groups proportionally to the available prey biomass** (prey biomass · pPREY). This means that if the available prey includes 1000mg of clams and 1mg of fish, and the clearance rate determines that only 10% of all available biomass can be consumed for a given time step, the predator will ingest 10% of available clam biomass and 10% of available fish biomass or 100mg of clams and 0.1mg of fish.

This has important implications for optimising the maximum available biomass $p_{prey,CX}$ in the prey availability matrix. The availability of biomass pool and especially detritus prey should be low for predators that prefer to eat fish. Otherwise a predator, such as a seal, might entirely fill up on invertebrate prey and impose no top-down control on fishes. Note, this can happen even if the availability of invertebrates to seals is as low as 0.001, as the final proportion in the diet is determined by availability and the biomass in the box. It is of crucial importance to **carefully inspect realised diets** from *DietCheck.txt* and *DetailedDietCheck.txt* output files, which can be analysed using for example atlantistools R package or other tools described in chapter 2.9.

Table 10.1 General feeding parameters and units

Parameter	Description
Key parameters for modified Holling Type II functional response	
predcase_XXX	Flag indicating the feeding functional response to use for a predator XXX
flagfishrates	Flag defining whether fully age-structured C_ and mum_ parameters are absolute (given per individual) or mass-specific (per unit of body mass)
mum_XXX_T15	Maximum ingestion rate of biomass pools at 15C
C_XXX_T15	Clearance rate of biomass pools at 15C
mum_XXX	Age specific maximum ingestion rate for age structured groups: mgN ind^{-1} day^{-1} or mgN mgN^{-1} day^{-1}
C_XXX	Age specific clearance rate for age structured groups: m^3 mgN^{-1} day^{-1} or m^3 ind^{-1} day^{-1}
pPREY1XXX1 pPREY1XXX2 pPREY2XXX1 pPREY2XXX1	Maximum availability of prey to a predator XXX (given as a vector with the column the prey group in the same order as in the functional_groups.csv file). Four values are given for each predator-prey combination: juv-juv, juv-ad, ad-juv, ad-ad.
E_XXX	Assimilation efficiency when feeding on animal prey
EPlant_XXX	Assimilation efficiency when feeding on plant prey
EDR_XXX	Assimilation efficiency when feeding on refractory detritus
EDL_XXX	Assimilation efficiency when feeding on labile detritus
Parameters for optional further details	
flag_fine_ontogenetic_diets	Flag indicating whether more refined, age-specific, prey availability should be used (so an availability value is required per predator age class rather than per stage (typically juvenile and adult))
p_split_YY	Flag identifying prey groups YY for which age-specific diets should be given
age_structured_prey_XXX	Flag identifying predators for which age-specific diets should be given
p_YYXXX	A vector of maximum prey availability for each combination of prey YY

	and predator XXX identified in p_split_YY and age_structured_prey_XXX parameters

10.2.3. What determines prey biomass available for predation?

The proportion of prey biomass available to a predator at any given time is determined dynamically as

$$B^*_{prey} = p_{prey,CX} \cdot \delta_{overlap} \cdot \delta_{habitat} \cdot \delta_{size} \cdot B_{prey}$$

This means that the available prey biomass depends on:

1) Static parameter $p_{prey,CX}$ (which can range from 0 to 1) defined in the pPREY matrix (or optional ontogenetic diet matrices)
2) The $\delta_{overlap}$ refuge defined by co-occurrence in the same cell at a same time step and the predator is active at that time of day. This is determined by horizontal and vertical distribution as well as activity parameters.
3) The $\delta_{overlap}$ refuge also defined by the predator's ability to access at least one of the habitats inhabited by prey. This is determined by the **static** habitat_XXX parameter and is activated only if a secies is habitat dependent (see below)
4) The δ_{size} refuge for age structured prey. This is determined by gape limitation of the predator relative to the prey size.
5) The $\delta_{habitat}$ refuge for infaunal (_INF) invertebrates (CX) defining access to biomass pool prey buried in the sediment. This is determined by the depth of the oxygen layer and depth predators can dig into
6) The optional $\delta_{habitat}$ refuge for age structured prey. This is determined by optional habitat dependency and habitat refuge parameters.

NOTE!

What is $p_{prey,CX}$?

This parameter defines the maximum proportion of the prey biomass available to a consumer at a given time. If < 1 then it means that at any given time the consumer cannot access all the prey biomass even if it can fit the prey into its mouth and the prey is not protected by a refuge.

The parameter is similar to the vulnerability parameter in EwE and is influenced by the feeding arena theory.

The $\delta_{overlap}$ is dependent on the physical overlap of prey and predator. It is determined by:

1) horizontal and vertical distribution parameters that define which cell and which distribution type (water column, sediment, epibenthic) each functional group is found in (see Distribution and Movement section below). Feeding of water column predators on sediment prey is determined by the depth consumers can dig into the sediment, whereas predators living inside

the sediment cannot feed on the water column prey

2) period of the day that a group is active (a group that is not active during that period of the day does not eat, but is available for consumption, see below),

Atlantis determines whether a prey and predator (both age structured and biomass pools) can access same habitats. If a predator **cannot access any of the habitats inhabited by the prey** (has 0 for prey habitat types), then it also cannot access any prey biomass associated with that habitat type. If a predator **can access at least one** of the habitats that the prey is associated with, then it is assumed to have **access to all of the prey biomass**.

10.2.4. Habitat refuge

The $\delta_{habitat}$ is the **optional prey habitat refuge** which is activated **only** when **all** of the points below are true:

1) The general habitat dependency flag flaghabdepend is set to 1
2) habdepend_XXX is >0 for the specific prey age-stuctured prey group
3) flag_refuge_model > 0, which will activate the standard (=1) or rugosity related (=2) refuge model
4) prey is in the bottom water column, where water is in contact with the epibenthic habitats and sediment

The habitat refuge is calculated in the *Vertebrate_Assess_Enviro()* routine in **atvertprocesses.c**

The $\delta_{habitat}$ is used to reflect the assumption that for **age-structured prey** all habitats can provide some degree of refuge even when predators can access the habitat the prey inhabits. The $\delta_{habitat}$ = 1 for prey species that are not dependent on habitat (XXX_habdepend set to 0), whereas for habitat dependent species it is defined either as a simple habitat refuge (flag_refuge_model =1) or rugosity related (flag_refuge_model=2).

A **simple habitat refuge** is calculated as:

$$\delta_{habitat} = Acov_{prey} \cdot \left(e^{(-K_{prey} \cdot Cover_{habitat} + Bcov_{prey})} + \frac{1}{K_{prey}} \right)$$

where $Cover_{habitat}$ is the weighted relative cover in the cell for the prey, $Acov_{prey}$ (Acov_ad_XXX and Acov_juv_XXX) is a scalar of the overall habitat refuge effect for prey i , $Bcov_{prey}$ (Bcov_ad_XXX and Bcov_juv_XXX) is the habitat steepness coefficient, K_{prey} (Kcov_ad_XXX and Kcov_juv_XXX) is the exponent of refuge provided by the habitat.

The availability of suitable cover is calculated as

$$Cover_{habitat} = \left(\delta_{substrate,habdegrad} \cdot p_{substrate} + p_{biogenic} \right) \cdot \left(1 + p_{canyon} \right)$$

where $\delta_{substrate,habdegrad}$ is degradation in the physical habitat due to coastal development (set by the flagdegrade flag, which also controls other habitat degradation parameters), $p_{substrate}$ is the proportion of the cell covered with suitable substrate types, $p_{biogenic}$ is the proportion of the cell covered by suitable biogenic habitats (determined by the isCover parameter in the *functional_group.csv*), p_{canyon} is the proportion of the cell covered by canyons. Note that canyons are treated differently to other habitats and, for species that live in canyons (set through the habitat_XXX vector) the canyon acts as an enhancement factor (multiplier), where 10% of canyon area in the cell will enhance the habitat cover to 1.1. This is because canyons are known to concentrate production, but their absence does not prevent the establishment and growth of the groups.

Rugosity related habitat refuge is **only** available for

1) models that include corals and
2) is only used for species that are dependent on (interact with) corals.

Further description of the rugosity related habitat cover is available here

If flag_refuge_model=2 but a species does not live on corals, then the simple refuge model described above is applied. Three extra parameters are needed for rugosity refuge model: RugCover_Coefft, RugCover_Const, RugCover_Cap.

NOTE!

Calculating the amount of habitat cover available $Cover_{habitat}$

The amount of habitat available for the species can be calculated either as a sum of total accessible habitat or as an average proportion of suitable habitats. This is controlled by flag_rel_cover parameter and will give **very different habitat cover outcomes**! For example, a cell that has three suitable habitats, covering 10%, 20% and 30% of the cell area respectively gives the following:

If flag_rel_cover is set to 0 the amount of available cover will be 0.1+0.2+0.3=0.6

If flag_rel_cover is set to 1 the amount of available cover will be (0.1+0.2+0.3)/3=0.2

The habitat refuge can be interpreted as a scalar on the available prey biomass determined by the available suitable habitat cover in the cell.

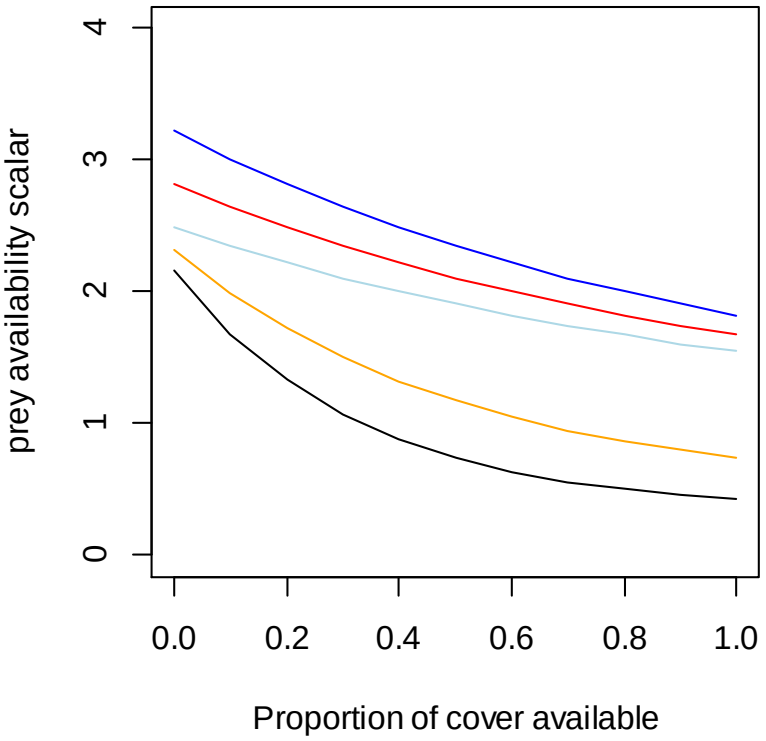


Figure 10.2. Habitat refuge scalar on prey availability as a function of cell area covered with suitable habitats (see box above for details on calculations of suitable cover area).
Black: Acov=1, Bcov=0.6, Kcov=3 (parameters similar to SETas model and originally derived from a meta-analysis of prey accessibility on coral and temperate reefs)
Effect of Kcov:
Orange: Acov=1, Bcov=0.6, Kcov=2
Red: Acov=1, Bcov=0.6, Kcov=1
Effect of Bcov:
Light blue: Acov=1, Bcov=0.4, Kcov=3
Dark blue: Acov=1, Bcov=0.8, Kcov=3

The Acov is a linear scalar that only affects the height of the curve along the y axis. If needed it could be used to scale prey availability to 1 when cover=0

NOTE!

When habitat dependency and refuge are used then the models are often parameterised so that **prey availability is upscaled (>1) when a cell contains zero available habitat** (as prey are more exposed than normal to predators). It is assumed that normally around 0.2-0.3 of the cell will have suitable habitat, which is why prey availability scalar is ~1 at this proportion of cover available (see Fig. 10.2). This is based on a (now >10 year old) meta-analysis of prey accessibility on coral and temperate reefs.

NOTE!
Limitations of current implementation of habitat refuge

1) The curve of habitat refuge does not have a final inflection that would match the case where even good quality habitat has become saturated and fails to provide any refuge for a larger proportion of the total population.
2) The curve is deterministic and time invariant so there is no explicit dynamic trade-off between predation risk and hunger-driven searching in open ground as there is in Ecosim (Walters et al 1997).
3) This approach ignores the use of cover by predators, where improved cover can actually lead to higher predation.

For **biomass pools** (CP) the $\delta_{habitat}$ refuge is available only for prey living in the **sediments** and is calculated by the depth the predators can dig into (set in KDEP_XXX) and depth of the oxygenated layer sedoxdepth, calculated and reported in the *output.nc* file, see Chapter 5.6). If sedoxdepth < KDEP

then no refuge is provided by the sediment ($\delta_{habitat}$ = 1), if sedoxdepth > KDEP then

$\delta_{habitat}$ = (sedoxdepth - KDEP)/ sedoxdepth,

For example, if the oxygenated layer depth is twice as deep as the sediment penetration depth, then only half of the sediment living biomass pool will be available (and this will be further scaled by $p_{prey,CX}$ value). If sedoxdepth=0 a small number (1e-08) is added to avoid division by zero

Table 10.2 Parameters defining habitat associations and habitat refuge.

Parameter	Description
flaghabdepend	Flag turning on habitat dependency routines. If it is set to 0, all habitat associations and routines are ignored.
XXX_habdepend	Flag identifying which age structured groups are habitat dependent
habitat_XXX	A vector of values identifying which of the available habitat preferences for biomass pool group XXX. Traditionally these used to be only 0 or 1 values, indicating absence or presence in each habitat. Now more resolved setting is allowed. A value > 0 indicated that the organism prefers the habitat and of < 0 it avoids it. The absolute value indicates the relative weighting, where proportion of suitable habitat is calculated as a multiplier of habitat proportion and habitat preference given in habitat_XXX vector.
juv_habitat_XXX ad_habitat_XXX	Same as above but applied for age structured groups and setting available habitats for juvenile and adult life history stages separately
flag_refuge_model	If set to 0, no habitat refuge for age structured groups will be applied. If set to 1 the standard refuge is calculated as described above. If set to 2 the habitat refuge is rugosity related
flag_rel_cover	A flag indicating whether to use cumulative habitat (0) or average relative cover (1) when interacting with predators
Kcov_juv_XXX Kcov_ad_XXX	exponent of refuge provided by the habitat for age structured groups (provided for juveniles and adults)
Acov_juv_XXX Acov_ad_XXX	scalar of the overall habitat refuge for age structured groups (provided for juveniles and adults)
Bcov_juv_XXX Bcov_ad_XXX	habitat steepness coefficient for age structured groups (provided for juveniles and adults)
KDEP_XXX	depth consumers can dig into the sediment, in meters. Used to determine habitat refuge for prey digging in the sediments
isCover in .csv	Which epibenthic groups also serve as a habitat

10.2.5. Size refuge from predation

The **non-optional size refuge of age-structured prey** δ_{size} is defined by gape limitation, which is used to simulated size selectivity and physical limitation of predators when feeding (i.e. what can fit in their mouth). The gape limitation can be calculated in two ways – ***hard feeding window*** or ***smooth feeding curve*** (defined by the UseHardFeedingWindow flag)

The hard feeding window defines knife-edge size selectivity, where availability of the prey is either zero (prey not available to the predator) or one (prey available). The prey is available to the predator if its size falls within the lower and upper prey selection size limits (KLP_XXX and KUP_XXX), or gape size, of the predator. In these calculations **size is defined by structural nitrogen (SN) only**!

$$\delta_{size} = \begin{cases} = 1, & if\ KLP \cdot SN < SN_{prey} < KUP \cdot SN \\ = 0, & otherwise \end{cases}$$

In many models the lower and upper limits are 0.1% and ca 40% of the predator size respectively; so the predator of 1000mg SN could consume age structured prey with from 1 to 400mg (KUP ranges from 15 to 80%).

The smooth feeding window models a smoother transition of prey availability and is set by UseHardFeedingWindow=0. The original implementation of the smooth feeding window used a humped window calculation; now defined with an additional flag UseHumpedFeedingWindow=1. If humped window is used the availability of prey on the upper predator's size limit is calculated as

$\delta_{size} = relSize \cdot e^{(Kmax \cdot (1 - relSize))}$, where

$$relSize = \frac{SN_{prey}}{SN \cdot KUP}$$

represents **relative size** of the prey compared to the predator's upper size limit (*KUP*), and *Kmax* (Kmax_coefft_XXX) is the steepness of the smooth selectivity curve.

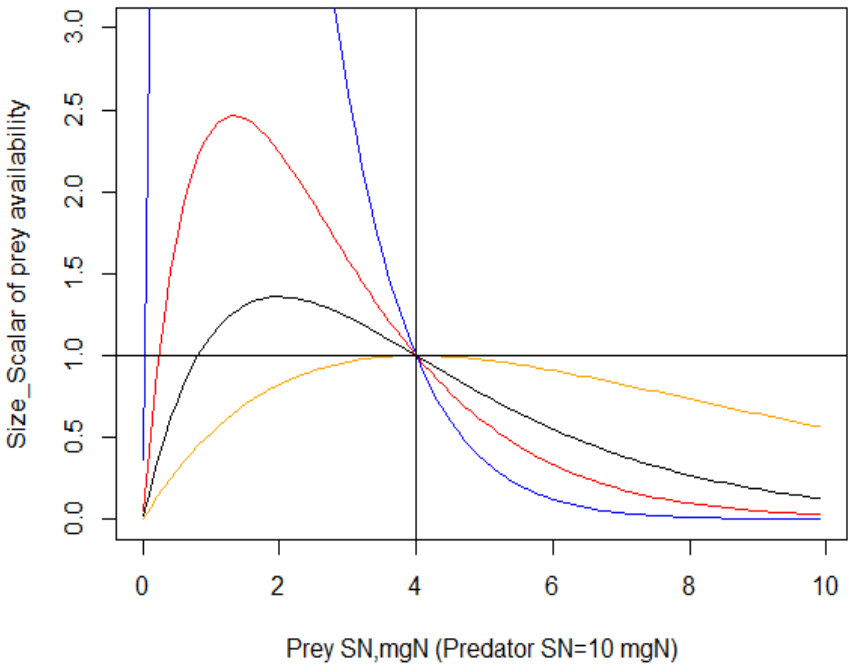


Figure 10.3. Prey availability scalar Size_Scalar$_{prey}$ of different sized prey for different Kmax_coefft values for humped selectivity curve.

Black: *Kmax* = 2
Orange: *Kmax* = 1
Red: *Kmax* = 3
Blue: *Kmax* = 5
Predators SN=10mg and KUP=0.4 (shown by vertical blue line). The horizontal black line shows that the maximum value of δ_{size} is set to 1 (values larger than one are capped at 1)

As Fig.10.3 shows, the humped window calculation is very sensitive to the Kmax_coefft_XXX parameter, where availability of prey at lower size limit can increase very rapidly. An alternative

formulation is now available as a logistic shaped curve to calculate δ_{size}. This option is appplied if UseHardFeedingWindow = 0 and UseBiLogisticFeedingWindow= 1 **and it is based on total length rather than SN of predator and prey**

$$\delta_{size} = \begin{cases} = \dfrac{1}{1 + e^{\left(-Kmax \cdot \left(Len_{prey} - Len_{prey,50\%KLP}\right)\right)}}, & if\ Len_{prey} < Avail_{top} \cdot Len_{pred} \\ = \dfrac{1}{1 + e^{\left(Kmax \cdot \left(Len_{prey} - Len_{prey,50\%KUP}\right)\right)}}, & if\ Len_{prey} > Avail_{top} \cdot Len_{pred} \end{cases}$$

where *Kmax* is the steepness of the logistic selectivity curve (Kmax_coefft_XXX), $Len_{prey,50\%KLP}$ and $Len_{prey,50\%KUP}$ are midpoints of the logistic selectivity curve or length of prey at which availability is half of the maximum possible value. The $Avail_{top}$ is the length of prey for which availability is highest.

The $Avail_{top}$ is calculated as a midpoint between the KLP and KUP values, hence assuming a symmetrical selectivity curve. For example, if the KLP=0.2 and KUP=0.6, the $Avail_{top}$ = 0.4. Similarly the $Len_{prey,50\%KLP}$ is the midpoint between the lower (KLP_XXX) predator size limits and the $Avail_{top}$, and $Len_{prey,50\%KUP}$ is the midpoint between the upper (KUP_XXX) predator size limits and the $Avail_{top}$. In the example above the $Len_{prey,50\%KLP}$ = 0.3 and $Len_{prey,50\%KUP}$ = 0.5.

Note, that if the steepness of the curve (*Kmax* value) is low, the the δ_{size} in the equation above may never reach 1 (see Figure 10.4). It is NOT recommended to use Kmax values lower than 0.5 because it makes the feeding window very wide and allows feeding on almost all sizes of prey (orange line in Fig. 10.4).

The bilogistic feeding window uses length rather than SN of prey and predator, because large differences between SN (many orders of magnitude) makes the size scalar highly sensitive to the Kmax parameter.

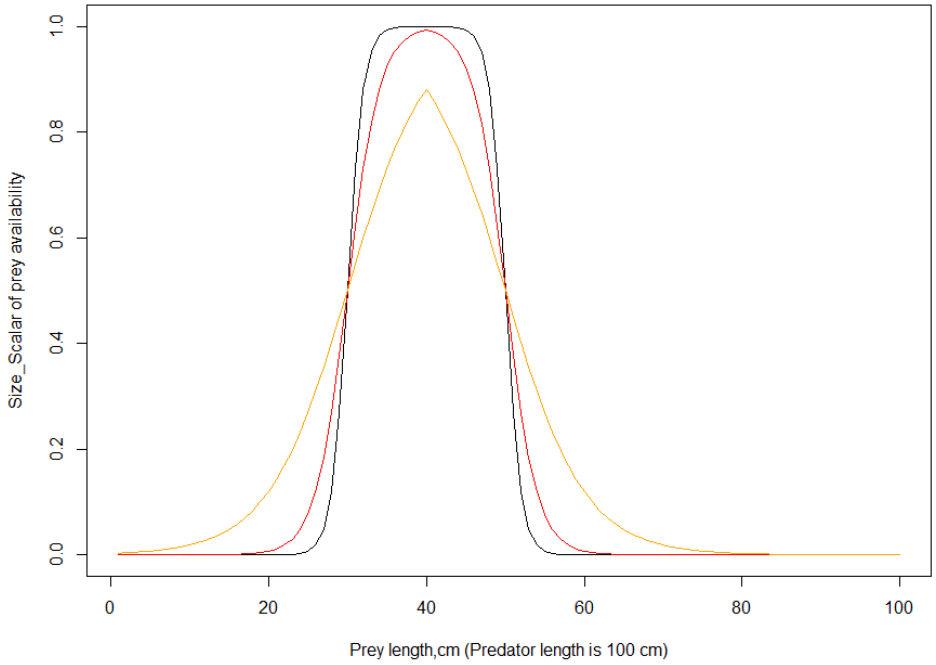


Figure 10.4. Prey availability scalar δ_{size} of different sized prey for bilogistic selectivity curve.

