# TRACE and ODD Documentation for an Individual Based Model on Eastern Baltic Cod Ecology

Maria E. Pierce*, Thuenen Institute of Baltic Sea Fisheries

29th October 2021

## Revision History

| Revision | Date | Author(s) | Description |
|---|---|---|---|
| 0.9.0 | 1st October 2021 | ME Pierce | Added Rules and Experiments |
| 0.9.1 | 18th October 2021 | ME Pierce | Refined descriptions of rules |
| 0.9.2 | 29th October 2021 | ME Pierce | Feature complete |
| 0.9.3 | 29th March 2022 | ME Pierce | Added Information on Points 3,5,6 and 7 |

*maria.pierce@thuenen.de, Alter Hafen Süd 2, 18069 Rostock

# Contents

# 1 Problem formulation

This **TRACE** element provides supporting information on:
The decision-making context in which the model will be used; the types of model clients or stakeholders addressed; a precise specification of the question(s) that should be answered with the model, including a specification of necessary model outputs; and a statement of the domain of applicability of the model, including the extent of acceptable extrapolations.

The models documented here are meant to provide an example of a platform for knowledge integration of level- and domain-spanning knowledge of Eastern Baltic Cod (EBC) ecology. The Eastern Baltic Cod (EBC) stock is an example of a fish stock currently in a highly dynamic state, both in terms of biomass as well as its individuals' condition. After experiencing a high in biomass in the 1980s it has currently declined to a point where fisheries had to be closed in 2019 [1]. It is likely that efforts to aid in recovery of the stock will require deep knowledge of underlying mechanisms [2]. Therefore the models are meant to support the gathering and integration of information, reducing ambiguities, establishing gaps in knowledge and developing and refining hypothesis pointing at avenues of further investigation.

The models and simulation experiments are aimed at domain scientists researching the EBC. More specifically they are aimed at the different scientist investigating specific facets impacting EBC ecology and are meant to provide them with a focal point for communication and functional integration of their different perspectives. They thereby also provide an example of a transparent, unambiguous and executable platform for knowledge integration.

Rather than answer one specific research question, the models, submodels and experiments of this simulation provide a "thinking tool" to aid in the very vague and general question of what has driven the decline of the EBC and how its recovery can be supported. This will require refining and specifying open questions and identifying likely drivers of EBC ecology.

Simulation experiments conducted with the models should provide insight of the mechanisms and drivers under investigation at each respective step of knowledge integration (e.g. movement, reproduction, parasitation, etc.) and the interactions of these mechanisms, but at least information about physiological performance in terms of growth and condition.

The models, submodels and experiment of of this simulation study are meant to be applicable to the physiological performance of EBC. Interpretations and extrapolations are conditional on validation.

# 2 Model description - ODD protocol

---

This **TRACE** element provides supporting information on:
The model, i.e. a detailed written model description. For individual/agent-based and other simulation models, the ODD protocol is recommended as standard format. For complex submodels, include concise explanations of the underlying rationale. Model users should learn what the model is, how it works, and what guided its design.

---

In this section we will follow the ODD-protocol ([3], [4]) as suggested for individual/agent based models. How concise explanations of the various submodels are included please see 'some notes on this section':

**Some notes on this section:**
Ml-Rules due to its rule-based structure offers the opportunity to describe the exact formalisation and the underlying rationales for each process in detail. Following the ODD-protocol this rule-by-rule description will take place in the 'submodel' section. The following structure will be used: first the code will be given in ML-Rules syntax, then a quick overview will be given in the form of an info-box, providing the before and after state, the guards (conditions to be met for the execution of the rule) and the rate at which the rule is executed. Finally a natural language description will provide more detail, sources and, where applicable 'known further development requirements' will be elaborated upon.
Similarly the functions used in the models will be provided as they are used in the model will be introduced. First the function will be provided in ML-Rules syntax, then as an equation and where applicable sources will be provided and were considered useful figures providing visualisations of the functions will also be provided.
Experiments will also follow this structure, providing at least one example per conceptually similar experiments: here the experiment code will be provided in SESSL (scala) syntax, then an info-box containing the model used, the duration (simulated time), the parameters scanned (were applicable), the information scanned for (number of individuals and/or attributes), the interval at which they are observed and the number of replications is given and finally a thorough explanation of the experiment is provided.