Black: Kmax = 1, KLP=0.2, KUP=0.6
Red: Kmax=0.5 , KLP=0.2, KUP=0.6
Orange: Kmax = 0.2, KLP=0.2, KUP=0.6

NOTE!

Calculating length in age-structured groups and biomass pools

If length-based selectivity is selected for a fishery, it will be applied to both age-structured groups and biomass pools. The application of length-based selectivity to biomass pools is a new feature in Atlantis and it is important to get it right, or else invertebrates may not be caught by the gear (see wiki post here)

Length of an age group in age-structured groups is calculated using length-weight conversion parameters (li_a_XXX and li_b_XXX) given in the *biology.prm* file. For this, first, the structural and reserve nitrogen (RN+SN, in mg) are added up and converted to wet weight in grams (*wgt*) as:

$$wgt = (RN + SN) \cdot wetdry \cdot X\_CN \cdot 1000$$

where *wetdry* is the wet weight to ash free dry weight ratio (k_wetdry in *biology.prm*) typically set to 20 in many Atlantis parameter files, based on using C as a proxy for dry weight (but noting that in many models and literature the wet : dry ratio used is often closer to 6-8) and *X_CN* is the carbon to nitrogen Redfield ratio (X_CN in *biology.prm*) typically set to 5.7.

The *wgt* (in grams) is then converted to length (in cm) using the standard equation

$$wgt = a \cdot length^b \qquad \text{or} \qquad length = \left(\frac{wgt}{a}\right)^{1/b}$$

where *a* is li_a_XXX and *b* is li_b_XXX in the *biology.prm*. Note that these parameters could be set in such a way to return length in metres instead, but by convention it is in cm (as are any other faunal lengths).

Since for **biomass pools** only one nitrogen pool is tracked (individuals are not tracked separately), the structural nitrogen of an average individual must be given in the *biology.prm* file. This is given in the XXX_sn parameter for each consumer biomass pool group (not plants, bacteria or detrital pools). In addition to the length-based selectivity, this SN value is used in length-based feeding interactions when biomass pool predators feed on the age-structured prey (see chapter 10.2.5).

The length of an average biomass pool invertebrate is calculated in the same way as for age structured groups above. However, before the calculation the SN is converted to the SN+RN weight using the X_RS ratio parameter in the *biology.prm* file, so that

$$wgt = (1.0 + X\_RS) \cdot SN$$

hence assuming that the ratio of hypothetical RN and SN in an invertebrate is optimal. **The length-weight conversion parameters li_a_invert and li_b_invert given in the *biology.prm* are identical for all biomass pool species.**

NOTE!

Smooth and hard feeding window

Applying hard feeding window (knife-edge selectivity) means that very small changes in prey's size can abruptly change their availability from 0 to 1. Although this approach is used in a range of ecological models it can lead to abrupt changes in growth or predation mortality between cohorts.

If smoother transition on size selectivity is desired, a smooth feeding window should be used. Note, however, that the "humped" implementation of the smooth feeding curve extends the availability of prey above the upper gape limit (KUP_XXX) of the predator (Fig. 10.3). The curve is also very sensitive to the Kmax_coefft parameter. Applying this smooth feeding curve might require decreasing the KUP_XXX values.

When using bilogistic curve the prey availability is generally decreased compared to the hard feeding window – the availability is equal to that of hard feeding window only for a small size range of the prey (Fig. 10.4). This means that when changes from hard to bilogistic curve are applied, the overall prey availability may have to be increased to ensure enough food for the predator.

Table 10.3 Parameters defining size refuge for age structured groups

Parameter	Description
KLP_XXX	Min gape limit of the predator (age structured or biomass pool)
KUP_XXX	Max gape limit of the predator (age structured or biomass pool)
UseHardFeedingWindow	Flag setting up hard (0) or smooth (1) feeding window
UseBiLogisticFeedingWindow	Flag indicating whether using humped (0) or bi-logistic (1) smooth feeding window
Kmax_coefft_XXX	Steepness of the smooth feeding window curve of the predator
XXX_sn	Typical size (amount of SN) of a biomass pool consumer to determine the size of age structured prey available to it

10.2.6. Availability of fisheries catches to opportunistic catch-eaters

The availability of fisheries catches is an optional set of parameters that allow groups to access additional prey. Proportion of catch available from each fishery (set in PropCatch_XXX parameter) is added to the total available prey. No gape limitation (size refuge) is applied to fisheries catches.

Table 10.4. Parameters defining feeding on fisheries catches

Parameter	Description
isCatchGrazer in .csv file	Identifies species that can feed on fisheries catches
pFCXXX	A vector of values (0 and 1) identifying which of the fisheries catches are available to each catch eating species identified as isCatchGrazer. The vector should be as long as there are fisheries in the *fisheries.csv* file (assumes the same order as given in the *fisheries.csv* file).
PropCatch_XXX	Proportion of the catch in each fishery that can be exploited by the isCatchGrazer groups

10.2.7. Effect of temperature and salinity on feeding parameters

Feeding rates and assimilation efficiencies will be temperature dependent if a species is temperature sensitive. Salinity dependency is optional. See chapter13 for further details.

10.2.8. Effect of oxygen limitation on epibenthic invertebrate feeding rates

Oxygen limitation affects the feeding rates of SED_EP_FF, MOB_EP_OTHER, SED_EP_OTHER, SM_INF, LG_INF and CORAL functional groups (applied in group routines *Epibenthic_Ivert_Process()*, *Invert_Consumer_Process()* and *Coral_Process()*). This dependency is also an option for age structured groups in newer code releases that are currently in the pipeline.

The oxygen limitation scalar δ_{O2} is applied to clearance rate *C*. Atlantis has four options to calculate oxygen limitation scalar, set by O2case parameter. The oxygen limitation scalar is calculated by *Oxygen()* function in **atprocess.c**

1. Ambient oxygen limitation (O2case=0) uses ambient oxygen levels O_{amb}, lethal oxygen concentration (KO2_XXX, mgO2 m^{-3}) and limiting oxygen concentration (KO2lim_XXX, mgO2 m^{-3}) parameters. Remember that oxygen solubility in seawater at 5C and 1bar pressure is 10 mg/l or 10000 mg m^{-3}.

$$\delta_{O2} = \begin{cases} = \dfrac{(O_{amb} - KO2)^2}{(O_{amb} - KO2)^2 + (KO2LIM - KO2)^2} & , \text{if } O_{amb} > KO2 \text{ and } KO2LIM > KO2 \\ = 1 \quad , & \text{if } O_{amb} > KO2 \text{ and } KO2LIM < KO2 \\ = 0 \quad , & \text{if } O_{amb} < KO2 \end{cases}$$

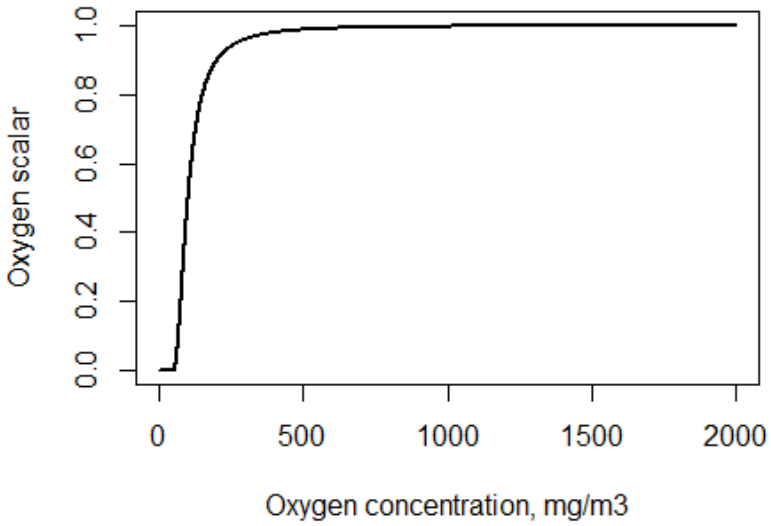


Figure 10.5. Oxygen limitation scalar δ_{02} for different ambient oxygen concentrations, assuming KO2=50 mgO2 m^{-3} and KO2lim = 100 mgO2 m^{-3}.

2. Depth based limitation (O2case=1) only uses sediment depth of half oxygen mortality mD (mD_XXX, m) parameter and depth of the oxygenated sediment layer sedoxdepth (see chapter 5.6) where

δ_{O2}=sedoxdepth/(sedoxdepth+mD)

This limitation only reflects reduced feeding within the sediment due to low oxygen conditions, so it is mostly relevant to epibenthic invertebrate biomass pools that can dig into sediment for food or infauna.

3. IGBEM based limitation (O2case=2) is modified after the ERSEM model and is calculated as:

$\delta_{O2}=O_{amb}/(KO2+O_{amb})$, where O_{amb} is the concentration of oxygen in the cell

4. Quadratic limitation (O2case=3) calculates both ambient and depth based limitation (points 1 and 2 above) and uses the larger of the two values as the oxygen scalar δ_{O2}

10.2.9. Effect of space limitation on epibenthic invertebrate feeding rates

Space limitation can affect *C* or *mum* rates of benthic invertebrate biomass pools if:

1) A biomass pool group XXX is a filter feeder or mobile epibenthic group (SED_EP_FF and MOB_EP_OTHER GroupType) **and** is habitat dependent (habdependXXX=1) **and** space limitation is active (flagXXXlim=1). If these three conditions are satisfied then the Crowding Effect scalar is applied to the clearance rate *C* of XXX (see chapter 10.5.3).

2) A biomass pool group is habitat dependent (with at least some habitat preference parameters not set to 1), benthic and flag_benthos_sediment_link=1. See chapter 10.5.3 for further details.

10.2.10. Other factors affecting feeding parameters

The feeding interactions described above are further dynamically modified by a range of ecological and environmental factors, such as species activity, starvation, or pH.

Species activity

Atlantis has an option to set whether a species is active during the day or night, or all the time. This is done using flagXXXday parameter. It is important to note that if a diurnal timestep is used (i.e. 12h or less) and if a species activity preference is set to day or night, it will not be active in the model during the other half of the day – it will not initiate ecological processes such as eating, but can still be preyed upon.

NOTE!

If a species is only active during the day or night (flagXXXday set to 0 or 1) it means that **no ecological routines will be initiated by this species during the period of each day that it is inactive.**

This means that an inactive species **will not feed, move, reproduce and will not breathe**, if respiration is included in the model. This also means that even if species physically overlap and have diet connections, but a predator is not active at that time of the day, no feeding interaction will occur.
An inactive group is succeptible to fishing and predation.

Spawning

Species that are in the spawning period will not feed if the parameter feed_while_spawn is set to 0. The biomass of the spawning stock will subtracted when calculating the feeding biomass of a predator.

Starvation

Starvation has no effect on *C* and *mum* in the modified Holling functional responses (predcase 1-3) and ratio dependent feeding responses (predcase 4).

For other feeding responses handling time (ht_XXX in biomass pools and hta_XXX and htb_XXX in age-structured groups) is scaled by KHTD or KHTI scalars when the condition of an age structured group

falls below a limit set in Kthresh1 and Kthresh2 (see box below). See chapter 10.5.2 for further details on how starvation is determined and what it affects.

Acidification

Atlantis includes explicit representation of the possible effects of acidification. The model is activated with the flagmodelPH=1 in the *biology.prm* file. It allows the user to scale prey availability, assimilation efficiency, *C* and *mum* rates, mortality, recruitment and other processes. See chapter 13 for further details.

10.2.11 Other feeding functional responses

Twelve different feeding functional responses are currently implemented in Atlantis. They are selected using the predcase_XXX parameter, which can take values from 0 to 12 (option 3, Ecosim based feeding rate, is currently unavailable). The first three cases are for the modified Holling Type I-III responses adopted from the PPBIM. They use the *mum* parameter, as described above. Two responses – minimum-maximum and Holling size dependent type III - are adopted from the ERSEM model (see Table 10.5), but have not been used in many models. The remaining responses can be grouped into standard Holling responses (Holling type I to IV) and ratio depdenent responses (see e.g. Arditi and Ginzburg 1989 and critique of the approach by Abrams 1994). The ratio dependent responses take into account not only the prey biomass but also the ratio of prey to predator biomass. These responses don't use the *mum* parameter but a mass specific handling time (ht_XXX) parameter.

NOTE!

Search volume (vl) and handling time (ht) as alternative parameters to C and mum

The modified Holling type functional responses use age-specific *C* and *mum* parameters. For other functional responses either one or both of these parameters are replaced with search volume (vl_XXX, vla_XXX and vlb_XXX) and handling time (ht_XXX, hta_XXX and htb_XXX).

Unlike *C* and *mum*, however, the search volume and handling time parameters are always set **based on mass specific structural ash-free dry weight** (not nitrogen!). This means that only one value is provided in the parameter file, and the age specific values for age-structured groups are calculated based on the individual's body weight. For biomass pools the units of vl_XXX are in $m^3\ mgN^{-1}\ day^{-1}$. For age structured groups vla_XXX and vlb_XXX determine an allometric search volume-mass relationship, and they are converted to individual nitrogen-based values in *Determine_Fish_Feeding_Prms()* in **atvertprocesses.c** where

$$VL = vla \cdot (SN \cdot XCN)^{vlb}$$

where SN is the structural nitrogen of an age class and XCN is the C:N Redfield ratio (X_CN in the biology.prm) typically set to 5.7 used to convert nitrogen mass to total ash-free dry mass.

For demersal species (flagdem=1) the search volume VL is halved (VL=VL/2).

The handling time HT is calculated as

$$HT = \delta_{starve} \cdot hta \cdot \left(SN \cdot XCN\right)^{-htb}$$

where δ_{starve} is a scalar that reduces the handling time when an individual is starving (see chapter 10.5.2 on the effects of starvation).

Note, that only SN and not total nitrogen values are used for search volume and handling time calculations.

The table below gives equations for the 12 functional feeding response options. In all the equations B denotes the predator's biomass, B^*_{prey} is the available biomass of a given prey, scaled by the optional habitat refuge, and obligatory size and availability parameters, as explained above (note that prey list also includes the available fisheries catch, if the predator is identified as catch-eater in the *functional_group.csv* file)

$$B^*_{prey} = p_{prey,CX} \cdot \delta_{overlap} \cdot \delta_{habitat} \cdot \delta_{size} \cdot B_{prey}$$

and $\sum B_{prey,i}$ is the sum of all available prey.

Table 10.5. Equations defining grazing term in the twelve currently available feeding functional responses. Some of these responses are described and compared in Fulton et al. 2003a "Mortality and predation in ecosystem models: is it important how these are expressed?" *Ecological Modelling* 169,157–178

Pred case	Functional response	Equations
0	Modified Holling type II, adopted from PPBIM	$$Gr_{prey} = \frac{B \cdot C \cdot B^*_{prey}}{1 + \frac{C \cdot \sum_i \left(E_i \cdot B^*_{prey,i}\right)}{mum}}$$
1	Modified Holling type I, adopted from PPBIM This response has linear increase with prey density, but is capped at	if $(C \cdot B^*_{prey}) > (mum / E_{preyLIVE})$

	the maximum determined by maximum growth rate (*mum*) divided by the assimilation efficiency on live prey (which is typically highest)	$$Gr_{prey} = \frac{B \cdot mum \cdot B^{*}_{prey}}{E_{\text{prey LIVE}} \cdot \sum_i B^{*}_{prey,i}}$$ else $$Gr_{prey} = B \cdot C \cdot B^{*}_{prey}$$
2	Modified Holling type III, adopted from PPBIM Same as Holling type II, but prey biomasses are squared	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*2}_{prey}}{1 + \frac{C \cdot \sum_i \left(E_i \cdot B^{*2}_{prey,i}\right)}{mum}}$$
4	Minimum-maximum Adopted from ERSEM, where it was used to describe fish feeding. However, it is not used in ERSEM anymore, because higher trophic level predators are not currently included in ERSEM	$$Gr_{prey} = \frac{B \cdot C \cdot mum \cdot \frac{B^{*2}_{prey}}{L + B^{*}_{prey}}}{U + \sum_i \frac{B^{*2}_{prey,i}}{L + B^{*}_{prey}}}$$ *L* = lower prey biomass threshold for feeding by predator XX (KL_XX) *U*= half saturation coefficient for feeding by predator XX(KU_XX)
5	Holling type III – size dependent Adopted from ERSEM, where it was used to describe seabird and mammal feeding (it is not used in ERSEM anymore, see above)	$$Gr_{prey} = \frac{B \cdot VL \cdot B^{*}_{prey} \cdot \sum_i B_{prey,i}}{1 + VL \cdot HT \cdot \sum_i B_{prey,i}}$$ VL and HT are search volume and handling time (see above). $\sum B_{prey.i}$ is the sum of all available prey. Remember that for demersal species (flagdem=1), the search volume is halved
6	Ratio dependent See Arditi and Ginzburg 1989 and text above. This approach accounts for competition among predators	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*}_{prey}}{1 + C \cdot HT \cdot \sum_i B_{prey,i} + Compet}$$

	through ratio of predator and prey biomasses	*Compet* = interpredator competition. Please check the wiki for further details as this part of the code is changing based on discussion with experts in the field.
7	Standard Holling Type I These are the standard Holling type responses. They have been added to Atlantis in 2015	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*}_{prey}}{\sum_{i} B_{prey,i}}$$
8	Standard Holling Type II These are the standard Holling type responses. They have been added to Atlantis in 2015	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*}_{prey}}{1 + C \cdot HT \cdot \sum_{i} B_{prey,i}}$$
9	Standard Holling Type III These are the standard Holling type responses. They have been added to Atlantis in 2015	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*2}_{prey}}{1 + C \cdot HT \cdot \sum_{i} B^{2}_{prey,i}}$$ As in Holling Type II but the prey and total prey biomasses are squared
10	Standard Holling Type IV These are the standard Holling type responses. They have been added to Atlantis in 2015	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*}_{prey}}{1 + C \cdot HT \cdot \sum_{i} B^{2}_{prey,i}}$$ As in Holling Type II but the total prey biomasses are squared
11	Hassel Varley This response allows for interference among predators	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*}_{prey}}{1 + C \cdot HT \cdot \sum_{i} B_{prey,i} + TP^{HVM}}$$ *TP* = total predator biomass or abundance *HVM* = is the coefficient of mutual interference for the top carnivores
12	Crowley Martin Like Standard Holling type II but includes competition among predators as in option 6	$$Gr_{prey} = \frac{B \cdot C \cdot B^{*}_{prey}}{1 + C \cdot HT \cdot \sum_{i} B_{prey,i} \cdot (1.0 + Compet)}$$

		Compet = interpredator competition. Please check the wiki for further details as this part of the code is changing based on discussion with experts in the field.

Earlier when type I response was used it was important to cap the clearance rate of the predator, to ensure it does not get unrealistically high. This was done by setting flag_satiation=1. This approach caps the clearance rate at the value of $mum/E_{typeLIVE}$, where $E_{typeLIVE}$ is the assimilation efficiency on live prey, which is typically the highest assimilation efficiency.

$$Gr_{prey} = \frac{B \cdot mum \cdot B^*_{prey}}{E_{typeLIVE}}, if \quad clearance > \frac{B \cdot mum}{E_{typeLIVE}}$$

The flag_satiation parameter is a special case and in most cases will not be required.

10.3. Food assimilation in consumers

The total consumed food $\sum Gr_{prey}$ is assimilated according to the consumer assimilation efficiencies of four different types of prey i, defined as animals (including fisheries catch), plant, labile detritus and refractory detritus. The assimilation efficiency can be further externally scaled by a scalar *gs* supplied through forcing files and used to modify growth due to factors not included in the model.

In consumer biomass pools (CP) all assimilated food is directly allocated to growth.

$$G_{CP} = A_{CP} = \sum_i (E_i \cdot Gr_{prey}) \cdot gs$$

In age-structured consumers (CX) the assimilated food is allocated to a temporary pool *A* (called **growth* in Atlantis code). The energy in the pool A is used to meet optional respiration costs and the remainder then allocated to growth of SN and RN pools. Hence the *A* pool is not tracked but only used for computational purposes.

$$A_{CX} = \sum_i (E_i \cdot Gr_{prey}) \cdot gs$$

Assimilation rate in age structured groups can be affected by temperature, salinity and pH (see chapter 13).

NOTE!

Accounting for maintenance costs through assimilation efficiency

The assimilation parameter is required for both biomass pools and age-structured groups. However,

depending on whether respiration costs are explicitly included in age-structured groups, the assimilation efficiency coefficient can take a different meaning.

In Atlantis explicit mass-specific respiration is only available in age-structured groups, where size of an average individual in an age group is tracked. For biomass pools maintenance costs are assumed to be represented implicitly in assimilation (in)efficiency. Such representation assumes that temperature responses of assimilation efficiency and maintenance processes are the same. It also assumes that when assimilation (*Acx*) is zero (no food is available) then also respiration or maintenance is zero.

This has important implications for obtaining assimilation efficiency values and comparing them to other models that include maintenance explicitly. For example, in ERSEM model respiration includes two components, basal respiration that is dependent on temperature and independent of the food intake, and activity respiration that represents a fraction of the assimilated food. In Atlantis basal respiration is optional and is only included for age-structured groups. The activity respiration is not explicitly modelled but is implicitly included in the assimilation efficiency.

If respiration of age structured groups is explicitly included in the model (see below), the assimilation efficiency of age structured groups and biomass are not directly comparable and the values for biomass pools should be much lower than that for age-structured groups.

10.4. Maintenance (respiration) in consumers

In Atlantis the only explicit maintenance costs are those of respiration. Respiration is only included in fully age structured groups and is optional. The majority of existing Atlantis models do not explicitly model maintenance or respiration. The respiration or maintenance costs are assumed to be implicitly accounted for by assimilation inefficiencies (see NOTE! in chapter 10.3).

Age-structured group respiration is carried out by the *Fish_Respiration()* routine in **atvertprocesses.c**

Explicit respiration in the age-structured groups represents base respiration costs and depends on body mass, temperature and condition. It is activated by setting flagresp to 1. The nitrogen used up in respiration is taken out of the temporary *A* pool (before that pool is allocated to growth) and added to the NH pool.

The nitrogen consumed for respiration is calculated as

$$Rs = e^{(Ktmp \cdot T_{H2O})} \cdot KA \cdot Wgt^{KB}$$

where *Ktmp* (Ktmp_fish, Ktmp_shark, etc), *KA* (KA_XXX) and *KB* (KB_XXX) are parameters defined for each age structured group or larger functional groups (fish, sharks, mammals, birds) and *Wgt* is individual dry weight calculated as

$$Wgt = (SN+RN) \cdot C\_XN$$

The C_XN parameter represents N:C Redfield ratio, typically set to 5.7. This means that ***KA* and *KB* parameters relate to total dry weight** of an individual and **not to the nitrogen** content tracked in the model.

NOTE!

Should models include explicit respiration?

Of course, all organisms have maintenance costs and they have to breathe. The question is whether including these costs implicitly in assimilation efficiencies provides sufficient representation of the physiological-ecological dynamics.

The key difference in using explicit respiration is that it depends on body mass, temperature and condition and is independent of the food intake rate. It is therefore more accurately reflects scaling of respiration to body size and to temperature and allows for different temperature scaling of assimilation and respiration rates.

If a large proportion of the biomass of the total model is in age-structured groups with strong variation in size or environmental conditions, then explicit use of respiration (flagresp=1) is a sensible option to take. However, if there is little variation in the environment, size at age or the model is dominated by biomass pool groups then setting flagres =0 may provide sufficient representation of the system. Also, when the focus of a model is to explore temperature sensitivities or growth and size dynamics of age-structured groups, then including explicit respiration may be advisable.

One of the reasons that Atlantis was developed in place of the earlier IGBEM and ERSEM models is because the explicit physiological detail included in those models was unwieldy to use and calibrate in each new location (imagine that a model harder than Atlantis to use!!). Therefore a simpler representation that produce approximately the same dynamics was developed and implemented in Atlantis (see for example Fulton et al. 2004c). Explicit maintenance costs are excluded from many models, and their best representation remains a topic of debate (e.g. Persson et al. 2014).

10.4.1 Scaling of respiration costs depending on individual's condition

Respiration costs are scaled by a factor KST in starving individuals (see chapter 10.5.2. for further details on starvation).

The optimum RN to SN ratio is defined by a general X_RS parameter, which is typically set to 2.65. If the RN/SN is below the optimum, the respiration costs are scaled by a factor KST. The Kthresh2 parameter controls how far below the optimum RN/SN the individual can be before the scaling of respiration costs starts.

$$if\ \frac{RN}{X_{RS}} \cdot SN < Kthresh2,$$
$$Rs = Rs \cdot KST$$

Table 10.6. Parameters defining respiration rates in age structured groups

Parameter	Description
flagresp	Flag turning-on explicit respiration in age structured groups
KA_XXX	Scaling coefficient of allometric respiration- weight rate
KB_xxx	Exponent of allometric respiration-weight rate
Ktmp_fish, Ktmp_shark, Ktmp_bird, Ktmp_mammal	Parameter defining how respiration increases with temperature in the four group types FISH (and FISH_INVERT), SHARK, BIRD and MAMMAL
Kthresh2	Threshold relative reserve below which respiration rate is scaled by KST
KST_fish, KST _shark, KST _bird, KST _mammal	Scalar of respiration when relative reserve drops below Kthresh2 for the four group types FISH (and FISH_INVERT), SHARK, BIRD and MAMMAL

10.5. Growth

10.5.1. Growth and energy allocation in age-structured groups

Once the optional respiration costs (*Rs*) are subtracted from the temporary pool *A*, **ALL remaining** energy is allocated to RN and SN. The energy is partitioned between RN and SN depending on:

1) the consumer's condition, defined by the current ratio of the RN and SN pool and the X_RS parameter that sets the optimal ratio of RN to SN in well fed age structured groups
2) the consumer's preference for rebuilding of reserves over structure, defined by the pR_XXX parameter (where higher values show lower preference for allocating to RN).

The proportion of energy in consumer CX going to RN and SN is allocated as:

$$G_{RN} = (A - Rs) \cdot (1 - \lambda)$$

$$G_{SN} = (A - Rs) \cdot \lambda$$

where

$$\lambda = \begin{cases} \dfrac{\frac{1}{X\_RS} + pR \cdot \left(\frac{RN}{pR \cdot SN} - 1\right)}{\frac{1}{X\_RS} + \frac{RN}{pR \cdot SN}}, if\ (A - Rs) > 0 \\ 0, \qquad otherwise \end{cases}$$

The equation above shows that if respiration costs are higher than assimilated energy at a given time step (Rs>A), then λ=0 and no energy will be allocated to or taken out of SN. Moreover, (A-Rs) will be negative and the difference between A and Rs is taken out of RN.

Traditionally in Atlantis negative (A-Rs) was not taken out of the RN and in fact most models do not include maintenance, so (A-Rs) never gets negative. To enable negative (A-Rs) to be taken out from RN pool, set the flag_shrinkfat=1. This is **strongly recommended for model that use respiration**. If flag_shrinkfat=0 (for legacy issues) the negative (A-Rs) is converted to zero and nothing is taken out from either RN or SN. This means that the only way RN pool can decrease is through spawning, as it is not used to cover maintenance costs during starvation.

Figure 10.6 shows the value of λ (y axis) depending on the RN/SN. Until the RN/SN is close to the X_RS value, all available energy will go to rebuilding reserves (λ=0). Black and blue lines show energy allocation under two different X_RS parameters. Once RN/SN > X_RS, the condition of the animal is considered to be good and some energy is allocated to growth of SN. Once RN is replenished, all available energy will be directed to SN. The steepness of the curve and value of RN/SN above which all energy will be sent to SN depends on the pR_XXX parameter, where **larger pR values lead to steeper curve and earlier allocation of all energy to SN**.

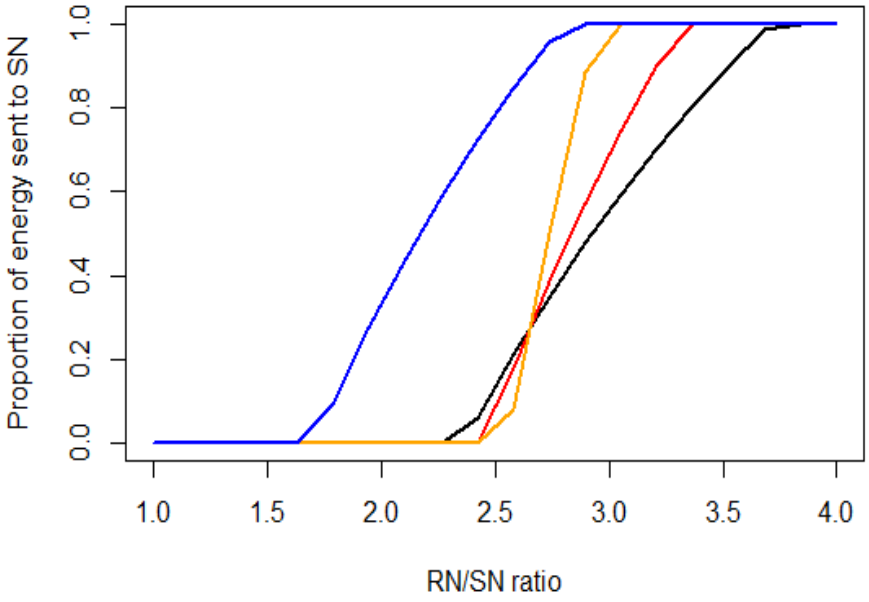


Figure 10.6. Proportion of available energy allocated to SN, depending on the individual's condition (RN/SN, on x axis), and pR and X_RS parameters.

Black: X_RS=2.65, pR=3.5
Red: X_RS=2.65, pR=5
Orange: X_RS=2.65, pR=10
Blue: X_RS=2, pR=3.5

10.5.2. When is an age-structured group starving and what does it mean?

Condition of an age-structured group is determined by the optimal RN to SN ratio, defined by the X_RS parameter, which is typically set to 2.65. When RN < X_RS*SN the age structured group is considered to be starving (or at least food deprived).

This interpretation is different from some other models (e.g. Dynamic Energy Budget approach), where even a small amount of reserve available means that an individual has reserves and is in good condition. In Atlantis RN is not treated as pure reserve, but also includes gonad weight, fat and other body parts that can be reabsorbed during starvation.

NOTE!

If respiration is not included, the only way RN pool can decrease is through spawning. This means that a group will always have lower than optimum RN/SN ratio after a spawning event, until RN pool is replenished (even when abundant food is available). If no food is available, the RN/SN ratio will not decrease further, because no maintenance costs are required.

If respiration is included and flag_shrinkfat = 1 then RN pool will decrease at low food abundance and an age group will be "starving" as it is usually understood.

There are different levels of starvation and they have increasing effects on ecological processes:
*1) When RN/X_RS*SN < 1*
- the amount of spawn produced by an individual is reduced (see Spawn in Chapter 10.8).

*2) When RN / X_RS*SN < Kthresh1*
- handling time is scaled by the KHTD parameter value. This is only applied if predcaseXXX > 4

*3) When RN / X_RS*SN < Kthresh2*
- handling time is scaled by the KHTI parameter value. This is only applied if predcaseXXX > 4
- respiration rate is scaled by the KST parameter. This is only used if flagresp = 1 (respiration used)

Additionally, starvation will lead to starvation mortality, as specified by the starvation mortality parameter, see Chapter 10.6.2

10.5.3 Growth in consumer biomass pools

No explicit respiration costs are included in biomass pools, hence all assimilated energy A_{CP} is allocated to growth. However, in epibenthic organisms space limitation scalar δ_{space} is applied to limit growth in crowded conditions

$$G_{CP} = A_{CP} \cdot \delta_{space}$$

The δ_{space} scalar is calculated differently in different invertebrate GroupTypes, and is determined by habitat dependency and/or the flagXXXlim parameter.

NOTE!

Different meaning of flagXXXlim and space limitation in different invertebrate groups

Space limitation is controlled by flagXXXlim and is applied differently for different types of invertebrates:

For filter feeding and mobile epibenthic invertebrates (**SED_EP_FF** and **MOB_EP_OTHER**) flagXXXlim space limitation will be either simple (flagXXXlim=1) or ERSEM based (flagXXXlim=2) – called through *Epibenthic_Invert_Processes()*routine

If flagXXXlim=0 there is no space limitation
If flagXXXlim=1 space limitation is calculated as δ_{space} scalar and applied **to the biomass**.
If flagXXXlim=2 space limitation is calculated as Crowding effect and applied to **clearance rate** (C)

For **infaunal** and other **sediment invertebrates** (**SM_INF, LG_ING, SED_EP_OTHER and CORAL**) flagXXXlim means either no limitation (flagXXXlim=0) or simple limitation (flagXXXlim=1) – called through *Invertebrate_Consumer_Process()*, *Sediment_Epi_Other_Process()* or *Coral_Process()*routines

If flagXXXlim=0 no space limitation is applied
If flagXXXlim=1 space limitation is calculated as δ_{space} scalar and applied **to the biomass**.

The space limitation scalar δ_{space} is calculated as

$$\delta_{space} = \frac{1 - B_{CP}}{Max_{CP} \cdot habitat}$$

where Max_{CP} is the XXXmax parameter defining maximum biomass mgN m^{-2} and *habitat* is the proportion of habitat suitable for the consumer (area of the habitat available for the species while accounting for its optional degradation). The B_{CP} is the consumer biomass (mgN per m^2).

The Crowding effect is applied when flagXXXlim=2. It is based on ERSEM II model (Blackford 1997) and requires three extra parameters (XXX_low, XXX_sat, XXXthresh). The Crowing effect is **applied on clearance rates and not on biomass**.

$$CrowdingEffect = \begin{cases} 1 - \dfrac{\delta_{substrate} \cdot \delta_{habdegrad} \cdot (CX - \theta_{\mathrm{CXlow}}) \cdot \frac{(CX - \theta_{\mathrm{CXlow}})}{CX - \theta_{\mathrm{CXlow}} + \kappa_{\mathrm{CXsat}}}}{\delta_{substrate} \cdot \delta_{habdegrad} \cdot (CX - \theta_{\mathrm{CXlow}}) \cdot \frac{(CX - \theta_{\mathrm{CXlow}})}{CX - \theta_{\mathrm{CXlow}} + \kappa_{\mathrm{CXsat}}} + \kappa_{\mathrm{CXthresh}}} & , \; CP > \theta_{CXlow} \\ 1 & , \quad \text{otherwise} \end{cases}$$

where θ_{CXlow} is the crowding lower threshold (XXX_low, mgN m^{-2}), κ_{CXsat} is the crowding half saturation level (XXX_sat, mgN m^{-2}), $\kappa_{\mathrm{CXthresh}}$ is the crowding threshold (XXXthresh, mgN m^{-2})

By default a consumer can only have intraspecific (intra functional group) competition. However, if

flag_competing_epiff=1 then interspecies competition is included and B_{CP} is calculated as the total biomass of all corals (CORAL), epibenthic macrophytes (PHYTOBEN, SEAGRASS) and filter feeders (SED_EP_FF), which means that all epifauna will compete for space. When flag_competing_epiff=1 the user can also allow the maximum available habitat area to be higher than 1. This means that the epifaunal groups can grow on top of each other (as happens in complex reef structures). This option is set with the max_available_habitat parameter, which acts as a scalar on the available habitat. Only set this value to >1 when you have good reasons to increase the habitat availability to the epifauna.

The δ_{space} space limitation scalar is applied either to the biomass or to the clearance rate parameter C (see NOTE! box above). If more habitat limitation is needed, the *mum* values of invertebrate **epibenthic and sediment** biomass pool consumers (SM_INF, LG_INF, SED_EP_FF, SED_EP_OTHER, MOB_EP_OTHER) can be scaled by the available habitat. This scaling is applied by setting flag_benthos_sediment_link=1. If the scaling is applied the *mum* values of the epibenthic and sediment invertebrate biomass pool consumers will be multiplied (scaled) by the proportion of the cell with the suitable habitat (accounting for degradation effects)

$mum^*_{CP} = mum_{CP} \cdot habitat$

NOTE!

Habitat availability for biomass pool consumers

Traditionally in Atlantis biomass pool consumers could only use the physically defined habitat (reef, flat, soft, canyon). These physical habitats are typically static throughout the simulation, i.e. their proportion in the cell does not change (unless degraded, see below).

This has now been modified, so that biomass pool groups can also use and be affected by the dynamic biohabitats created by functional groups identified with the IsCover parameter in the *functional_group.csv*. Such habitats include corals, mussel beds and similar. To allow the biomass pool to access the biohabitats set flag_invert_biohab=1

NOTE!
Degrading the geological properties

The physical habitats are typically static throughout an Atlantis simulation, i.e. their proportion in the cell does not change. However, in some circumstances change is desired (e.g. representing a scenario of coastal development or dredging etc). In this case set flagdegrade to 1 and provide values for REEFchange_max_num, FLATchange_max_num and SOFTchange_max_num. Additionally the following parameters will be needed

REEFchange_start N	where N is the number of changes given by REEFchange_max_num
d1 d2…. dN	where d1, d2, dN etc represent the day of the run where a change begins
REEFchange_period N	where N is the number of changes given by REEFchange_max_num
p1 p2…. pN	where p1, p2, pN etc represent the period in days that a change lasts

REEFchange_mult N where N is the number of changes given by REEFchange_max_num
m1 m2…. mN where m1, m2, mN etc are the multiplicative level of change

Similarly for SOFT and FLAT.

Note, users must ensure that when multipliers are applied to habitat cover, the total cell cover of geological habitats must still sum to 1. Otherwise, the suitable area in the cell will shrink or expand. So if a REEF habitat is degraded and, for example, a multiplier of 0.5 is applied, other habitats must have multipliers >1 to compensate for the change.

Lastly the boxes where the change will occur will need to be identified (with the same change applied in all these boxes simultaneously)

Box_degraded N where N is the number of boxes in the model domain
0 0 0 0 1…. 0
where there are N entries, one for each box with 1 indicating changes occur in that box.

Table 10.7. Parameters affecting growth in consumers

Parameter	Description
X_RS	Optimal RN to SN ratio in age-structured groups
pR_XXX	Parameter controlling preference of energy allocation to RN and SN (Fig. 10.6)
flag_shrinkfat	If set to 1, negative energy budget at a given time step (when respiration costs exceed assimilated food intake) will be taken out of the RN pool. This is important for models that include respiration. If set to 0, negative energy budget is converted to 0.
flaghomog_sp	If set to 1, body condition of age-structured groups is homogenised across the entire model domain after each timestep. This means that RN and SN ratio will be the same for a functional group throughout the whole model domain. If set to 0, condition can vary among boxes depending on food availability
flagXXXlim	Flag defining what kind of space limitation should be applied to biomass pool consumers (see NOTE! in chapter 10.5.3)
XXXmax	maximum biomass of a biomass pool consumer (mgN m^{-2}), used in simple space limitation (flagXXXlim=1)
XXX_low XXX_sat XXXthresh	Lower crowing threshold, crowding half saturation and crowding threshold (all in mgN m^{-2}) used to calculate crowding effect in ERSEM based space limitation (flagXXXlim=2)
flag_competing_epiff	If set to 1, space competition will be calculated taking into account all epifaunal groups (all epifaunal groups are assumed to compete for space) rather than only intra-functional competition
max_available_habitat	Scalar for maximum available habitat for epifaunal groups. If > 1 epifaunal groups can grow on top of each other

flag_benthos_sediment_link	Flag activating additional space limitation scaling in epibenthic and sediment biomass pool consumers, where *mum* values are scaled by the available habitat in the cell
flag_invert_biohab	Flag allowing biomass pool consumers to access biohabitats, identified through IsCover parameter in the *functional_group.csv* (e.g. corals, mussel beds).
flagdegrade	Flag initiating geological habitat (REEF, FLAT, SOFT) degradation. If set to 1, additional parameters will be required as explained in chapter 10.5.3

10.6. Mortality in consumers

Mortality for age-structured groups is modelled through the numbers of individuals lost, whereas for biomass-pools it is expressed as biomass lost. The general non-predation and non-fishing mortality equation is:

$$M_{xx} = \left((mL + mQ \cdot xx + mSt + (1 - \delta_{O2}) \cdot mO + mA + mE) \cdot xx\right) \cdot mortsc$$

where *mL* is linear mortality (day^{-1}), *mQ* is quadratic mortality (day^{-1}), *mSt*, is **age-structured group only** starvation mortality (day^{-1}), *mO* is **benthic invertebrate only** low oxygen mortality (day^{-1}), δ_{O2} is the oxygen limitation scalar (see below), *mA* is an optional acidification mortality (day^{-1}), *mE* is the optional extra mortality of **age-structured groups only** (due to seabirds and fish not included in the model) (day^{-1}), *xx* is the biomass or numbers in the biomass pool or age group and *mortsc* is the optional external mortality scalar, provided through forcing files (see details here). Below these mortality terms are discussed for age-structured and biomass pool groups separately.

Note that the oxygen mortality is assumed to only apply to benthic invertebrates as they cannot easily escape the oxygen minima, whereas fish and other vertebrates will move away (ultimately disappearing from the model if the entire domain becomes too oxygen poor and if the group is dependent on the oxygen – flagO2depend).

10.6.1. Mortality in age-structured groups

The natural non-predation mortality of an age-structured consumer CX (in numbers not biomass) is calculated for each age group as:

$$M_{CX} = \left((mL + mQ \cdot Num_{CX} + mSt + mA + mE) \cdot Num_{CX}\right) \cdot mortsc$$

where *mL* is an stage-specific (juvenile and adult) linear mortality for CX (mL_XXX, day^{-1}), *mQ* is an stage-specific quadratic mortality for CX (mQ_XXX, day^{-1}), *mSt* is the starvation mortality (day^{-1}, see below), *mA* is an optional acidification mortality (see below), *mS* is the optional extra mortality due to seabirds and fish not included in the model (adopted from ERSEM I, Bryant *et al*. 1995) (mS_XXX, day^{-1}), *Num* is the numbers in an age group and *mortsc* is the optional external mortality scalar, provided through forcing files. See chapter on the forcing file for details on extra mortality scaling.

NOTE!

Terminal or senescence mortality in age structured groups and age structured biomass pools

At the end of the lifespan the individuals (biomass) can either all die (senesce) or remain in the last cohort ("plus group" in stock assessment jargon) for as long as they survive predation or other mortality factors. The senescence mortality is activated by the parameter flagsenesce.

If flagsenesce=1 all individuals above the maximum age die. The maximum age is set as the number of calendar years in an age group (XXX_AgeClassSize) multiplied by the number of age groups (NumCohorts). If flagsenesce=0 the oldest calendar year group in the oldest age group retains all the individuals (see chapter 10.9.6 on tracking calendar years in age groups).

Terminal mortality (mT) behaves somewhat similarly for biomass pools – indicating what proportion of the oldest age class dies due to old age.

The mortality calculations are done in *Vert_Mortality()* routine in **atvertprocesses.c**

NOTE!

Role of linear and quadratic mortality in multi-species models

Linear and quadratic mortality terms are included to account for mortality factors not explicitly represented in the model – such as disease, behaviour-based density dependence (e.g. male rivalry), predators not explicitly represented in the model and other factors. Ideally, the linear mortality terms should be as low as possible (1-e15) or even zero across the board, while quadratic mortality should be low (or zero) for lower trophic levels and of moderate size for the largest of the top predators (where unrepresented density dependent factors appear to be missing from the model as formulated).

If high linear and quadratic mortality values are used, the model will be insensitive to species interactions. High linear and quadratic mortalities mean that the species will be controlled by these terms alone and predation will have little to no effect on the group's biomass. As a result, the ecosystem model will end up being a set of parallel simulations of many single species models.

The exception is for top predators, such as mammals or seabirds, where predation mortality alone is typically too low to impose adequate control. In these cases, higher terms of quadratic mortalities are often applied. There is a good argument that for mammals and birds competition for breeding sites, disease and other factors indeed may be the main processes controlling population growth, and they should be represented via the quadratic mortality term.

Rates of linear and quadratic mortality are provided for juveniles and adults separately in age-structured groups. The age groups that belong to the juvenile versus adult stage are determined based on the XXX_age_mat parameter in the *biology.prm* file, which sets the first mature (adult) age group.

10.6.2. Starvation mortality in age-structured groups

Starvation mortality is only available in age-structured groups, as the condition (ration of SN to RN) is tracked for these groups only. Starvation mortality is modelled as

$$mSt = mStarve \cdot \frac{(Kthreshm \cdot optCond - W)}{optCond}$$

where *optCond* =(1.0 + X_RS)·SN, *W* is weight calculated as SN+RN, *mStarve* (mStarve_XXX, day^{-1}) is starvation mortality rate, Kthreshm is a global (same for all species) parameter defining the threshold undernourishment levels when starvation mortality starts and X_RS is a global parameter that sets the optimum RN to SN ratio (typically set to 2.65).