## 2.1 Purpose

The purpose of the model/submodels is to provide a "thinking tool" for ecological hypothesis on Eastern Baltic Cod (EBC), particularly where these span several system levels and/or scientific (sub)disciplines.
The aim is explicitly **not** to provide quantitative answers but rather to allow to investigate the interdependencies of mechanisms, phenomena and dynamics.
For extensive description of the purpose of the model see parent publication "*Multi-faceted modelling for inclusive and transparent knowledge integration - a case study of the eastern Baltic cod.*" DOI: NA

## 2.2 Entities, state variables, and scales

The central entity of the model/submodels is the cod *Gadus morhua*. Although the aim is to investigate hypothesis about the Eastern Baltic Cod stock, the input data continues to be sourced from all information on cod. This needs to be taken into account for quantitative estimations

and assessments as physiological adaption of EBC are not accounted for.

#### GM - *Gadus morhua*

**Attributes** The COD is defined by the attributes listed in table 2.

**Sub-Entities** GM can have the following nested, sub-entities:
Stomach, Liver and Gonads

**Co-Entities** GM has the following same-level, co-entities:
Eggs (GME), Larvae (GML), Prey (Prey)

Table 2: GM attributes.

| Attribute | Use | Range | Quantity |
|---|---|---|---|
| **GM - *Gadus morhua*** | | | |
| length | GM length | 292-721 | [mm] |
| body mass | GM mass (excluding gonads) | 150 - 4500 | [g] |
| oxygen content | tracks oxygen | 0 - 30 | 0.1 [mg] $O_2$ |
| energy content | tracks metabolites | ? - ? | $\frac{2.7\ [J]}{0.2\ [mg]O_2}$ |
| sex | assign sex | 0,1 - female, male | |
| reproductive cycle | track reproductive cycle | 1,2,3,4 - spent, preparing, ready, spawning | |
| hunting success | control feeding | 0,1 | |
| stock affiliation | assign Eastern, Western, Hybrid | 'E','W','H' | |
| particular | force properties | as needed | |
| volume | track position | 1 - 48 | 1 |
| Identifier | track individuals | $-\infty$ - $+\infty$ | 1 |
| **GME - *Gadus morhua* Eggs** | | | |
| developmental stage | track developmental stage | 1 - 100 | 1 |
| stock affiliation | assign Eastern, Western, Hybrid | 'E','W','H' | |
| **GML - *Gadus morhua* Larvae** | | | |
| dry weight | track weight | NA | [mg] |
| stock affiliation | assign Eastern, Western, Hybrid | 'E','W','H' | |

The Stc is defined by the following attributes:
The Stc is defined by the following attributes:

## 2.3 Process overview and scheduling

ML-Rules is a Rule-based modelling language processes are described by rules and triggered according to rates scheduled by continuous-time-markov-chains (CTMCs) ???. consequently the

Table 3: Organ attributes:

| Attribute | Use | Range | Quantity |
|---|---|---|---|
| **Stc - Stomach** | | | |
| prey consumption | tracks consumed prey | 0 - max. stc. cont. | [g] |
| variability of food intake | allows to ? | 0-1 | 0.01 |
| identifier | track Stc and match with GM | $-\infty$ - $+\infty$ | 1 |
| **AJPO - Food Pieces** | | | 0.013 g |
| oxygen demand | set oxygen demand for digestion | | 0.1 [mg] $O_2$ |
| energy transfer | set energy transfer | 0-20 | 0.0001 |
| digestive factor | speed of digestion | $-\infty$ - $+\infty$ | 1 |
| **Liver - Liver** | | | |
| *Anisakis simplex* | track infection with parasite | | |
| *Contracaecum osculatum* | track infection with parasite | 0-20 | 0.0001 |
| identifier | track Liver and match with GM | $-\infty$ - $+\infty$ | 1 |
| **Gon - Gonads** | | | |
| sex | assign sex | 0,1 - female, male | |
| gametes | number gametes | | 1 |
| maturity | track maturity of gametes | 0-7 | 1 |
| energy | track energy allocation to development | | [g] |
| weight | track gonad weight | | [g] |
| identifier | track Gon and match with GM | $-\infty$ - $+\infty$ | 1 |

process overview and scheduling is covered by the Rule-by-Rule model description below.