Note, that weight (W) used here is the **nitrogen weight**, defined as SN+RN and is different from weight used in respiration (where AFDW is used and calculated as N multiplied by the nitrogen to carbon Redfield ratio X_CN).

The calculations for starvation mortality are done in the *Vert_Mortality()* routine in **atvertprocesses.c**

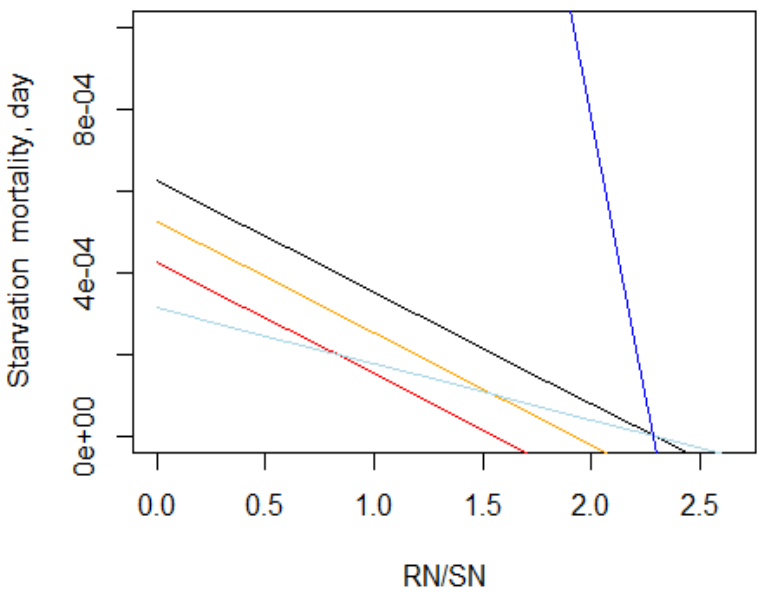


Figure 10.7. Starvation mortality (per day) for different Kthreshm and mStarve values.

Black: Kthrem=0.9, mStarve=0.001
Orange: Kthrem=0.8, mStarve=0.001
Red: Kthrem=0.7, mStarve=0.001
LightBlue: Kthrem=0.9, mStarve=0.0005
Blue: Kthrem=0.9, mStarve=0.01

10.6.3. Extra mortality in age-structured groups

The **optional** extra mortality *mS* allows users to simulate **quarterly age structured group** mortality by seabirds and fish (or other top predators) **not included** in the model. It is given by mS_SBXXX

(seabird imposed mortality on age structured group XXX) and mS_FDXXX (fish imposed mortality on age structured group XXX) parameters, each with four values, providing mortality for each quarter of the year.

Additional parameters describe the assimilation efficiency of these top predators not included in the model; the unassimilated food is all added to the ammonia pool (see chapter 10.7 for further details)

Note that the extra mortality is rarely used now, because models typically resolve all important functional groups explicitly in the model. As a result this form of mortality will likely be removed in future code releases.

10.6.4. Mortality in biomass pool consumers

Non-predation mortality in consumers is expressed in term of the biomass lost to the biomass pool, where separate calculations are done for each pool in age-structured biomass pools.

$$M_{CP} = \left((mL + mQ \cdot B_{CP} + (1 - \delta_{O2}) \cdot mO + mA) \cdot B_{CP}\right) \cdot mortsc$$

Only one linear (mL) and quadratic (mQ) mortality parameter is provided for biomass pool groups, whereas for age-structured biomass pools separate parameters are needed for each pool. The oxygen mortality depends on the oxygen limitation scalar and the oxygen mortality rate (mO_XXX; day^{-1}). The oxygen limitation mortality is determined by the δ_{02} oxygen scalar, described in chapter 10.2.8.
As in age-structured groups, the age-structured biomass pools can have senescence mortality (see chapter 10.6.1).

10.6.5. Temperature and acidification effect on mortality

Linear and quadratic mortality parameters can be temperature dependent, as described in chapter 13. If a model includes acidification effects (flagmodelpH=1) then parameters imposing additional acidification induced mortality can be specified.

Table 10.8. Parameters affecting mortality in consumers

Parameter	Description
mL_XXX	Stage-specific (juvenile and adult) linear mortality, day^{-1}
mQ_XXX	Stage-specific (juvenile and adult) quadratic mortality, day^{-1}
mS_XXX	Optional extra mortality due to seabirds and fish not included in the model, day^{-1}
flagsenesce	If set to 1, all individuals or biomass in age-structured groups and age-structured biomass pools that are above the maximum age die (see NOTE! in chapter 10.6.1)
mT	Terminal senescence mortality in age-structured biomass pools (proportion of biomass that dies from senescence when above maximum age)
mStarve_XXX	Starvation mortality, day^{-1}

mS_SBXXX	Extra mortality of age-structured groups due to seabirds not included in the model, day^{-1}
mS_FDXXX	Extra mortality of age-structured groups due to fish not included in the model, day^{-1}
mO_XXX	Oxygen mortality rate of biomass pool consumers, day^{-1}
Kthreshm	Threshold starvation level, when starvation mortality starts (chapter 10.6.2)

10.7. Waste production

The unassimilated food (faeces) and non-predation mortality products are sent to labile detritus (DL), refractory detritus (DR) and ammonia (NH) pools. The production of waste products by biomass pool and age structured group consumers are handled in the same way, but in the case of age structured groups the mortality terms are converted from numbers of individuals to biomass before being used in the following equations.

Flow of nitrogen from food intake to detritus and the ammonia pool can be illustrated with the flowchart in Fig. 10.7. It shows that first, the assimilation efficiency parameters determine how much of the grazed food is assimilated. The rest is sent to waste products. The waste products are then split between detritus and the ammonia fraction, where the amount of detritus is determined by the detritus proportion parameters (FDG, FDGDL, FDGDR).

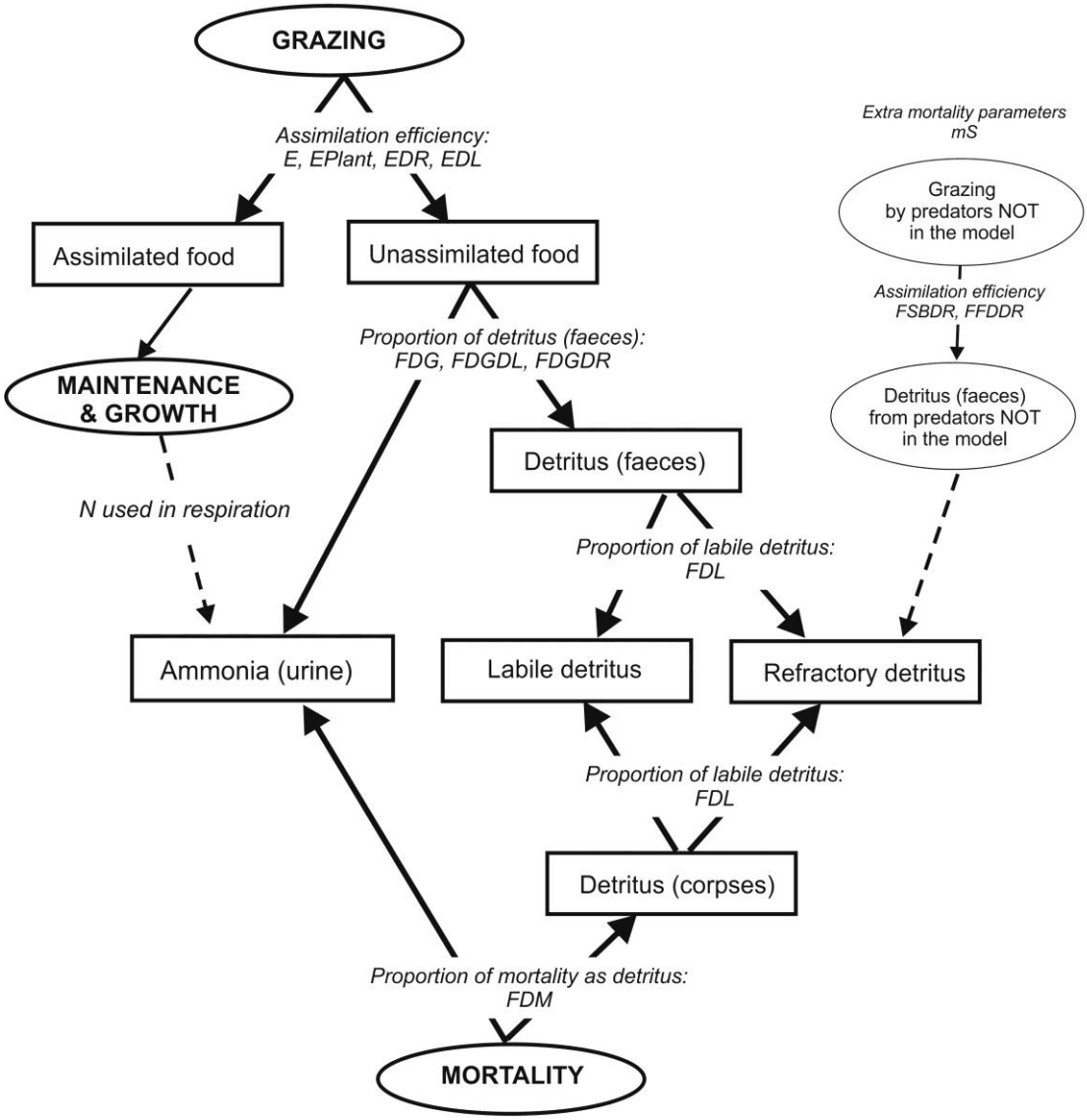


Fig. 10.8. Nitrogen flow from consumers to detrital and ammonia pools. Parameters that determine allocation between the pools are shown on the arrows. Optional fluxes are shown in dashed arrows.

Production of **labile detritus (DL)** by consumer group XX is given by:

$$W_{DL} = \left((1 - E_L) \cdot FDG \cdot \sum_{i=type} Gr_{CX,i} + (1 - E_{DL}) \cdot FDGDL \cdot Gr_{CX,DL} + (1 - E_{DR}) \cdot FDGDR \cdot Gr_{CX,DR} + FDM \cdot M_{CX} \right) \cdot FDL$$

where E_L is the assimilation efficiency on live prey (E_XXX) and plants (EPlant_XXX), $Gr_{CX,i}$ is the amount of different live prey types consumed, E_{DL} assimilation efficiency on labile detritus (EDL_XXX)

and E_{DR} assimilation efficiency on refractory detritus (EDR_XXX), FDG is the proportion of the unassimilated prey (animal, plant and catch grazing) that is sent to detritus (FDG_XXX), FDGDR is proportion of unassimilated refractory detritus that is sent to detritus (FDGDR_XXX), FDGDL is proportion of unassimilated labile detritus that is sent to detritus (FDGDL_XXX), and FDM is the proportion of mortality products assigned to detritus (FDM_XXX). FDL determines the allocation of detritus between labile and refractory pools and is defined not by species but by four types of groups: fish and sharks (FDL_fish), birds and mammals (FDL_top), water column biomass pools except CEP and PWN (FDL_wc), other non-vertebrate biomass pools (FDL_benth – note that this includes CEP and PWN not just the true benthic invertebrate groups).

Production of **refractory detritus (DR)** uses the same equation, except that:

1) the final multiplication is done not by the FDL parameter but by (1-FDL)
2) detritus produced from **all** unassimilated food consumed by optional seabirds and fish NOT explicitly represented in the model is added to DR (see Fig 10.8 and below)

$$W_{DR} = \left((1 - E_L) \cdot FDG \cdot \sum_{i=type} Gr_{CX,i} + (1 - E_{DL}) \cdot FDGDL \cdot Gr_{CX,DL} + (1 - E_{DR}) \cdot FDGDR \cdot Gr_{CX,DR} + FDM \cdot M_{CX} \right) \cdot (1 - FDL) + FFSB \cdot M_S$$

Here M_S represents the optional extra age structured group mortality by predators NOT included in the model (modelled through the mS_XXX parameter), and FFSB is the assimilation **inefficiency** of these age structured groups. The assimilation inefficiency of **seabirds NOT** included in the model is specific for each age structured group consumed and is set in parameters FSBDR_XXX where XXX is the age-structured group eaten. The assimilation inefficiency of **fish NOT** included in the model is same for all prey species and is set in parameter FFDDR.

NOTE!

Assimilation (in)efficiency of predators included and not included in the model

For **predators included in the model** (normal grazing term) assimilation efficiencies E (E, EPlant, EDL and EDR parameters) show the **proportion of assimilated food**. Fraction of unassimilated food then equals (1-E).

For optional **predators NOT included in the model** (represented through extra age structured group mortality terms mS) assimilation parameters FSBDR and FFDDR show the **proportion of unassimilated food**. This proportion of the optional extra mortality term is sent to the refractory detritus pool and the remaining fraction is **taken out of the model**.

It only makes sense to include non-zero parameters for FSBDR_XXX and FFDDR when extra mortality due to seabirds and fish NOT included in the model is represented through mS_SBXXX and

mS_FDXXX arrays.

The remaining fraction of the mortality products (1-FDM) and unassimilated food (1-FDG, 1-FDGDL and 1-FDGDR) and nitrogen used up in respiration is sent to the **ammonia pool**.

Table 10.9. Parameters determining waste production processes

Parameter	Description
E_XXX EPlant_XXX EDR_XXX EDL_XXX	Assimilation efficiency of consumer XXX when feeding on: animal prey (and catch), plant prey, refractory detritus and labile detritus
FDG_XXX	Proportion of unassimilated animal and plant food (faeces) by consumer XXX that becomes detritus
FDGDL_XXX	Proportion of unassimilated labile detrital food (faeces) by consumer XXX that becomes detritus
FDGDR_XXX	Proportion of unassimilated refractory detrital food (faeces) by consumer XXX that becomes detritus
FDM_XXX	Proportion of mortality products of group XXX that is assigned to detritus
FDL_fish FDL_top FDL_wc FDL_benth	Proportion of labile detritus of all detritus produced by each of the four groups: fish and sharks (fish), birds and mammals (top), water column biomass pools except CEP and PWN (wc) and other non-vertebrate biomass pools including CEP and PWN (benth)
FSBDR_XXX	Assimilation inefficiency of seabirds not included in the model that feed on a functional group XXX. This proportion is sent to refractory detritus pool
FFDDR	Assimilation inefficiency of fish not included in the model (same for all functional groups preyed upon). This proportion is sent to refractory detritus pool

10.8. Spawn

Spawning and recruitment is only explicitly modelled in age structured groups and age-structured biomass pools. They are handled by multiple routines in the **atdemography.c** file. In simple biomass pools reproduction is implicitly included in the growth term.

10.8.1. Sexual maturation in age-structured groups

Maturation in age-structured groups is determined by two parameters. First, a fraction of each age class that is sexually mature is set in the FSPB_XXX parameter (ogive), with an option for gradual external change in this parameter through forcing files. Second, an optional minimum length (where the allometric length/weight conversion parameters li_a_XXX and li_b_XXX are used to get length) required for sexual maturation is set in min_li_mat_XXX. If this parameter is > 0 then the no maturation will occur in an age class if it is below the maturation length even if FSPB parameter for that age class is > 0.

NOTE!

Juveniles, adults (life history stages) and sexual maturation (or spawning contributions)

Atlantis does not assume that the age of sexual maturation aligns with the ontogenetic shift in the use of habitats and prey. Sexual maturation (or the contribution to spawning in any one year by members of each age class) is given by the spawning ogive FSPB_XXX or the min_li_mat_XXX maturation length.

In contrast the age at which individuals switch from juvenile behaviour (in terms of feeding, habitat use, prey) to adult behaviour is given by XXX_age_mat. This is the first age group where adult behaviour is expressed and is modelled as a hard transition point with all individuals switching to adult behaviour at once.

10.8.2. Spawning in age-structured groups

The number of times an age-structured group spawn per year is given by NumSpawns in the *functional_groups.csv* file, although it is typically 1 (more frequent spawning is new functionality so let us know if you are having issues with it). When spawning occurs all spawn is produced in one time step. The day of the year that a species XXX spawns is set by the parameter XXX_Time_Spawn (if more than one spawning event occurs per year a vector of values is required). Although there is a parameter called XXX_spawn_period in the *biology.prm* file, it does not control the spawn period, but is just used to calculate recruitment delays (see below).

For each age group the amount of actual spawn produced is determined according to the condition of individuals in the age group at the time of spawning. The optimum spawning weight (Wsp_i) of an age group i is calculated as

$$Wsp_i = (1 + XRS) \cdot SN_i$$

where X_RS is a parameter defining the optimum ratio of RN to SN in well fed age structured groups (typically set to 2.65) and SN_i is SN of an age group i

The amount of spawn (Sp, mgN) produced by an age group is calculated as

$$Sp = \Big(\big(W_{sp} \cdot FSP - KSPA\big) - \big(W_{sp} - W\big)\Big) \cdot Num \cdot FSPB$$

where (W_{sp} –W) is the weight difference between the optimal weight for spawn and the actual nitrogen based weight of an individual (W=SN+RN), *FSP* is the proportion of weight-for-spawn used for spawn (FSP_XXX; typically ranging between 0.25 to 0.42), *KSPA* is a constant used by the spawning formulation (KSPA_XXX), *Nm* is the number of individuals in an age group, and *FSBP* is the proportion of mature reproducing individuals in the age group (FSPB_XXX). For those groups where not every adult reproduces ever year this value should be set <1 even for fully mature age classes. Note, that if individual's weight is smaller than the ideal weight at given age (defined by X_RS ratio), then the

amount of spawn produced is decreased by the weight deficit (W_{sp} –W) is subtracted from the spawn that would be produced in case of ideal weight.

The *KSPA* accounts for costs of reproduction or the amount of N taken out of weight but not included in the produced spawn. It is tuned during the parameterisation, and represents the base investment needed to build gonads and other reproductive materials. It is not however cohort specific, so the same cost is applied to each cohort.

Vertebrate spawn is calculated by the *Ecology_Age_Structured_Spawn()* routine in **atdemography.c**. This routine is called from two other routines: *Ecology_Do_Internal_Age_Structured_Spawning()* and *Ecology_Do_External_Age_Structured_Spawning()*, both in **atdemography.c**. The latter two routines convert individual spawn to spawn produced by a species in the model (or individuals that migrated outside the model domain), summing spawn over age groups to get the total spawn produced by the group.

10.8.3. Maturation and spawn in age-structured biomass pools

In age-structured biomass pools the proportion of mature biomass in an age pool is set by the FSPB_XXX parameter. The amount of spawn (mgN) produced by an age biomass pool *i* is then calculated as

$$Sp_i = B_i \cdot FSP \cdot FSPB_i$$

where B_i is the biomass of the age pool (mgN), FSPB is the proportion of the mature biomass in the pool *i* (FSBP_XXX), and FSP is the proportion of spawning biomass in the mature biomass (FSP_XXX)

The amount of nitrogen taken from the biomass pool after spawning is not necessarily equal to the spawn produced (Sp), but is set by the parameter prop_spawn_lost_XXX. This lost biomass represents semelparous reproduction and is used to represent the mortality of exhausted adults or can be set = 1 for full semelparity (all individuals die after reproduction, as in some squid species). The lost biomass is treated as normal mortality and sent to DL, DR and NH3 pools.

Reproduction of age structured biomass pools is done in the *Invertebrate_Reproduction()* routine in **atdemography.c**

Table 10.8. Parameters determining maturation and spawn

Parameter	Description
FSPB_XXX	A fraction of each age class in an age-structured groups and age-structured biomass pools that is sexually mature
min_li_mat_XXX	Minimum length required for sexual maturation in age-structured groups (calculated using length-weight conversion parameters li_a_XXX and li_b_XXX)
XXX_age_mat	The first age group where adult behaviour is expressed in distribution, feeding, prey availability. It is modelled as hard transition and can be different from sexual maturation defined by FSPB_XXX parameters

XXX_Time_Spawn	Day of the year that a species XXX spawns (all spawn occurs at once)
FSP_XXX	Proportion of ideal weight for spawn that is used for spawn
KSPA_XXX	Constant used by the spawning formulation, it represents reproductive costs and does not contribute to spawn
prop_spawn_lost_XXX	Amount of nitrogen taken from age-structured biomass pool after spawning. Used to represent both nitrogen lost to spawn and reproduction related mortality

10.9. Recruitment and ageing

Atlantis does not currently model the larval period explicitly. In fully age-structured groups and age-structured biomass pools nitrogen produced as spawn is temporarily taken out of the model and is returned as recruits after a set larval period. In simple (non-age structured) biomass pools reproduction is not modelled explicitly. The larval period is set as

XXX_larval_period = XXX_Recruit_Time + XXX_Spawn_Period + optional_small_scale_random_variation

After this time recruits start "arriving" into the model throughout the period set in Recruit_Period_XXX. Nineteen different recruitment options are currently available for converting the base spawn amount into the biomass and numbers of recruits. The most common function applied for fish species is the Beverton-Holt (BH) function, dependent on both spawn produced and biomass of the species. For mammals, birds and some sharks a constant recruit production per adult is often used.

Some of the recruitment options ignore the spawn altogether, others ignore both spawn and species biomass, others use random number generation and others allow for scaling of recruitment by plankton production. In most cases the amount of nitrogen in spawn is not equal to the amount of nitrogen in recruits. Recruitment is modelled in the same way in age-structured groups and in age-structured biomass pools, but in age-structured groups the recruited biomass is subsequently converted to numbers, using the SN and RN values of recruits provided by the user (KWSR_XXX and KWRR_XXX respectively).

The amount of recruits produced can be influenced by one or several of these factors, and none of them are obligatory:

1) amount of spawn produced
2) stock biomass
3) phytoplankton
4) zooplankton
5) external recruitment forcing

Recruited individuals are attributed to the first year of the first age group. The day before the recruits "arrive", all other individuals age one year up, and can be allocated into a different age group depending on the number of years in each age group. Ageing is done in one time step, regardless of whether recruits arrive all at once or during a certain period of time.

10.9.1 Routines used to calculate recruitment in age-structured groups

Several routines in Atlantis are involved in obtaining the final number of recruits. For age-structured groups (vertebrates), the total spawn produced by a species (calculated in the routines above, in the chapter on spawning) is passed to:

1) The *Ecology_Find_Embryos()* routine, which applies environmental scalars, such as scaling by plankton production. It also applies external recruitment forcing, for cases where recruitment time series are supplied. In the case of Beverton-Holt or Ricker recruitment, which are dependent on the spawn or species biomass, nothing is done in this routine.
2) The value above is then passed to the *Get_Recruits()* routine which calculates the recruitment (*Rs*) depending on the chosen recruitment option.
3) Next, two routines – *Get_Vertical_Recruit_Position()* and *Find_Final_Recruit_Distribtuion()* – determine the horizontal and vertical distribution of new recruits, depending on the distribution parameters provided by the user, whether the group has maternal care, set as flagmotherXXX or bear live young, set as flagbearliveXXX (in which case the recruits arrive where the adults are), and any optional environmental limitations (salinity, temperature, oxygen). The flagmotherXXX parameter allows users to set semelparous reproduction, where value of <0 kills reproducing (FSPB_XXX proportion) adults after reproduction.
4) Next, the routine *Get_Settlers()* distributes the recruits on the temporal scale, depending on the length recruitment period and shape of the recruitment distribution.
5) Finally, immediately before the recruits arrive, all other age groups age one year up. This is done in *Update_Ageing_Numbers()* and a few other routines in the **atdemography.c** (which track the relative proportions per annual age class per age group)

Next we will give details on how Atlantis calculates recruitment (*Rs*) from spawn (*Sp*) for different recruitment options. Equations below combine calculations done in both *Ecology_Find_Embryos()* and *Get_Recruits()* routines.

10.9.2. Stock-recruitment options

The most common Beverton-Holt recruitment (selected as flagrecruitXXX=3) is implemented as:

$$Rc = \frac{Sp \cdot BHa}{Biom + BHb}$$

where *BHa* and *BHb* are species-specific Beverton-Holt alpha and Beverton-Holt beta parameters (BHalpha_XXX and Bhbeta_XXX, mgN), *Sp* is the spawn produced (mgN, see Spawn chapter) and *Biom* is the species total biomass. Note that because this *Rc* is dealing with nitrogen and not recruits entering a fishery, the obtained values will not match those given in stock assessment models.

NOTE!

Converting recruit biomass *Rc* to numbers

The *Rc* is calculated in nitrogen biomass. For age-structured groups it is then converted to numbers using the recruit SN and RN values provided by the user as KWSR_XXX and KWRR_XXX parameters .

Other recruitment options are:

1) Constant recruitment flagrecruitXXX=1

Recruitment is constant through time and is independent on the stock size. The KDENR_XXX parameter in *biology.prm* file gives the number of "arriving" each year.
$Rc = KDENR$

This is a useful representation if recruitment is dictated by a large external stock not represented in the model.

2) Recruitment determined by chlA concentration flagrecruitXXX=2

Recruitment is calculated separately in each cell. It is determined by the amount of chlA in the cell, species specific PP_XXX parameter relating recruitment to primary production and a ref_chl parameter providing the reference ChlA level (from both LG_PHY and SM_PHY)

$$Rc = PP \cdot \frac{chlA}{ref_{chl}}$$

3) Beverton-Holt spawn and species biomass dependent recruitment flagrecruitXXX=3

This is the most commonly applied recruitment option for fish and is described above.

4) Random lognormal recruitment flagrecruitXXX=4

In this option recruitment is completely random and does not depend either on the spawn produced or on the biomass of the species. The centre and variation of the lognormal distribution curve (δ_{lonorm}), from which the recruitment is sampled, is set by the lognorm_mu and lognorm_sigma parameter respectively. Further, each species has a specific recruitment multiplier XXX_log_mult *(l_multiplier* in the equation below) converting the lognormal value, which is typically in the range from 0 to 1 (depending on lognorm_mu and lognorm_sigma values) to the actual number of recruits.

$$Rc = \delta_{lonorm} \cdot l_{multiplier}$$

Note, that even though Atlantis is fundamentally a deterministic model (the same results are obtained with same sets of conditions and parameters) the lognormal recruitment option would generate different results for each simulation, adding some stochasticity to Atlantis simulation outcomes.

5) Recruitment determined by phyto- and zooplankton abundance flagrecruitXXX=5

Recruitment is calculated similarly to the case 2, but now depends not on ChlA concentration but on the sum of the large phytoplankton and small zooplankton biomass, such that

plankton=LG_PHY+LG_ZOO+MED_ZOO+SM_ZOO

The same two parameters PP_XXX and ref_chl used in the primary production based option are also

used here to relate plankton levels to recruitment. Note that in ChlA or plankton dependent recruitment, spawn or species biomass is not taken into account. **SM_PHY is not included here** because it is not believed to be an important fish larvae food source.

$$Rc = PP \cdot \frac{plankton}{ref_{chl}}$$

6) Beverton-Holt recruitment with lognormal variation and plankton dependency flagrecruitXXX=6

This recruitment option combines options 3, 4 and 5. Recruitment depends on spawn and species biomass, as in option 3, plankton levels as in option 5, and is also modified with a lognormal scalar δ_{lonorm} calculated based on lognorm_mu and lognorm_sigma parameters, as described in option 4.

$$Rc = \delta_{lonorm} \cdot \frac{Sp \cdot BHa}{Biom + BHb} \cdot \frac{plankton}{ref_{chl}}$$

7) Beverton-Holt recruitment with encouraged recovery flagrecruitXXX=7

Recruitment is calculated as in option 3, but a species-specific recruitment encouragement multiplier recover_mult_XXX is applied to *Rc* if a species biomass is below a threshold set in the recover_trigger parameter. This threshold levels shows a proportion of the initial stock biomass below which the recruitment multiplier is applied. The recruitment multiplier is applied for the number of years given in recover_subseq parameter. Moreover, the recruitment multiplier will only kick-in after a set period of days after the threshold stock proportion has been reached. This lag between low stock levels and encouraged recruitment is set in recover_span parameter. This recruitment option represents adaptive stocking or density dependent processes that only activate at low recruit levels.

8) Beverton-Holt recruitment with prescribed recovery flagrecruitXXX=8

Same as above, but the recruitment encouragement multiplier recover_mult_XXX is applied to *Rc* regardless of the stock levels for the period set in recover_span and recover_subseq parameters (potentially represented a period where an unknown environmental or ecological driver facilitated recruitment).

9) Ricker recruitment flagrecruitXXX=9

Ricker recruitment is not dependent on the spawn produced, but is based on biomass and the two parameters Ralpha_XXX and Rbeta_XXX. The *Rc* is calculated as

$$Rc = Biom \cdot e^{Ralpha \cdot \left(1 - \frac{Biom}{Rbeta}\right)}$$

The Ricker recruitment parameter Ralpha and Rbeta are sometimes interpreted as r or intrinsic population growth and K or carrying capacity (see here for example)

10) Beverton-Holt recruitment based only on species biomass flagrecruitXXX=10

Calculated similarly to option 3, but instead of spawn (Sp), only **total species** biomass is used.

$$Rc = \frac{Biom \cdot BHa}{Biom + BHb}$$

11) Recruitment is equal to spawn flagrecruitXXX=11

This is the simplest recruitment option, where the nitrogen in recruits is simply equals the nitrogen spawned.

$$Rc = Sp$$

12) Fixed number of offspring per individual flagrecruitXXX=12

In this option the KDNER_XXX parameter provides the number of offspring per individual. This recruitment option is often applied for mammals, birds and viviparous sharks. When setting this option remember that Atlantis models an average individual, hence the actual number of offspring per female may need to be halved in a population of equal sex ratio or reduced if there is in utero dynamics (such as siblicide in sharks), or if mortality sources outside the model domain (e.g. land-based predation of turtle eggs) needs to be implicitly represented.

In this case, recruitment produced by an age group is calculated as

$$Rc = KDENR \cdot FSPB \cdot Num$$

where *FSPB* is the proportion of mature individuals in the age group (FSPB_XXX) and *Num* is the total number of individuals in the age group. Note, that for this recruitment option spawn will still be calculated and taken out of the model, but recruitment will only be determined by the number of offspring per individual multiplied by the number of spawning individuals.

13) Recruitment forced from time series file flagrecruitXXX=13

In this option the recruitment value Rc is taken from externally forced time series files. The name of the recruitment time series file must be provided in the *force.prm* file. In this option, spawn will still be calculated and taken out of the model, but recruitment will only be determined by external forcing.

14) Larval recruitment – This option leaves space for explicit larval recruitment to be tracked, but it is currently not available in Atlantis.

15) Jacknife recruitment flagrecruitXXX=15

This option allows two levels of recruitment depending on the spawning stock biomass (SSB), calculated as the wet weight in tonnes (not nitrogen weight!) of spawning individuals and a set threshold B_{jack}, calculated as a fraction of total biomass:

$$B_{jack} = jack_b \cdot Biom$$

If SSB < B_{jack}

$$Rc = jack_a \cdot SSB$$

If SSB ≥ B_{jack}

$$Rc = jack_a \cdot B_{jack}$$

The jack_a_XXX is interpreted as slope of recruitment, usually set as biomass of recruits / SSB. The jack_b_XXX is empirical proxy for a relationship with the carrying capacity (in the North Sea they used 0.3 * long term average total biomass). This recruitment relationship is taken from the North Sea stock assessment - see a more detailed description on the wiki

16) Coral recruitment flagrecruitXXX=16

Coral recruitment is determined by three specific coral recruitment parameters CrecruitA_XXX, CrecruitB_XXX and CrecruitC_XXX and a KDENR_XXX parameter setting a total number of recruits.

The recruitment is calculated as

$$Rc = KDENR + CrecA \cdot max\left(0, \left(CrecB - e^{-CrecC \cdot B^*}\right)\right)$$

where B* is the spawning biomass (if corals are modelled as age-structured biomass pools the spawning biomass will be determined by the FSPB parameter).

17) Baltic Sea version of the Ricker recruitment flagrecruitXXX=17

The Baltic Sea version of the Ricker recruitment uses wet weight (not nitrogen based!) biomass B_{WW} in **thousands of tonnes** and calculates Rc as:

$$Rc = 1000 \cdot Biom_{WW} \cdot Ralpha \cdot e^{-1 \cdot Rbeta \cdot Biom_{WW}}$$

18) Multiple time series of recruitment – specific forcing in separate boxes flagrecruitXXX=18

This option allows forcing of recruitment from specific boxes. Numbers of recruits are forced by external forcing files and they only arrive in boxes specified in the forcing files. The horizontal and vertical distribution of these recruits is then determined according to the general rules, described in

the next chapter below. See further description on Atlantis wiki.

19) Beverton-Holt recruitment based only on spawn only flagrecruitXXX=19

Calculated similarly to option 3, but instead of biomass, only species spawn (*Sp*) is used.

$$Rc = \frac{Sp \cdot BHa}{Sp + BHb}$$

Finally, in addition to the recruitment options above, it is also possible to supplement recruits with external stocking. This is used to imitate, for example, supplementary stocking of salmon. Recruit stocking for a species XXX is set setting flagstockingXXX=1 and providing details on recruit stocking files. These files, and the code the recruit stocking triggers, are the same used to apply multiple time series of recruitment (flagrecruitXXX=18). See this post on the wiki for further details.

NOTE!

Interspecies dependency scalar

It is possible in Atlantis to scale recruitment by the biomass of another species Y. This could be used to simulate, for example, consumption of cod eggs by sprat in the Baltic Sea. The interspecies dependency of species XXX recruitment is set with a flag intersp_depend_recruit_XXX, where 1 means linear scaling and 2 is inverse scaling. Parameter intersp_depend_sp_XXX gives the ID number of species Y (from the functional_group.csv file). The intersp_depend_scale_XXX is a scalar to apply to recruitment, showing the influence of the biomass of species group Y on the recruitment of XXX.

If recruitment of species XXX is dependent of species Y, and intersp_depend_recruit_XXX=1 recruitment Rc of XXX is calculated as

Rc=Rc*intersp_depend_scale*biomassY

If intersp_depend_recruit_XXX=2

Rc=Rc*intersp_depend_scale/biomassY

NOTE!

Calculating recruitment for different stocks

If several stocks per species are used (identified with NumStocks parameter in the functional_ group.csv file) the procedures described above calculate recruitment for each stock separately, using the biomass of that stock rather than the total species biomass. The user also has an option to use stock specific recruitment scalars recSTOCK_XXX to upscale or downscale recruitment in different stocks.

10.9.3. Environmental, habitat and external recruitment scalars

The recruitment value *Rc* calculated above can be further modified by habitat availability, biomass of other species, environmental conditions, and additional recruit supplementing (stocking). All of these factors are optional and will be applied as multipliers on the base recruitment. Most of them affect recruitment in a negative way. When several scalars are applied (pH, temperature, contaminants), the user must be careful about the multiplicative nature of all these scalars, as they can drastically reduce recruitment.

The pH, temperature, salinity, contaminant and external time series scalars are calculated in *Get_Enviro_Recruit_Forcing()* routine. They use pHcorr, Tcorr and Scorr values that are applied to different physiological processes of a species and calculated in relevant pH and temperature routines (see chapter 13).

1. pH scalar

If a species is sensitive to pH (flagfecundsensitive_XXX=1) its recruitment will be scaled by a pH scalar, based on the pHcorr scalar calculated in the pH related routines (see below). The scalar is calculated as
pHscalar=1/pHcorr

as it assumes that pH effect on recruitment will be negative (scalar will be <1)

2. Temperature scalar

If a species is sensitive to temperature (flagtempsensitiveXXX) **with the kind of effect dictated by the recruitment temperature dependency** (flagq10receffXXX=1 or 2 – 1 for poorer when cooler, 2 if poorer when warmer, 0 indicates no actual effect regardless of flagtempsensitiveXXX setting) then recruitment is scaled by a multiplicative effect of pH and temperature (pHscalar*Tcorr). The pH effect will be 1 if a species is only sensitive to temperature and not pH. If species is temperature insensitive (flagq10receffXXX=0) only the pHscalar on recruitment is applied. If the pHscalar·Tcorr > 1 then the inverse is applied to scale the recruitment (1/(pHscalar·Tcorr)) as the model assumes that effect of temperature and pH on recruitment will be negative

The **effect of temperature** on recruitment of temperature sensitive species (flagtempsensitiveXXX=1) is also simulated through minimum and maximum spawn temperatures (XXX_min_spawn_temp and XXX_max_spawn_temp). If temperature in the cell is outside these ranges, then the scalar is set to 0 and

recruitment **in that cell** will be zero.

3. Salinity scalar

If a species is sensitive to salinity (flagSaltSensitive_XXX=1) the recruitment is first scaled by the Scorr scalar, and second, the models checks whether the salinity conditions in the cell are within the minimum and maximum spawn salinity ranges (XXX_min_spawn_salt and XXX_max_spawn_salt). If salinity in the cell is outside these ranges, then the scalar is set to 0 and recruitment **in that cell** will be zero.

4. Contaminant scalar

If the model tracks contaminants, then a scalar on recruitment due to contaminants is also calculated and applied.

5. Externally forced scalar

Users can supply external recruitment forcing scalar through time series files, listed in the *force.prm* file (see chapter 8). The external scalar will act as a multiplier on all other scalars calculated above. To indicate that external environmental recruitment forcing will be used set flagtsforcerecruit=1 in the *biology.prm* file. If this flag is set to 1, but no external forcing is provided then Atlantis will quit. This forcing represents environmental influence on recruitment (such as El Nino etc) and is called from *Get_Enviro_Recruit_Forcing()*.

On top of these scalars an additional external forcing can be applied to recruitment, in the same way as forcing is applied to growth, mortality or other processes. This recruitment forcing is done separately to allow further representation of external processes, such as for example contaminant spill. This final kind of *force.prm* recruitment scalar is applied within the *Get_Final_Distribution()* routine and is called from *Ecology_Get_Recruitment_Scalar()*.

6. Habitat scalar

Habitat scalar on recruitment acts as an additional multiplier on all environmental scalars above. It is applied if species recruitment is habitat dependent (XXX_rec_HabDepend=1). The habitat scalar is calculated as a proportion of suitable habitat in the cell.

10.9.4. Spatial distribution of recruits

Once the recruitment biomass and/or numbers are calculated Atlantis will first define their vertical and then horizontal distribution. This is done in two routines *Get_Vertical_Recruit_Position()* and *Find_Final_Recruit_Distribtuion()*. The first routine uses the XXX_recruit_vdistrib parameters. This parameter provides entries for a maximum number of water layers so the routine rescales the vertical recruit distribution in each box depending on how many layers it has and weighs the vertical distribution towards the bottom or surface depending on whether a species is identified as demersal (flagdemXXX=1) or not.

Next, the routine *Find_Final_Recruit_Distribtuion()* determines the final spatial distribution of recruits.

It applies all the environmental and other scalars calculated above, and finds the final recruit number for each box and layer given the vertical and horizontal distribution parameters. The base horizontal distribution, just like the vertical distribution is given by the user in XXX_recruit_hdistrib parameter, which simply gives proportional number of recruits allocated in each box. This potential distribution is modified based on whether the recruits actually recruit to the same location as the adults (flaglocalrecruit_XXX=1) and whether local temperature, salinity, habitat conditions are suitable.

The final number of recruits in a cell is then given as

Recruits_in_cell = AllRecruits · All_scalars · Vertical_distribution · Horizontal_distribution

It is also possible to apply an externally forced larval dispersal matrix (if flaglocalrecruit_XXX is set to 3) to modify the horizontal and vertical distribution of the recruits. This is described further here.

NOTE!

Minimum and maximum salinity and temperature can greatly reduce recruitment

It is important to realise that **minimum and maximum spawn temperature and salinity act as scalars on recruitment in a box**. Unlike the way in which settled individuals will contract away from unsuitable conditions during movement, recruit distributions **do not** contract to the boxes with suitable conditions but rather **kill all the recruits** that end up in unsuitable boxes. This means that if many boxes have conditions that are outside the suitable recruitment values the total recruitment in the model can be greatly reduced.

10.9.5. Factors defining temporal arrival of recruits

Once the total number of recruits to arrive in each cell is calculated using the procedures above, their arrival is spread throughout the Recruit_Period_XXX. This is done by the *Get_Settlers()* routine. Two options are available for this, controlled with a global (applied to all species) flagtrecruitdistrib parameter. The recruit arrival can be flat, so that same number of recruits arrive every day (flagtrecruitdistrib=1) or it can be humped (lognormal) (flagtrecruitdistrib=0).

If a lognormal distribution is used the user should also provide rec_m and rec_sigma parameters, determining the shape of the lognormally shaped curve (the same number of recruits will arrive, but not evenly spread through time).

10.9.6. Ageing of individuals

As recruits "arrive" they enter the first age group of a species. For species that have more than one calendar year per age group they are entered into the first year of the first age group.

NOTE!

Tracking calendar years within Atlantis age groups

Although some species have several calendar years in one age group for the purposes of simplifying the ecological processes, for aging Atlantis actually tracks individuals of each calendar year within age group separately. This is illustrated in the left Figure 10.9 below where three calendar years (bars of the same colour) are tracked in each of the four age groups (AG1-AG4). The ageing routines move the individuals from one calendar year to the next (solid line arrows).

All individuals in one age group AG are assumed to be identical for predation, reproduction, fishing and other processes. Mortality can be proportionally allocated across annual age cohorts within an age class. However, it is also possible to dampen strong year classes during the ageing process and equalise their relative abundance. This assumes that the most abundant year classes will be more affected by predation and fishing that less abundant ones and that through time the relative abundance of year classes will even out. The figure on the right shows that the abundant youngest year class in year t becomes less abundant in year t+1, while the relative abundance of the other two year classes in that age group increases (shown with bold outlines). In the year t+1, the newly arrived year class is shown in green, and the oldest year class from age group 1 is now in the age group 2.

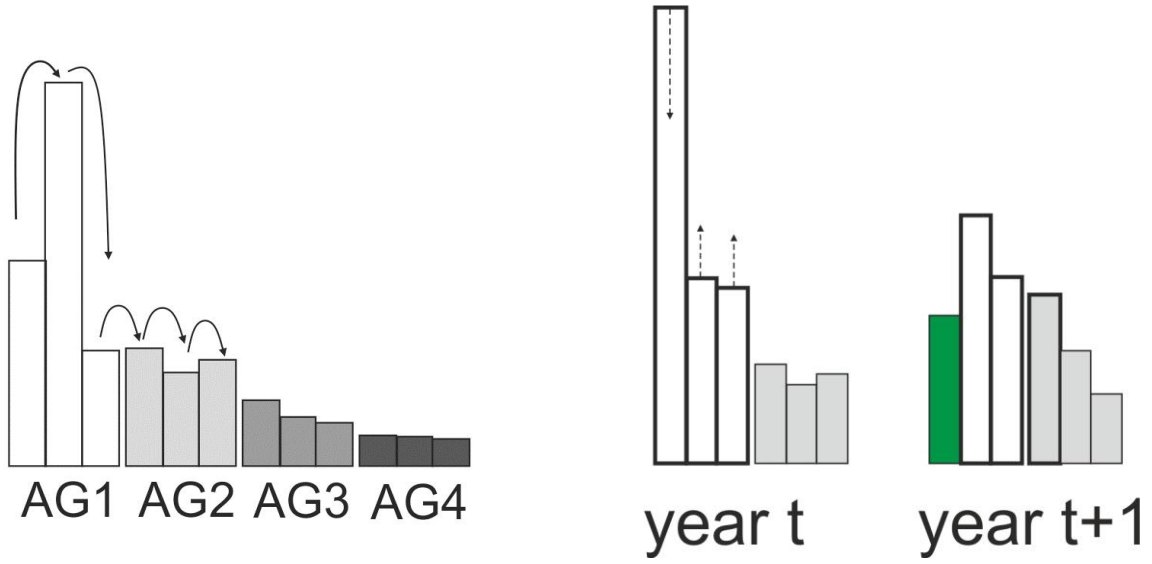


Figure 10.9. Tracking of calendar years in Atlantis age groups. Left: three calendar years for each of the four age groups (AG1-AG4) illustrated. Right: dampening of strong year classes through time.

Dampening of strong year classes during ageing routines is controlled by the flagrecpeakXXX parameter used in the *Update_Age_Distrib()* routine. If flagrecpeakXXX=0 then dampening of strong year classes will use the recruitRangeFlat multiplier, and if the flagrecpeakXXX=1 then dampening of strong year classes will use the recruitRange multiplier. By setting the values for recruitRangeFlat and recruitRange the user can control how much dampening will occur. If no dampening is wanted set the values to 1.

The dampening effect and the related parameters described above are **irrelevant for species that only have one calendar year in an age group**.

11. DISTRIBUTION AND MOVEMENT

The spatial distribution of organisms in the model domain is first set in the *initial_conditions.nc* file and is then updated at each time step. The main routine handling the distribution and movement is *Ecology_Total_Verts_And_Migration()*for fully age structured groups and *Ecology_Invert_Migration()* for biomass pools.