## 2.4 Design concepts

The Domain Specific Language (DSL) used to formalise the model is ML-Rules. This allows for dynamic hierarchical nesting and has underlying time-continuous-markov chains allowing for stochastic races.

### 2.4.1 Basic principles

There is a core model, which is not meant to be static over further iterations of model development but should be validated with every iteration. This core model represents the oxygen and food dependent physiology of the central agent - the cod. It has to be able to consume oxygen and prey, grow, starve, starve to death and asphyxiate.
Building on the core model as many submodels as needed are and will be formalised such as:

Table 4: Environment attributes:

| Attribute | Use | Range | Quantity |
|---|---|---|---|
| | **Prey** | | |
| name | readability | | |
| size | define size | 0 - ? | 0.0013 [g] |
| oxygen demand | set oxygen demand for digestion | | 0.1 [mg] $O_2$ |
| energy transfer | set energy transfer | 0 - 20 | 0.0001 |
| digestive factor | speed of digestion | $-\infty$ - $+\infty$ | 1 |
| *Anisakis simplex* | track infection with parasite | | |
| *Contracaecum osculatum* | track infection with parasite | 0 - 20 | 0.0001 |
| | **Baltic Boxes** | | |
| number | track box, assigned name | 1-48 | 1 |
| temperature | provide temperature | 0 - 20 | 1 |
| oxygen | provide oxygen concentration | 0-100 | 1 [%] |
| salinity | provide salinity | | |
| day | provide photo-period | 1-365 | 1 |
| zone | ? | | |
| substrate | ? | | |
| prey availability | track prey availability | | |
| area | provide for calculations | | |
| minimum depth | | | |
| maximum depth | | | |

behaviour, reproduction, migration and parasitisation.
These can then be fused in a white-box manner [5].

## 2.5 Initialisation

The initial state of the model is defined by the so-called initial solution. With experiments input can be varied.
Over all the initial state of the model must be defined giving all entities and their exact attributes and relations to each other in terms of nesting.
For validation experiments the initial situation of the wet-lab experiment is reproduced as closely as possible.
When investigating hypothesis the initial state of the model is part of the hypothesis and will be formalised keeping this in mind.

## 2.6 Input data

External forcing takes place via EXACT-Rules. Data is taken from surveys or is part of hypothesis.
This can be varied to almost any extent.

## 2.7 Submodels

The intended purpose of the model relies intrinsically on the use of submodels. Several submodels are built for each concept, hypothesis or process. These are then combined in varying combinations to investigate interdependencies of the respective concepts, hypothesis and processes.
Each submodel is also complete in the sense that it has the purpose of formalising a hypothesis (or several) to be integrated while deepening the understanding of the ecological system supporting the EBC.

The **Rule by Rule** descriptions of the submodels will follow a set structure: first a representation of the model code will be given, here changes from the before to the after execution state will be signified by underlining the respective sections. Next a short "in a nutshell" description will be given, this is highlighted by a framing box. Finally a long-form explanation of the rule will be given in text form.

The description of the **functions** will follow directly after the descriptions of the rules they are used in (meaning if its not there, scroll upwards). These also follow a set structure: first a representation of the model code will be given, next a long form text description and finally a mathematical representation. in some cases a visual representation of the function will be added.

### 2.7.1 Rule by Rule: Basic Physiology

The Basic Physiology is the physiology extensively described in [6] updated to reflect a change in focus. The entities representing oxygen and available metabolites have been moved to become attributes of the cod. The order of the attributes has been changed and they have been extended to improve tracking of states and evaluation of experiments.

**Oxygen Intake**

```
//Oxygen Intake
GM(l,bm,ox  ,jpo,s,rc,hs,sta,p,v,ui)[gm?] ->
GM(l,bm,ox+1,jpo,s,rc,hs,sta,p,v,ui)[gm?]
@ if (ox < 30)
then amr(o,t) * lbmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Cod with oxygen content increased by 1 |
| Guard: | Cod oxygen content is lower than 30, threshold based on [7] and [8] |
| Rate: | At active metabolic rate (amr(o,t)) [9], o, t environmental oxygen and temperature respectively; scaled for time units and cod weight (lbmf(bm)) |

This rule formalises the intake of oxygen. The guard is derived from the combined findings of [7] and [8] for an upper threshold of cod oxygen content. The upper threshold avoids unrealistically high oxygen content and transfer of oxygen from favourable to depleted conditions by the individual. The rate for the process is based on experimental findings in wet-lab experiments conducted by [9]. The resulting active metabolic rate function (see below) makes this process sensitive to the environmental conditions of water oxygen saturation and temperature. The rate is normalised for body mass, time units and oxygen units with the linear body mass function (lbm - see below).

**active metabolic rate - function**