The simplest form of spatial distribution is called "prescribed movement" and is done when species do not have any density dependent movement (XXX_ddepend_move=0) or migrations (flagXXXMigrate=0). In this case the spatial distribution is determined or prescribed by FXXX_SY parameters where the Y is typically 1-4 (representing seasonal shifts), but can now be set to whatever number the user desires (see details here). These vectors determine the horizontal distribution, whereas VERTnight_XXX and VERTday_XXX parameters define vertical distribution. This is described in chapter 11.1

When XXX_ddepend_move=1 a species is set as sedentary. This means that once an individual (biomass) recruits into a box, there will be no further horizontal movement. So no prescribed movement parameters (FXXX_S1 to FXXX_S4) will be applied! The vertical movement is still allowed.

When density dependent movement is used (XXX_ddepend_move=2, 3, 5, 7) the parameters for "prescribed" horizontal movement are mostly or entirely overwritten by the movement and distribution due to food availability (except for the movement of the spawning and not feeding individuals during the spawning period, see below). This is a very important aspect of the model and makes full use of the spatially explicit modelling approach. However, for many models parameterising the density dependent movement can be challenging and it is often applied only to a few key species. This is because turning the density dependent movement without proper parameterisation can lead to all individuals accumulating in one box and completely depleting the food resource, or switching between boxes too rapidly. This is described in chapter 11.2

When XXX_ddepend_move=4 no explicit movement, vertical or horizontal, is permitted.

It is also possible to use home range movement (XXX_ddepend_move=6)

Finally, when species have annual migrations then the spatial distribution is also corrected for the proportion of individuals (or biomass) moving into and out of the model domain.

Below the options determining spatial distribution are described in further detail

11.1 Spatial distribution without migration or density dependent movement

Atlantis updates horizontal and vertical distribution of species at each time step. When no density dependent or migratory movement is used, this is simply done by interpolating the proportion of biomass provided by the user in each box for each quarter of the year given in the FXXX_SY (typically FXXX_S1 to FXXX_S4) parameters. Note, that for the remainder of this discussion it will be assumed that numMoveEntries is set to 4 (representing traditional quarterly shifts in movement distributions). The same logic applies if numMoveEntries does not equal 4, just adjust the frequency and number of entries accordingly.

The horizontal distributions are given separately for adult and juvenile stages of the species, and the transition from juveniles to adults is given by the XXX_age_mat parameter, indicating the first mature age class (this does not mean that all individuals in that age class are mature or will spawn, see the chapter on Spawning for further details).

The abundance of species CX (juvenile or adult) in a cell j ($Abund_j$) at a given timestep is calculated as

$$Abund_j = Abund_{total} \cdot \left(\varepsilon \cdot \left(F_{Q+1,j} - F_{Q,j}\right) + F_{Q,j}\right)$$

where $Abund_{total}$ is the abundance of the species in the entire model domain (in biomass for biomass pools and in numbers), ε is the proportion of the quarter of the year that has passed, whereas F_Q and F_{Q+1} is the proportion of biomass in a cell j in quarter Q and Q+1 provided by the user (FXXX_S1 to FXXX_S4 parameters). Note that if Q is the last quarter of the year, then Q+1 is the first quarter of the year.

The equation above can be illustrated with a simple example, where $Abund_{total}$ =100, F_Q =0.1, F_{Q+1} =0.2, and ε =0.5 (the current time step is half way between the start of the quarter Q and quarter Q+1. In this case the $Abund_j$ = 100·(0.5· (0.2-0.1)+0.1)=100·(0.05+0.1)=15. This shows that while at the start of the quarter Q the proportion of the total biomass in box j is 0.1 and at the start of the quarter Q+1 it is 0.2, at the timestep half way through the quarter Q the proportion will be 0.15.

Once Atlantis calculates the horizontal distribution for a given time step it will then apply the vertical distribution VERTnight_XXX and VERTday_XXX parameters to distribute the individuals (or biomass) in each box among the vertical layers.

NOTE!

Effects of environmental sensitivity on spatial distribution

If a species is sensitive to temperature (flagtempsensitiveXXX), salinity (flagSaltSensitive_XXX) or oxygen (flagO2depend global parameter that affects all species), then its spatial distribution will be affected by the values of these environmental variables in a cell.

If the values in the cell are outside the tolerated range of the species – given by the min_move_temp_XXX, max_move_temp_XXX, min_move_salt_XXX, max_move_salt_XXX, min_O2_XXX parameters – then the abundance of the species in that cell will be set to 0 and the species distribution will be contracted to cells that have suitable conditions.

If none of the cells have suitable conditions the species will die.

11.2 Spatial distribution with density dependent movement

Vertebrates and pelagic invertebrates can have density dependent movement, which means that the spatial distribution will be determined by the food availability in different boxes. At each time step Atlantis will evaluate potential feeding rates of each age group of a species (for which density dependent movement is turned on) in each box of the model. This will construct a spatial landscape of boxes determined by the potential feeding possibilities they provide. The distribution at each step will then be determined according to this landscape and two parameters, setting the threshold difference and ranking (see below). Two types of density dependent movement are available in Atlantis – basic (XXX_ddepend_move=2) and "sticky" (XXX_ddepend_move=3). We first describe how the basic density depdent movement works.

First, Atlantis calculates potential consumption (amount of assimilated food) for each age group in each of the model cells j:

$$A_{potential,j} = A_{potential,j} \cdot E_1$$

where $Gr_{potential,j}$ is the potential grazing term of an age group if it was in the box j and E_1 is the assimilation efficiency on the live prey.

NOTE!

Currently, potential consumption is calculated assuming the modified Holling type II response (predcase=0) and hard feeding window. If other feeding functional responses are used, be aware that the density dependent movement might not be exactly right (though the relative forage field may be sufficient that it makes little difference). If you are concerned about this please contact the model developers as they do intend to allow for more functional responses in the movement code.

Also, currently the roc_wgt is a global parameter, which means that the same weighting of the potential growth rate surface (i.e. how clumpy to make it around hotspots) is applied to all species regardless of their identity, life history stage or size.

Next, Atlantis assess whether this food amount is higher than the threshold γ_{thresh} that would trigger movement. This threshold set with global k_roc_food parameter and SN amount in the age group *i*:

$$\gamma_{thresh} = SN_i \cdot kroc_{food}$$

The k_roc_food parameter is a proportional increase of SN weight (new weight / old weight) that would trigger movement into a better box, and it should always be >1 (as fish should not move to boxes where food conditions are worse)

The spatial cells in the model are then ranked depending on whether the $A_{potential}$ is larger or smaller than the threshold γ_{thresh}

$$A^*_{potential,j} = A_{potential,j} \cdot roc_{wgt} \qquad if\ A_{potential,j} > \gamma_{thresh}$$

$$A^*_{potential,j} = \frac{A_{potential,j}}{roc_{wgt}} \qquad if\ A_{potential,j} < \gamma_{thresh}$$

New proportional abundance $F_{j,t+1}$ in the cell j is then calculated as

$$F_{j,t+1} = \frac{A^*_{potential,j}}{\sum A^*_{potential,j}}$$

This can be illustrated with a hypothetical 4 box example with different food availability (potential assimilation rate) and an age cohort that has SN=100, k_roc_food=1.5, and roc_wgt=100

	Box1	**Box2**	**Box3**	**Box4**
Hypothetical potential assimilated food amount ($A_{potential}$)	100	50	200	1000
Current proportional distribution	0.25	0.25	0.25	0.25
Threshold values for each box (γ_{thresh}) assuming k_roc_food=1.5	150	150	150	150
The threshold condition $A_{potential}$ versus γ_{thresh}	100<150	50<150	200>150	1000>150
Modified $A^*_{potential}$ after evaluating the threshold condition, where roc_wgt=100 ($A_{potential}$*roc_wgt or $A_{potential}$/roc_wgt)	100/100 = 1	50/100 =0.5	200*100 =20000	1000*100 =10^5
New distribution $A^*_{potential} / \sum A^*_{potential,j}$ (where $\sum A^*_{potential,j}$=120001.5)	0.00	0.00	0.17	0.83

It can be seen that after one time step the distribution can shift quite drastically. Atlantis users are encouraged to explore different values of roc_wgt and k_roc_food parameters on possible changes in the distributions. To prevent species teleporting about the model based on short term forage field shifts, the movement rate is adjusted by the swimming speed given in Speed_XXX parameter. The rate at which individuals (biomass) can move from one box to another is scaled by this swimming speed divided by the width of the box they are currently in.

It is also possible to use a more conservative approach to density dependent movement by using "sticky" movement XXX_ddepend_move=3. In this case the same test calculating the potential food assimilation is applied to all boxes. However, instead of redistributing all individuals across the model domain, the individuals that are in boxes where $A_{potential,j} > \gamma_{thresh}$ remain where they are and only the individuals from boxes where $A_{potential,j} < \gamma_{thresh}$ move throughout the model domain. So in the example above, the Box 3 has passed the test yet the proportion of individuals in this box still decreased from 0.25 to 0.17, because Box 4 is better. When "sticky" density dependent movement is used, only individuals from Box 1 and Box2 are redistributed.

NOTE!

Spawning will affect density dependence movement

The density dependent movement mostly overrides the prescribed movement parameters given in the FXXX_S1 to FXXX_S4 (etc) parameters. However, for species that do not feed while spawning (feed_while_spawnXXX=0) the proportion of the age group spawning will not migrate according to the density dependence rules but according to the prescribed movement rules.
There are two options to control this for the simple (not "sticky") density dependence movement by setting XXX_ddepend_move to 5 or 7.

When XXX_ddepend_move=5 only prescribed movement for the entire species will be used during the spawning period and only simple (non "sticky") density dependent movement will be used at other times. When XXX_ddepend_move=7 the prescribed movement will be completely ignored and only simple density dependent movement will be applied at all times.

Note, that currently there is no way to eliminate prescribed movement during spawning for "sticky" density dependent movement. This can however be set by selecting a flag feed_while_spawn=1

11.3. Migration into and out of the model domain

It is possible to allow a part or all of an age cohort or biomass pool to leave the model domain for a set period of time. This is used to imitate seasonal migrations to areas outside the model domain. This option is typically used for large mammals such as whales, which only spend a specific period of time within the model domain, but it can be applied to any group. For age-structured groups separate migration parameters are given for juveniles and adults, and one parameter is needed for biomass pools. Migration is handled by the *Ecology_Invert_Migration()* and E*cology_Total_Verts_And_Migration()* routines in **atmovement.c**

The migration routines follow these key steps:

1) Check if a species is migrating: flagXXXMigrate=1

2) Check if the current time is the time when the species migrates out of the model. The day of the year that a group migrates out of the model is given in XXX_Migrate_Time for biomass pools, and jXXX_Migrate_Time and XXX_Migrate_Time for age-structured groups. The number of times a group migrates out of (and back into) the system is given by flagXXXMigrate. The arrays must have this many entries.

3) Check if the species migrates all at once or gradually. This is set in k_migslow, where gradual migration is indicated by 1. If k_migslow=1 then the period over which species migrates out and back into the model domain must be given in XXX_Migrate_Period parameter. If k_migslow=1 XXX_Migrate_Period value must be >1 or else Atlantis will quit with an error message.

4) If it is a migration time, check what proportion of the age class migrates out of each box and take that number (biomass) out of the model. This value set in the vector MigIOBox_XXX (with

a value given per box in the model). Separate vectors are required juveniles and adults in age-structured groups. The amount removed from the model is stored in the MIGRATION array.

5) Check if the current time is the time when the species returns into the model. The day of the year that a group returns back into the model is given in XXX_Migrate_Return for biomass pools, jXXX_Migrate_Return and XXX_Migrate_Return for age-structured groups.

6) Check whether there was some mortality while outside the model domain. The survivorship of migrants (proportion of biomass or numbers returning back into the model domain) is given in XXX_FSM for biomass pools and jXXX_FSM and XXX_FSM for age structured groups. The remaining nitrogen (1-FSM) is lost from the model.

7) Check whether the species has grown while outside the model domain. This is set by the XXX_FSMG and jXXX_FSMG parameters, which indicates proportional increase in both SN and RN while outside the model domain. Growth outside the model domain adds nitrogen into the system.

8) Checks which stock the migrants return to. The specific stock the migrants return to is given in jXXX_ReturnStock and XXX_ReturnStock parameters. If the parameter value is set to 0, then returning migrants are distributed evenly across all stocks.

12. PROCESSES IN BACTERIAL AND INANIMATE POOLS

12.1. Bacterial processes

Bacteria can be represented explicitly or implicitly in Atlantis. Not all early ecosystem models included bacteria, and PPBIM did not explicitly model bacteria. Below is an excerpt of the PPBIM report justifying the decision not to include bacteria:

Reasons for excluding bacteria in PPBIM (from Muray and Parslow 1997)

"Heterotrophic bacteria have been included explicitly in a number of previous models, most notable Fasham et al. (1990). Bacteria in aquatic ecosystems play a number of important roles. The most important role is to attack, ingest and remineralise both particulate and dissolved organic matter, releasing dissolved inorganic nutrients. This role is represented implicitly in the model by assigning a breakdown rate to the detrital pools. Given the other uncertainties associated with the nature of detritus, and the lack of information on bacterial biomass, growth rates and loss rates in Port Phillip Bay, there is little reason to believe that an explicit model of bacterial dynamics would increase the accuracy of predicted detrital breakdown rates.

Bacteria can also serve as a food source for microzooplankton, and thereby divert organic matter which would otherwise be "lost" back to higher trophic levels. This alternative food source may have some impact on microzooplankton dynamics, and their interaction with small phytoplankton. The small phytoplankton – microzooplankton interaction is important under oligotrophic conditions, but much less important under the higher nutrient loadings of concern in Port Phillip

Bay. Moreover, owing to the inefficiency of the microbial loop, relatively little bacterial production finds it way to higher trophic levels (Ducklow 1991). Recent studies have suggested that a large fraction of water-column bacterial production is not grazed but is killed by viruses instead (Weinbauer and Peduzzi 1995) leading to a further reduction in utilisation of bacterial production by higher trophic levels (Murray and Eldridge 1994)."

Atlantis explicitly models aerobic bacteria in the water column and sediments, identified as pelagic bacteria (PB) and benthic bacteria (BB) respectively. This means that functional groups can feed on bacteria rather than on detritus. The representation of bacteria used in Atlantis does a much better job than old consumption-based representations, which treated bacteria as just another consumer. By making bacterial biomass dependent on the volume of detritus the bacterial processes are dynamically represented in a more rigorous way.

A precursor to Atlantis also included anaerobic bacteria. At present these are only implicitly represented in Atlantis, but a dynamic form will be added in a future release.

Key bacterial processes

The growth of PB and BB bacteria is dynamically dependent on the concentration of labile and refractory detritus (DL and DR), where DL is typically considered a primary food source. **Bacteria uptake both DL and DR and produce DR, DON and NH3**. Bacterial wastes do not contribute to DL but convert detritus into nutrients as part of the nitrification-denitrification and remineralisation processes.

Mortality of bacteria is caused by predation (grazing) by different consumers, as well as non-predation mortality due to oxygen limitation, acidification (optional), extra mortality (optional) and linear mortality

Pelagic bacteria processes are run by the routine *Pelagic_Bacteria_Processes()* in **atGroupProcess.c**

The growth of pelagic bacteria is modelled as

$$G_{BP} = mum \cdot B \cdot \left(1 - \frac{B}{XPB_{DL} \cdot DL + XPB_{DR} \cdot DR}\right)^{kBP}$$

where mum is the maximum growth rate (mum_PB), XPB_{DL} and XPB_{DR} is the maximum ratio of the bacteria to DL and DR biomass (XPB_DL and XPB_DR parameters), *B* is the biomass of the pelagic bacteria, *DL* and *DR* is the biomass of labile and refractory detritus and k_PB is the constant determining the exponent of growth (1- linear, 3-drop off before maximum, 5- drop off at maximum). The growth of sediment bacteria (BB) is modelled in the same way, except that mum_BB and k_BB parameters are used.

There is an option to allow for stimulation of bacterial growth by bioirrigation and bioturbation. This option is turned setting flagbactstim=1 and will enhance growth depending on the biomass of infaunal organisms and porosity of the sediment (see chapter 4).

Mortality (non-predation) of pelagic and sediment bacteria is modelled as

$$M_{BP} = \left((mL + (1 - \delta_{O2}) \cdot mO + mA + mS) \cdot B_{BP}\right) \cdot mortsc$$

where *mL* is linear mortality, *mA* is the acidification mortality, *mS* is the extra mortality due to acidification effects, *mO* is the oxygen limitation mortality, where (1-δ_{O2}) is the oxygen limitation scalar, B_{PB} is the biomass of pelagic bacteria (or BB is sediment bacteria are modelled) and *mortsc* is an forced external mortality scalar.

The uptake of detritus by pelagic bacteria is the modelled as

$$\frac{d(DL)}{dt} = \frac{Gr_{BP} \cdot PB_{DL}}{B_{BP} \cdot E_{DL,BP}}$$

where $E_{DL, PB}$ is the assimilation efficiency on labile detritus (EDL_BP for pelagic bacteria and EDL_BB for benthic bacteria) and *PB_DL* is the current (box specific) partitioning of PB to DL (vs DR).

Uptake of refractory detritus DR is modelled in the same way except that PB_DR is used and assimilation efficiencies on refractory detritus is used (EDR_BP and EDR_BB).

12.2 Biogeochemical processes

Details on the fluxes in the inanimate pools have been described in Fulton et al. 2004a and more recently in Link et al. 2011 Atlantis-NEUS model report. The table of inanimate pools tracked in the water column and sediments is given in chapter 6.1.

13. INFLUENCE OF ENVIRONMENTAL FACTORS ON ECOLOGICAL PROCESSES

13.1. Temperature: two methods to get temperature scalar

Temperature can have a large influence on many biogeochemical and ecological processes, but in Atlantis the user has an option to turn the temperature effects off, by setting flagq10=0 (NOT recommended given the fundamental temperature dependence of most physiological processes). Temperature effects are mostly applied through the Tcorr scalar calculated in the *Get_Tcorr()* routine. Setting flagq10=0 means that Tcorr scalar will be set to 1.

There are two ways to calculate the Tcorr scalar and different methods can be applied for different species, selected with the q10_method_XXX parameter. Some details on the calculations are given here. If the q10_method_XXX=0 then a simple Q10 based correction scalar is calculated, using the reference temperature of 15C:

$$T_{scalar} = q10^{\frac{(T_{H2O}-15)}{10}}$$

where q10 is a species-specific parameter provided in the q10_XXX and is typically set to 2. The Figure 13.1 shows the effect of different q10_XXX values.

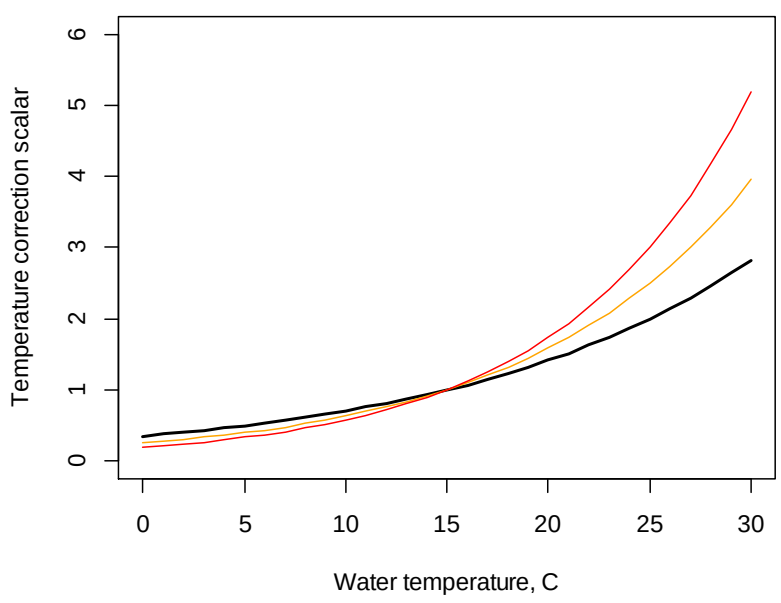


Figure 13.1. The T_{scalar} values at different water temperatures.

Black: q10=2
Orange: q10=2.5
Red: q10=3

A more complicated six parameter function is used if q10_method_XXX=1 (based on Gary Griffiths PhD thesis)

$$Tcorr = \log 2 \cdot \phi_A \cdot Cons_B^{Temp} \cdot \exp\left(\phi_C \cdot \frac{\left| Temp - Temp_{OPT} \right|^{Cons}}{\phi_{corr}} \right)$$

where $\emptyset_A$ is a species-specific coefficient (temp_coefftA_XXX), $Cons_B$ is the global coefficient (temp_coeffB), *Temp* is ambient water temperature, $Temp_{OPT}$ is a species-specific optimum temperature (q10_optimal_temp_XXX), $\emptyset_C$ is the global coefficient (temp_coeffC), *Cons* is a global exponent parameter (temp_exp), and $\emptyset_{corr}$ is a species q10 correction parameter (q10_correction_XXX).

This function aims to imitate a humped response, where rates are highest at optimum temperature levels and decrease when the temperature is below or above the optimum. The function is very sensitive to changes in parameters and before applying it the users should carefully explore the shape

of the function for the chosen parameter values (Figure 13.2). The black line shows the shape of the response curve for the original parameter values, except for $Temp_{OPT}$ which was chosen to be 19C

$Ø_A$ (temp_coefftA_XXX) =0.85
$Cons_B$ (temp_coeffB) = 1.06
$Temp_{OPT}$ (q10_optimal_temp_XXX) = 19C
$Ø_C$ (temp_coeffC) = 1
Cons (temp_exp) = 3
$Ø_{corr}$ (q10_correction_XXX) = 1000

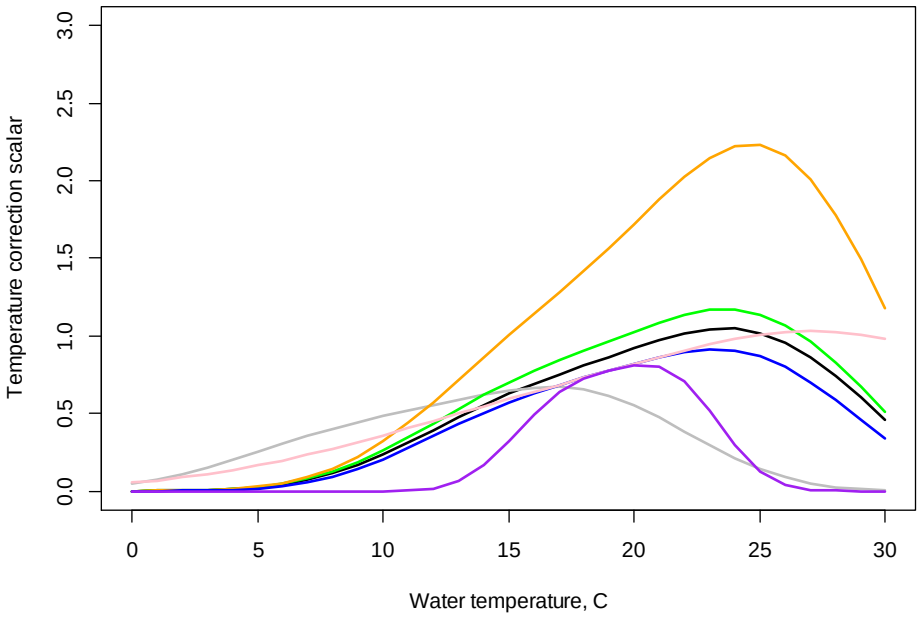


Figure 13.2. A humped temperature response used when q10_method_XXX=1.

Black: original values as shown in parentheses below
Grey: $Temp_{OPT}$ = 12C (19C)
Green: $Ø_A$= 0.95 (0.85)
Orange: $Cons_B$ = 1.1 (1.06)
Blue: $Ø_C$ = 1.1 (1.0)
Pink: Cons = 2.5 (3.0)
Purple: $Ø_{corr}$ = 100 (1000)

13.2. Temperature: effects on feeding parameters and assimilation efficiency

Once the Tcorr scalar has been calculated for the species and its ambient temperature in the cell using one of the two methods above, it is applied as a scalar to a range of processes. Typically, parameters that are scaled by Tcorr are indicated with T15, but this is not always the case.

A) Primary producers

In all primary producers the Tcorr scalar is applied to the light saturation (KI_XXX_T15) and maximum growth rate parameter (mum_XXX_T15)

B) Consumer feeding parameters

For ***biomass pools*** the C_ and mum_ values in the biological parameter file are given as rates at 15C and they are **always** scaled (multiplied) by Tcorr scalar for a given water temperature in a cell.

For **age structured groups** the search volume (vl_a) is always scaled by Tcorr, but the temperature scaling of C_ and mum_ is applied **only** if flagtempsensitiveXXX is set to 1. This is an inherited convention form the models the approaches were taken from and may change in the future.

C) Consumer assimilation efficiency – optional

Atlantis has an option to set improved or decreased assimilation efficiency depending on the temperature. There is no clear consensus in the ecological community on how temperature affects assimilation efficiency (it appears to be taxa specific); hence the user can decide whether to use this option.

To allow for temperature effect on assimilation efficiency set flagq10effXXX to 1

When **flagq10effXXX = 1** the **efficiency is poorer in cooler water** and the Tcorr scalar (Fig. 13.1) is applied **only** if water temperature is lower than the optimum or reference level (depending on the q10_method_XXX used).

If **flagq10effXXX= 2** the **efficiency is poorer in warmer water** and the Tcorr scalar (see Fig. 13.1) is applied **only** if water temperature is higher than the optimum or reference level (depending on the q10_method_XXX used).

In the calculations above if the Tcorr is >1 then the Tcorr=1/T_scalar. This means that Tcorr is always <1, which ensures that efficiency is decreased when correcting for temperature effects.

D) Mortality

For all species the linear mortality (mL), quadrating mortality (mQ) and extra mortality (mS) values are scaled by Tcorr.

E) Physical parameters

Parameters that determine the rate of breakdown are all scaled by the Tcorr scalar. They include:

r_DL_T15 – rate of labile detritus breakdown (day^{-1})
r_DC_T15 – rate of carrion breakdown (day^{-1})
r_DR_T15 – rate of refractory detritus breakdown (day^{-1})
r_DON_T15 – rate of dissolved organic nitrogen breakdown (day^{-1})
r_DSi_T15 – rate of detrital silica breakdown (day^{-1})
K_nit_T15 – rate of nitrification by free bacteria ($mgN\ day^{-1}$)

13.3. Salinity

The **salinity** effects on **biomass pool and age structured group** physiological processes are modelled through an optional Scorr, designed to reflect the sensitivity of physiological processes to

salinity conditions. The Scorr scalar is not calculated dynamically, but supplied by the user in salt_correction_XXX parameter. The Scorr scalar is applied only if:

1) an organism is identified as sensitive to salinity, with flagSaltSensitive_XXX
2) an organism an outside the salinity limits defined with XXX_min_salt and XXX_max_salt

The Scorr scalar is applied in the same way as for the Tcorr scalar described above. For age structured groups the Scorr cannot be applied alone without applying the Tcorr scalar. This means that if a species is identified as sensitive to salinity, but NOT sensitive to temperature, the Scorr scalar will not be applied to the physiological processes that have optional temperature scaling. This is not the case for biomass pool groups, where Scorr is applied regardless.

13.4. Acidification

Atlantis has an option to include effects of acidification on different physiological processes, predatory interactions and non-predation mortality. The pH effects are activated by setting flagmodelpH to 1 and flagpHsensitive_XXX to 1.

The calculation of the pH correction scalar (pHCorr) is described in detail on wiki here.
Briefly, depending on the pH_sensitivity_model selected, Atlantis will calculate the pHCorr using monodynamic, non-linear, linear or piecewise approach (see link above for details). The scalar will be <1 at decreasing pH values.

As for the Scorr scalar, **for age structured groups** the pHCorr will be applied to the processes affected by temperature (feeding rates, assimilation efficiency, mortality), **only if a species is sensitive to pH** (flagpHsensitive_XXX=1) **and sensitive to temperature.**

The pH can also affect other processes, as listed in Table 13.1

Table 13.1. Effects of pH on physiological and ecological processes. See detailed description here

What is affected	How to activate	How is it applied
Processes for which pHcorr scalar is applied at the same time as the Tcorr scalar, described in chapter 13.2		
Growth and non-predation mortality rates	flagpHsensitive_XXX=1 For age structured groups it is applied only if: flagtempsensitiveXXX=1	Growth and non-predation mortality are affected in opposite ways by pH. The unmodified C and mum will be multiplied by pHCorr (and decrease as a result) and the unmodified mL and mQ multiplied by 1.0/pHCorr (and increase)
Search volume (if predcase=5)	flagpHsensitive_XXX=1	The vla_T15 is multiplied by pHCorr (and decreases) This is only applied for age structured groups

Assimilation efficiency	flagpHsensitive_XXX=1 flagtempsensitiveXXX=1 flagq10eff_XXX=1 or 2	Four assimilation efficiencies are multiplied by pHCorr (and decreases)

Processes for which pHcorr scalar is applied differently from Tcorr scalar

Availability of prey to predators	flagpHsensitive_XXX=1 flagpredavaileffect_XXX = 1	For **biomass pool prey** availability to predators (defined in pPREY or ontogenetic diet matrices) is **increased** multiplying by 1.0/pHCorr For **age structured group prey** availability to predators (defined in pPREY or ontogenetic diet matrices) is **decreased** multiplying by pHCorr
Nutritional content of a species to its predators; mostly intended to simulate nutritional content of primary producers	flagpHsensitive_XXX=1 flagnutvaleffect_XXX=1	The amount of prey biomass available to a predator (defined in pPREY or ontogenetic diet matrices) is further multiplied by pHCorr (ot represent that more must be eaten to get the same nutritional content) and in this way decreased, reflecting lower nutritional content
Reduced larval survival before recruitment	flagpHsensitive_XXX=1 flagfecundsensitive_XXX =1	The number of recruits is multiplied by pHCorr and therefore decreased
Modifying thermal tolerance of a species	flagpHsensitive_XXX=1 flagcontract_tol_XXX = 1	The thermal tolerance decreases according to contract_tol_XXX parameter which defined the number of degrees to contract the temperature tolerances by as pH drops
Additional mortality	flagpHsensitive_XXX=1 flagPHmortcase >0	Extra mortality applied for ALL groups
Extra mortality	flagpHsensitive_XXX=1 pHmortstart = 1	Another logistic extra mortality term is added, see details here

13.5. Oxygen

The oxygen dependency is modelled differently from the temperature, salinity and pH effects described above. The oxygen content of the water does not affect the physiological processes, but recruitment and distributions. It also can lead to oxygen stress induced mortality of biomass pool groups.

The effect of oxygen limitation on feeding rates and linear mortality of biomass pool groups is non-optional and is modelled with O2case, mO_XXX, mD_XXX, KO2_XXX and KO2LIM_XXX parameters

The sensitivity of distributions and recruitment on oxygen concentration is optional. It is applied to both age structured groups and biomass pools and is activated by a global flagO2depend parameter

and species-specific minimum oxygen concentrations XXX_min_O2 parameter, setting the minimum tolerated oxygen level. If this option is used then species distribution will contract to areas above the minimum oxygen level. Further, the recruits that arrive into cells with oxygen concentrations lower than the minimum will be killed (not contracted!, see chapter 10.9.3.).

14. FINAL OVERVIEW OF ECOLOGY ROUTINES

The schematic representation below gives a brief overview of the key ecology routines and the order they are called. The Harvest and socio-economic routines are not included in this representation.

The *main()* routine controls the entire run, and calls the *runNextTimeStep()* if the run should continue (has not been terminated or the last time step has not been reached yet). The *runNextTimeStep()* checks the time of the year and initiaties any relevant annual routines, such as migration or spawning (they are executed later, but the initiation is started here). It then calls the *Ecology_Box_Biology()* routine to run all ecological processes in a given cell. The *Ecology_Box_Biology()* assesses the environmental conditions (light, oxygen, habitat, eddies) and calls for routines that should be executed in each of the five box types: **water column** (not in contact with the sediment layer), **sediment** (not in contact with the water column), **epibenthic** (bottom water layer and top sediment layer), **ice** and **land**. Within each of these boxes Atlantis first runs the main ecological processes, such as feeding, maintenance (if used), growth, nutrient cycling, waste production. These processes are run on an adaptive time step, which means that for the smallest groups (bacteria, phytoplankton) the processes might have to be dynamically repeated many times within a single model timestep (see introduction in Chapter 6 on adaptive time step). Once these processes are completed, Atlantis then runs the required annual ecology processes, such as reproduction, recruitment and ageing.

The land and ice boxes have only been added recently and are not discussed in details below (the execution of ice is quite similar to the water column, while the land is like the epibenthic layers).

Atlantis runs the water column, sediment or epibenthic processes depending on what kind of interactions are expected (epibenthic processes are run only in the epibenthic boxes). The water column, sediment and epibenthic processes are summarised below.

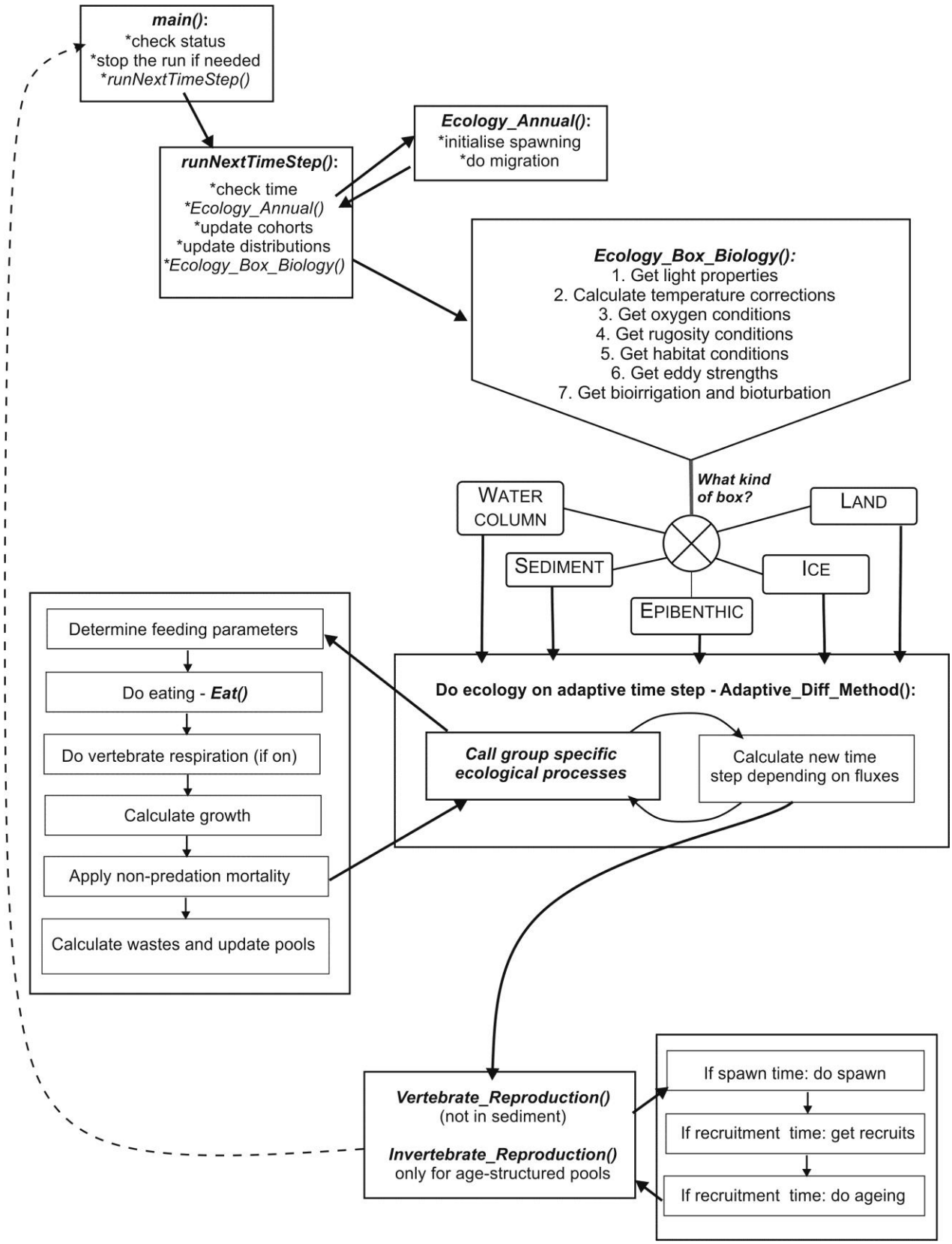

main():
*check status
*stop the run if needed
*runNextTimeStep()
runNextTimeStep():
*check time
*Ecology_Annual()
*update cohorts
*update distributions
*Ecology_Box_Biology()
Ecology_Annual():
*initialise spawning
*do migration
Ecology_Box_Biology():
1. Get light properties
2. Calculate temperature corrections
3. Get oxygen conditions
4. Get rugosity conditions
5. Get habitat conditions
6. Get eddy strengths
7. Get bioirrigation and bioturbation
What kind of box?
WATER COLUMN
LAND
SEDIMENT
ICE
EPIBENTHIC
Determine feeding parameters
Do eating - Eat()
Do vertebrate respiration (if on)
Calculate growth
Apply non-predation mortality
Calculate wastes and update pools
Do ecology on adaptive time step - Adaptive_Diff_Method():
Call group specific ecological processes
Calculate new time step depending on fluxes
Vertebrate_Reproduction()
(not in sediment)
Invertebrate_Reproduction()
only for age-structured pools
If spawn time: do spawn
If recruitment time: get recruits
If recruitment time: do ageing

Table 14.1. List of key ecological processes run in the three main distribution types: water colum, sediment, and epibenthic layer. List of the pools tracked in the three types is given in table 6.1.
PP – phytoplankton (including dinoflagelates), ZP – zooplankton, MA – macroalgae and seagrasses, BI – benthic invertebrates, PI – cephalopods and prawns (nektonic pelagic invertebrates), VE – vertebrates, MB - microphytobenthos

Process	**N**	**Si**	**PB**	**BB**	**DL**	**DR**	**PP**	**ZP**	**MA**	**BI**	**PI**	**VE**	**MB**
Water column processes													
Used by phytoplankton	+	+											
Used by bacteria	+				+	+							
Input from wastes	+	+			+	+							
Mineralisation			+		+	+							
Nitrification	+		+										
Oxygen production							+						
Nutrient uptake							+						
Lysis (nutrient stress)							+						
Feeding			+					+			+	+	
Growth			+				+	+			+	+	
Respiration												(+)	
Predation mortality/losses			+		+	+	+	+			+	+	
Other mortality			+				+	+			+	+	
Waste production			+				+	+			+	+	
Sediment processes													
Process	**N**	**Si**	**PB**	**BB**	**DL**	**DR**	**PP**	**ZP**	**MA**	**BI**	**PI**	**VE**	**MB**
Used by microphytobenthos	+	+											
Used by bacteria	+												
Input from wastes	+	+			+	+							
Mineralisation				+	+	+							
Nitrification	+			+									
Denitrification	+			+									
Oxygen production													+
Nutrient uptake													+
Feeding				+						+			
Growth				+						+			+
Predation mortality/losses				+	+	+				+			+
Other mortality				+			+			+			+
Waste production				+						+			
Bio turbation/irrigation										+			

Epibenthic processes													
Process	**N**	**Si**	**PB**	**BB**	**DL**	**DR**	**PP**	**ZP**	**MA**	**BI**	**PI**	**VE**	**MB**
Used by macrophytes	+												
Input from wastes					+	+							
Mineralisation					+	+							
Oxygen production									+				
Nutrient uptake									+				
Feeding										+			
Growth									+	+			
Predation mortality/losses					+	+			+	+			
Other mortality									+	+			
Waste production										+			
Bio turbation/irrigation										+			

Table 14.2. Basic assumptions and formulations in the Atlantis model (modified after Fulton et al. 2004b).

Feature	Assumptions and/or formulation notes
General features	
biomass units	mg N/m3
input forcing	nutrients, temperature and physics on interannual, seasonal, tidal frequencies
level of group detail	functional group (with a small number of individual species)
resolution of the formulation used for the biomass pool groups	follow the dynamics of the entire biomass pool of the functional group (or species) in the cell
resolution of the formulation used for the age structured group groups	follow the biomass dynamics (structural and reserve weight) of the 'average individual' for the functional group (or species) in the cell and the number of individuals in the cell
timestep	adaptive* daily or diurnal time step
Process related	
bioturbation and bioirrigation	yes, simple exchange between layers
consumption formulation	type II (asymptotic), with an availability parameter which can be habitat dependent
equations	five general sets of rate of change equations used (autrophs, biomass pool and age structured consumer, bacteria, inanimate)
formulation detail	general: only growth, mortality and excretion explicit
light limitation	optimal irradiance fixed
mixotrophy	yes, for dinoflagellates (if present)

nutrient limitation	external nutrients determine uptake
nutrient ratio	Redfield
oxygen limitation	yes
sediment burial	very low background rate included
sediment chemistry	dynamic, with sediment bacteria
shading of primary producers	yes
spatial (or habitat) limitation	yes (for benthic or demersal groups (and species))
spatial structure	flexible with the potential for multiple vertical and horizontal cells
temperature dependency	yes
transport model used for hydrodynamics flows	yes
Model closure	
top predators represented by static loss terms	some top predators are included explicitly, but predators not explicitly included in the foodweb are represented using quadratic mortality terms
mortality terms	linear and quadratic
Vertebrate and fisheries related	
age structure	multiple age classes (or stages, which equate to life phases), with final age class of each group a "plus group"
fishery discards	target and bycatch groups (and species)
incidental mortality due to fishing	yes
biomass pool fisheries	yes
management	variable, may be via effort limitations, gear limitations, minimum legal size, area or temporal closures and may be based on target or endangered stocks
stock-recruit relationship	Beverton-Holt, productivity-based or constant recruitment
stock structure	depends on recruitment function chosen – may be internal (all the stock within the bay and self-seeds) or external (the reproductive stock outside the bay produces the recruits and the oldest age classes migrate out of the bay to join this stock)

REFERENCES

Abrams, Peter A. "The Fallacies of" Ratio-Dependent" Predation." Ecology 75, no. 6 (1994): 1842-1850.

Ainsworth, C.H. and Walters, C.J. (2015) Ten common mistakes made in Ecopath with Ecosim modelling. Ecological Modelling 308, 14–17.

Ainsworth, C.H., Morzaria-Luna, H., Kaplan, I.C., Levin, P.S., Fulton, E.A., Cudney-Bueno, R., Turk-Boyer, P., Torre, J., Danemann, G.D., and Pfister, T. 2012a. Effective ecosystem-based management must encourage regulatory compliance: A Gulf of California case study. Mar. Policy 36(6): 1275–1283.

Ainsworth, C.H., Morzaria-Luna, H.N., Kaplan, I.C., Levin, P.S., and Fulton, E.A. 2012b. Full compliance with harvest regulations yields ecological benefits: Northern Gulf of California case study. J. Appl. Ecol. 49(1): 63–72.

Ainsworth, C.H., Schirripa, M.J. and Morzaria Luna, H. (2015) An Atlantis Ecosystem Model for the Gulf of Mexico supporting Integrated Ecosystem Assessment. NOAA Technical Memorandum NMFS-SEFSC-676. Miami, USA.Available from http://dx.doi.org/10.7289/V5X63JVH [accessed 21 December 2015].

Alin, S.R., Feely, R.A., Dickson, A., Hernandez-Ayon, J.M., Juranek, L.W., Ohman, M.D. and Goericke, R. (in revision) Robust empirical relationships for estimating the carbonate system in the southern California Current System using hydrographic data and application to CalCOFI hydrographic cruise data (2005–2011). Journal of Geophysical Research.

Arditi, R., & Ginzburg, L. R. (1989). Coupling in predator-prey dynamics: ratio-dependence. Journal of theoretical biology, 139(3), 311-326.

Audzijonyte, A., Kuparinen A., Gorton R., Fulton E.A. (2013) Ecological consequences of body size decline in harvested fish species: positive feedback loops in trophic interactions amplify human impact. Biology Letters, 9(2):1103

Audzijonyte, A., Kuparinen, A., Fulton, E.A. (2014) Ecosystem effects of contemporary life-history changes are comparable to those of fishing. Marine Ecology Progress Series 495: 219-231

Aydin, K., Gaichas, S., Ortiz, I., Kinzey, D. and Friday, N. (2007) A Comparison of the Bering Sea, Gulf of Alaska, and Aleutian Islands Large Marine Ecosystems Through Food Web Modeling.

Behrenfeld, M.J. and Falkowski, P.G. (1997a) A consumer's guide to phytoplankton primary productivity models. Limnology and Oceanography 42, 1479–1491.

Behrenfeld, M.J. and Falkowski, P.G. (1997b) Photosynthetic rates derived from satellite-based chlorophyll concentration. Limnology and oceanography 42, 1–20.

Brand, E.J., Kaplan, I.C., Harvey, C.J., Levin, P.S., Fulton, E.A., Hermann, A.J. and Field, J.C. (2007) A Spatially Explicit Ecosystem Model of the California Current's Food Web and Oceanography. NOAA Technical Memorandum NMFS-NWFSC-84.

Brassington, G.G., Pugh, T., Spillman, C., Schulz, E., Beggs, H., Schiller, A. and Oke, P.R. (2007) BLUElink> Development of operational oceanography and servicing in Australia. Journal of Research and Practice in Information Technology 39, 151–164.

Chassignet, E.P., Hurlburt, H.E., Smedstad, O.M., Halliwell, G.R., Hogan, P.J., Wallcraft, A.J. and Bleck, R. (2006) Ocean prediction with the hybrid coordinate ocean model (HYCOM). Springer.

Christensen, V., and Walters, C.J. 2004. Ecopath with Ecosim: methods, capabilities and limitations. Ecol. Model. 172(2-4): 109–139.

Fiechter, J., Rose, K.A., Curchitser, E.N. and Hedstrom, K.S. (2014) The role of environmental controls in determining sardine and anchovy population cycles in the California Current: Analysis of an end-to-end model. Progress in Oceanography.

Fulton, E.A., M. Fuller, A.D.M. Smith, and A.E. Punt. 2004a. Ecological Indicators of the Ecosystem Effects of Fishing: Final Report. Australian Fisheries Management Authority Report, R99/1546.

Fulton, E.A., A.D.M. Smith and C.R. Johnson. 2004b. Effects of spatial resolution on the performance and interpretation of marine ecosystem models. Ecological Modelling 176: 27 – 42.

Fulton E.A,. J.S. Parslow, A.D.M. Smith and C.R. Johnson. 2004c. Biogeochemical Marine Ecosystem Models II: The Effect of Physiological Detail on Model Performance. Ecological Modelling, 173: 371-406.

Fulton, E., Smith, A. and Punt, A. (2005) Which ecological indicators can robustly detect effects of fishing? ICES Journal of Marine Science 62, 540–551.

Fulton, E.A., Smith, A.D.M. and Smith, D.C. (2007) Alternative management strategies for southeast Australian Commonwealth Fisheries: stage 2: quantitative management strategy evaluation. Australian Fisheries Management Authority Report.

Fulton, E.A., Link, J.S., Kaplan, I.C., et al. (2011) Lessons in modelling and management of marine ecosystems: the Atlantis experience. Fish and Fisheries 12, 171–188.

Fulton, E.A., Smith, A.D.M., Smith, D.C. and Johnson, P. (2014) An Integrated Approach Is Needed for Ecosystem Based Fisheries Management: Insights from Ecosystem-Level Management Strategy Evaluation. PLoS ONE 9, e84242.

Gray, R., Fulton, E.A., Little, L.R., Scott, R., 2006. Operating Model Specification Within an Agent Based Framework. North West Shelf Joint Environmental Management Study Technical Report, vol 16. CSIRO, Hobart, Tasmania. 127pp.

Griffith, G.P., Fulton, E.A. and Richardson, A.J. (2011) Effects of fishing and acidification-related benthic mortality on the southeast Australian marine ecosystem. Global Change Biology 17, 3058–307.

Griffith, G.P., Fulton, E.A., Gorton, R., and Richardson, A.J. 2012. Predicting Interactions among Fishing, Ocean Warming, and Ocean Acidification in a Marine System with Whole-Ecosystem Models. Conserv. Biol. 26(6): 1145–1152.

Haidvogel, D., Arango, H., Budgell, W., et al. (2008) Ocean forecasting in terrain-following coordinates: Formulation and skill assessment of the Regional Ocean Modeling System. Journal of Computational Physics 227, 3595–3624.

Horne, P.J., Kaplan, I.C., Marshall, K.N., Levin, P.S., Harvey, C.J., Hermann, A.J., and Fulton, E.A. 2010. Design and parameterization of a spatially explicit ecosystem model of the central California Current.

Hunsicker, Mary E., et al. "Functional responses and scaling in predator–prey interactions of marine fishes: contemporary issues and emerging concepts." Ecology Letters 14.12 (2011): 1288-1299.

Ihde, T. F., I. C. Kaplan, E. A. Fulton, I. A. Gray, M. Hasan, D. Bruce, W. Slacum, and H. M. Townsend. 2016. Design and parameterization of the Chesapeake Bay Atlantis Model: A spatially explicit end-to-end ecosystem model. U.S. Dept. of Commer., NOAA. NOAA Technical Memorandum NMFS-F/SPO-166, 145 p.

Johnson, P., Fulton, E., Smith, D. C., Jenkins, G. P., & Barrett, N. (2011). The use of telescoping spatial scales to capture inshore to slope dynamics in marine ecosystem modeling. Natural Resource Modeling, 24(3), 335-364.

Juranek, L.W., Feely, R.A., Peterson, W.T., et al. (2009) A novel method for determination of aragonite saturation state on the continental shelf of central Oregon using multi-parameter relationships with hydrographic data. Geophysical Research Letters 36.

Kaplan, I.C., Levin, P.S., Burden, M. and Fulton, E.A. (2010) Fishing catch shares in the face of global change: a framework for integrating cumulative impacts and single species management. Canadian Journal of Fisheries and Aquatic Sciences 67, 1968–1982.

Kaplan, I.C., Horne, P.J., and Levin, P.S. 2012. Screening California Current Fishery Management Scenarios using the Atlantis End-to-End Ecosystem Model. Prog. Oceanogr. 102: 5–18.

Kaplan, I.C., Holland, D.S., and Fulton, E.A. 2014. Finding the accelerator and brake in an individual quota fishery: linking ecology, economics, and fleet dynamics of US West Coast trawl fisheries. ICES J. Mar. Sci. J. Cons. 71(2): 308–319.

Kroeker, K.J., Kordas, R.L., Crim, R., et al. (2013) Impacts of ocean acidification on marine organisms: quantifying sensitivities and interaction with warming. Global change biology.

Lehodey, P., Senina, I., and Murtugudde, R. 2008. A spatial ecosystem and populations dynamics model (SEAPODYM) – Modeling of tuna and tuna-like populations. Prog. Oceanogr. 78(4): 304–318. doi:10.1016/j.pocean.2008.06.004.

Link, J.S. (2010) Adding rigor to ecological network models by evaluating a set of pre-balance diagnostics: a plea for PREBAL. Ecological Modelling 221, 1580–1591.

Link, J.S., Fulton, E.A. and Gamble, R.J. (2010) The northeast US application of ATLANTIS: A full system model exploring marine ecosystem dynamics in a living marine resource management context. Progress In Oceanography 87, 214–234.

Methot Jr., R.D., and Wetzel, C.R. 2013. Stock synthesis: A biological and statistical framework for fish stock assessment and fishery management. Fish. Res. 142: 86–99. doi:10.1016/j.fishres.2012.10.012.

Morato, T., E.A. Fulton and T. Pitcher. In press. Approaches to the modelling of seamounts and their fisheries. In, Seamounts, Ecology, Fisheries and Conservation. Fish and Aquatic Resources Series (FAR), Blackwell Publishing, Oxford.

Nihoul, J.C.J. and S. Djenidi, 1998. Chapter 18: Coupled physical, chemical and biological models. In: K.H. Brink and A.R. Robinson (eds), The Sea. John Wiley & Sons Inc: New York, London.

Persson, L., Van Leeuwen, A., & De Roos, A. M. (2014). The ecological foundation for ecosystem-based management of fisheries: mechanistic linkages between the individual-, population-, and community-level dynamics. ICES Journal of Marine Science: Journal du Conseil, fst231.

Rose, K., Allen, J.I., Artioli, Y., et al. (2010) End-To-End Models for the Analysis of Marine Ecosystems: Challenges, Issues, and Next Steps. Marine and Coastal Fisheries: Dynamics, Management, and Ecosystem Science 2, 115–130.

Rose, K.A., Fiechter, J., Curchitser, E.N., et al. (2015) Demonstration of a fully-coupled end-to-end model for small pelagic fish using sardine and anchovy in the California Current. Progress in Oceanography.

Sainsbury, K.J., A.E. Punt A.E. and A.D.M. Smith. 2000. Design of operational management strategies for achieving fishery ecosystem objectives. ICES Journal of Marine Science, 57: 731–741

Savina, M., Condie, S.A., and Fulton, E.A. 2013. The role of pre-existing disturbances in the effect of marine reserves on coastal ecosystems: A modelling approach. PloS One 8(4): e61207.

Shchepetkin, A.F. and McWilliams, J.C. (2005) The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. Ocean Modelling 9, 347–404.

Shin, Y.-J., and Cury, P. 2004. Using an individual-based model of fish assemblages to study the response of size spectra to changes in fishing. Can. J. Fish. Aquat. Sci. 61(3): 414–431. doi:10.1139/f03-154.

Smith, D.C., Johnson, P. and Fulton, E.A. (2010) Developing integrated performance measures for spatial management of marine systems. Fisheries Research and Development Corporation Report

Uitz, J., Huot, Y., Bruyant, F., Babin, M. and Claustre, H. (2008) Relating phytoplankton photophysiological properties to community structure on large scales. Limnology and Oceanography 53, 614–630.

Uitz, J., Claustre, H., Gentili, B. and Stramski, D. (2010) Phytoplankton class-specific primary production in the world's oceans: Seasonal and interannual variability from satellite observations: PHYTOPLANKTON CLASS-SPECIFIC PRODUCTION. Global Biogeochemical Cycles 24, n/a–n/a.

Walters, C., Pauly, D., & Christensen, V. (1999). Ecospace: prediction of mesoscale spatial patterns in trophic relationships of exploited ecosystems, with emphasis on the impacts of marine protected areas. Ecosystems, 2(6), 539-554.

Walters, Carl, and James F. Kitchell. "Cultivation/depensation effects on juvenile survival and recruitment: implications for the theory of fishing." Canadian Journal of Fisheries and Aquatic Sciences 58, no. 1 (2001): 39-50.

Weijerman, M., Fulton, E.A., Kaplan, I.C., Gorton, R., Leemans, R., Mooij, W.M., and Brainard, R.E. 2015. An integrated coral reef ecosystem model to support resource management under a changing climate. PloS One 10(12): e0144165.

Wittmann, A.C. and Pörtner, H.-O. (2013) Sensitivities of extant animal taxa to ocean acidification. Nature Climate Change 3, 995–1001.

Appendix 1: TIPS FOR CALIBRATING BIOLOGICAL MODEL: NOTES FROM THE 2015 ATLANTIS SUMMIT

by Isaac Kaplan and Atlantis community

New Atlantis modelers at the Summit requested additional guidance on the steps useful for calibrating models for new systems.

The main software tools used to calibrate the model are OLIVE or DIVE (for visualising spatial output maps in out.nc), a good text editor to look through *log.txt*, Excel files that use lookup tables to compare model output with observations and assessments (as appropriate) or R (where scripts can be run to do spatial or time series plotting for comparison purposes).

At the Atlantis Summit, two useful R packages were unveiled to help with calibration:

- https://github.com/mareframe/vat
- Alex Keth's R scripts for Atlantis, which are being incorporated into https://github.com/r4atlantis/atlantisom
  Additional R packages are also being posted on Github:
- Github https://github.com/r4atlantis for R packages for Atlantis

Conversations about calibration are usually posted to the Atlantis google group (https://groups.google.com/forum/#!forum/atlantis-ecosystem-model), while solutions to problems are posted on the Atlantis wiki (https://confluence.csiro.au/display/Atlantis/Atlantis+Ecosystem+Model+Home+Page).

The general goals of calibration are, in this order:

1. Prevent all species present in the model from going extinct (unless they do so in the time series being fit to)
2. Have age structured groups grow such that size-at-age is reasonable (within 20% of initial conditions, typically)
3. For species with historical data are available, have the model recreate observations of abundance from surveys or assessments
4. For species with no historical data, the model should yield reasonable time series of abundance (especially under perturbation – such as environmental forcing or fishing pressure)
5. Capturing observed spatial distributions

Practical experience suggests that tackling model problems in the order above is best, since this sequence minimizes recalibrating bits too many times.

Once size-at-age is within 20% of the von Bertalanffy curve expected from the literature, then growth is generally acceptable. When calibrating age structured groups, attempt to get relative SN and RN values reasonable before worrying about numbers (den results). Note though that if numbers are too high (by an order of magnitude or more) the group maybe starving itself so you may need to reduce recruitment or increase predation to get numbers under control before you will make any headway with the growth of the group.

During calibration, tuning typically focuses primarily on changing growth rates, consumption rates, linear and quadratic mortality, and recruitment (Beverton Holt) parameters. These variables are in the *biology.prm* file. Below, XX represents the functional group code (e.g. FVD, FPL, etc.) from the functional_groups.csv file:

- **C_XX** consumption rates
- **mum_XX** maximum growth rates
- **mL_XX** 'unexplained' background linear mortality
- **mQ_XX** quadratic mortality ('unexplained' density dependence)
- **pPREYXX** diet or predator-prey interaction terms
- **BHalphaXX** Beverton Holt alpha parameters in recruitment

A thorough exploration of parameter distributions within the Atlantis models would be broadly useful for error checking inputs and understanding reasonable bounds during calibration. As suggested by Ainsworth and Walters (2015), other authors have already built databases and conducted meta-analyses on Ecopath parameters, and comparison of Atlantis values to Ecopath values would be informative. Finally, automated parameter comparison across models lends itself well to basic checks on model assumptions about biomasses and vital rates, akin to the PREBAL diagnostic test of Link (2010).

Calibration with fishing

A major part of calibrating the model is comparing how it performs when faced with various levels of fishing intensity, both real historical values and hypothetical values. This sort of qualitative exploration can reveal underlying problems, such as parameterizations that involve too little recruitment (and therefore inevitable decline of a stock) or too much recruitment (and therefore unreasonable resilience to fishing). Just as testing Ecopath models with fishing is a crucial step (Ainsworth and Walters 2015), Atlantis models can also use fishing to test basic parameterization of productivity. For heavily fished target species, we roughly expect Atlantis to behave like a single species model (and if not, need to consider why). For lightly fished, nontarget or forage species, we expect ecosystem dynamics to matter a lot more, and Atlantis may (reasonably) diverge from a single species assessment prediction.

Beth's suggestions on fishing scenarios to run are as follows (these in fact mirror some of the Common Scenarios runs for Atlantis Summit)

1. Unfished - Atlantis spinup estimate of the unexploited system (where possible from virgin, but also what system would it head to from current state if unfished)
2. Historical removals - can the Atlantis state support historically recorded catches (and suspected misreported catches); does the trajectory match assessment/survey data? (preferably survey data rather than assessment)
3. Fish the target(s) at current levels (and do so constantly through time)
4. Fish the target(s) at assessment estimate of RBC (Recommended Biological Catch, which may differ from MSY)
5. Fish the target(s) at 5xassessment estimate of RBC
6. Fish the target(s) at 0.2xassessment estimate of RBC
7. Fish the target(s) at 0.2xcurrent levels

8. Fish the target(s) at 20xcurrent levels (or more depending on current F)
9. Shock the system in some other way (e.g. nutrient pollution, habitat degration, etc.)
10. Combinations of shocks (typically high fishing + a non-fishing shock)

Tips for parameterizing biomass pools (invertebrates)

Benthic invertebrate abundance and distribution data are scarce, especially for non-harvested species, and generalized additive models are one approach to addressing this. Generalized additive models start from patchy observations or field sampling, and use habitat and oceanographic descriptors to predict invertebrate abundance or distribution. This was applied to SEAMAP data in the Gulf of Mexico (Ainsworth *et al.* 2015), and is being applied in the Bay of Biscay. Expert knowledge was also relied upon to parameterize these species.

Based on responses at the Atlantis Summit, invertebrate rate parameters such as consumption and growth rates are often taken from existing Ecopath models. These models serve as essential steps toward building Atlantis models, yet the assumptions and methods to estimate Ecopath parameters (PB and QB) were not clear to all Atlantis Summit participants. Furthermore, consumption and growth rates for invertebrates were often tuned heavily in Atlantis. A best practice would be to first understand Ecopath parameterization, then document how it informs Atlantis, before beginning the Atlantis calibration.

Invertebrate groups have been particularly hard to calibrate in many Atlantis models. It is not clear if this is due to poor data or noisy population dynamics. Cephalopods, shrimp, jellyfish, and some benthos are problematic. Cephalopods and shrimp often are parameterized with simple juvenile vs adult biomass pools, and users with the setup should view and test the associated code.

Most Atlantis modelers convert from carbon or dry weight to nitrogen using the Redfield ratio. However, greater attention to these conversions is warranted as the same value may not actually be appropriate across vertebrates and invertebrates of different kinds.

Tips for parameterizing age-structured (vertebrate) groups

Starvation and respiration

Parameterization of starvation is a topic that many Atlantis modelers have not focused upon, but the mortality rate for starving fish can be an influential parameter. Most participants at the Atlantis Summit felt that they should investigate higher values of starvation-induced mortality, as well as testing a model run where all diets are set to 0 (with starvation expected to result).

Very few Atlantis models currently include explicit respiration. In theory, respiration can be included implicitly by reducing assimilation efficiencies (from ~0.8 to 0.3 or less) (but see details on respiration in the manual above). Similarly, most models used a standard temperature effect on metabolism (Q10 = 2), but this should be questioned for new models and applications, particularly those related to climate change.

Movement

A mix of approaches to age structured group movement has been applied within Atlantis, and the exact approach needs to be tailored to the research question and model goals. Density dependent movement has been particularly useful for modeling marine mammals and birds in the Baltic Sea and Northeast USA (Link *et al.* 2010) and some fish groups in Guam (Weijerman *et al.* 2015). Temperature-dependent movements have less commonly been applied. Often density dependent

movement for some species has been combined with prescribed quarterly spatial distributions for other species; these prescribed movements can be based on expert opinion and rough categorization of species into a few general movement patterns.

Reproduction

To model recruitment (reproduction) of fish, most Atlantis models to date primarily rely on one of two forms of the Beverton-Holt relationship. This omits the possibility of ecological effects such as cannibalism on larvae or pre-recruits. When using Beverton-Holt, a starting point for parameterization is to use the spreadsheet available on the wiki, which converts from stock assessment quantities to Atlantis Beverton-Holt parameters. However, subsequent calibration is essential, in large part because 'recruits' in Atlantis are much younger than 'recruits' in stock assessments, and the intervening mortality means that stock assessment estimates must be scaled up to Atlantis numbers-of-recruits. In general, mammals and birds were often parameterized with fixed offspring per adult.

Growth, consumption, and predator-prey functional response

There are at least three methods for estimating consumption rates, prior to using these within Atlantis. For mammals and birds, Gompertz growth curves have often been used to specify size-at-age. For fish, size-at-age is often taken from von Bertlanffy growth relationships. Then one option is that consumption may be estimated from allometric relationships

$$C_i = a \cdot (RN_i + SN_i)^{0.7}$$

Where C is consumption , i indexes age, a is a constant, and RN and SN are weight-at-age in terms of reserve and structural nitrogen. This approach is used within the California Current Atlantis model.

A second alternative is that consumption may be estimated from growth:

$$C_i = b \cdot ((RN_i + SN_i) - (RN_{i-1} + SN_{i-1}))$$

where b is a conversion efficiency of ~10. This approach is presently used within the Icelandic Atlantis model.

A final alternative in ecosystems with better stomach sampling is that daily ration (consumption per day) can be directly taken from field studies. This approach is possible for the Baltic Sea Atlantis. Finally, it was noted that consumption estimates in Atlantis are maximum consumption, and the estimates above were therefore scaled up by 20%-300% to account for this.

A strong recommendation arose from Atlantis Summit conversations regarding predator-prey functional responses: Atlantis modelers should look at the code, understand the functional response chosen, and test alternative responses. Most models are using modified Holling Type II relationships for all or most of the age structured groups, though some species (such as menhaden and marine mammals in Northeast US) use modified Holling type III. The Holling Type II relationship in Atlantis was originally used for filter feeders, and assumptions need to be understood for age structured groups. An Excel functional response demo is available for common functional response.

All users were encouraged to carefully consider and test the functional response options they choose to use in the code. If the traditional modified Holling Type II relationship is chosen, the *mum* maximum growth rate parameter can usually be inferred from the equations listed above for consumption, and an assumption about assimilation or conversion efficiency. The 'clearance' ($C\_$) parameter would be related to water volume filtered by a filter feeding invertebrate; for a fish it is

roughly related to search volume. However, modelers in some systems have found that clearance= 0.1 * *mum* was a good starting point for calibration, while others have had better success with clearance = 10 x *mum*. Each modeler should test and plot the functional response and the parameters they are using, and consider alternative formulations as well.

During calibration of the model, most users experienced situations with low growth of age structured groups, and in these cases increasing *mum* was the recommended first step, before adjusting other functional response parameters.

Other mortality

In most Atlantis models, linear and quadratic mortality (*mL* and *mQ* in Atlantis) are generally used as calibration parameters of last resort, to represent mortality effects or carrying capacity not explicitly handled the by the model.

Tips on parameterizing biogeochemistry

Atlantis requires initial conditions for nutrients, primary producers, zooplankton, and detritus. Several modeling groups have used Nutrient-Phytoplantkon-Zooplankton-Detritus (NPZD) model output as the basis for these Atlantis initial conditions. These include the Baltic Sea and Strait of Sicily Atlantis models. This was suggested as a useful approach, as long as the NPZD models have been evaluated for skill against field observations. Some regions have extensive in-situ sampling and accessible databases, which can also be used as initial conditions for Atlantis. An example is http://gulfatlas.noaa.gov/ in the Gulf of Mexico.

Nutrient loading has been used in Atlantis models (forcing with nutrient time series), including the, California Current, Guam, Gulf of Mexico (Brand *et al.* 2007; Ainsworth *et al.* 2015; Weijerman *et al.* 2015) and Chesapeake Bay and Strait of Sicily model. One advantage of this approach is that it can be used as a proxy for multi-decadal trends in ocean productivity, even if full multi-decadal oceanography (ROMS etc.) is not available and must be 'looped'. Within the US, the Environmental Protection Agency has provided time series of point-source and non-point-source nutrients and sediments to Atlantis modelers.

Tips on handling ocean acidification

Several distinct approaches to handling ocean acidification impacts have been tested within Atlantis:

1) adding linear mortality to calcifying species, across all model polygons (Kaplan *et al.* 2010; Griffith *et al.* 2011).
2) allowing pH to evolve within Atlantis, but forced by atmospheric pCO2 (Weijerman *et al.* 2015),
3) Importing pH from a ROMS that includes biogeochemistry (California Current work in prep).

A different approach would be to predict pH or aragonite saturation state from temperature, oxygen, salinity, depth (Alin *et al.* in revision; Juranek *et al.* 2009). Based on experience with Common Scenarios at the Atantis Summit, we suggest the first option, before delving into the more detailed approaches.

After pH (or aragonite saturation state) fields are generated within or outside Atlantis, the biological response to pH must be parameterized. This is challenging and involves a high degree of uncertainty; global meta-analyses provide some guidance (Kroeker et al. 2013; Wittmann and Pörtner 2013) but

local conditions may require additional expertise. One concern was that these biological responses should be updated periodically as new information becomes available.