```
amr :: num -> num -> num;
amr oxy temp = (1-(e)^(0.34-0.035*oxy))*
               (30.01+17.29*(temp)^(1.062-0.015*temp));
```

The rate for oxygen uptake is based on the findings of Claireaux et al. [9], who determined the active metabolic rate of cod dependent on temperature and oxygen saturation of the surrounding water. The function for the active metabolic rate is given as:

$$\mathrm{AMR}(\mathrm{T}, \mathrm{S}_{\mathrm{O}_2}) = (17.29\mathrm{T}^{-0.015\mathrm{T}+1.062} + 30.01) \cdot (1 - \mathrm{e}^{-0.035\mathrm{S}_{\mathrm{O}_2}+0.34}) \, \frac{\mathrm{mg\ O_2}}{\mathrm{h\ kg}}, \qquad (1)$$

where T is the temperature in °C and $\mathrm{S}_{\mathrm{O}_2}$ is the oxygen saturation given in % (see Figure 1).

**linear body mass function - function**

```
lbmf :: num -> num;
lbmf bm = (bm/10000);
```

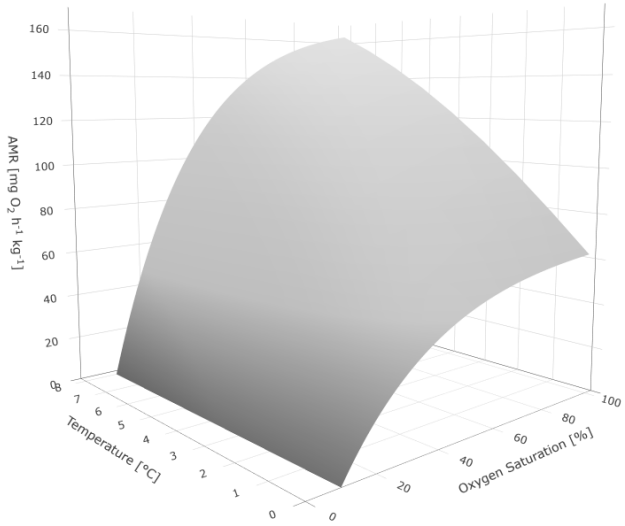



Figure 1: Active metabolic rate taken from [9].

This function scales rates linearly for body mass of individuals ('bm' given in [g]), maps the units of simulated time units TU (1h $\hat{=}$ 100TU) on to real world time and scales for oxygen units (1O $\hat{=}$ 0.1 mg $O_2$) to translate functions given in $[\frac{\text{mg } O_2}{\text{kg}\cdot\text{h}}]$:

$$\text{lwf}(bm) = 0.01 \cdot 0.001 \cdot bm \cdot 10 = \frac{bm}{10000} \tag{2}$$

where bm is the body mass in [g] (see Figure 2).

**Standard Metabolic Rate**

```
//Standard metabolic rate
GM(l,bm,ox  ,jpo  ,s,rc,hs,sta,p,v,ui)[gm?] ->
GM(l,bm,ox-1,jpo-1,s,rc,hs,sta,p,v,ui)[gm?]
@ if (ox>0) && (jpo > 0.3*bm)
then smr(t) * lbmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Cod has oxygen reduced by 1 and metabolites reduced by 1 |
| Guard: | Oxygen must be higher than 0, metabolites must be higher than 0.3 times body mass in [g] |
| Rate: | At standard metabolic rate (smr(t)) from [9], where t is temperature, scaled for time units and cod weight (lbmf(bm)) |

This rule formalises the oxygen and metabolite expenditure for basic metabolic functions, oxygen and metabolites are removed from the cod at the a basic metabolic rate, which is sensitive to temperature. The guard requires that sufficient free metabolites be available (above 0.3 time body mass [g], 6) if this is not the case the Starvation rule (see below) accounts for expenditure of basic metabolic functions. The rate for the process is based on experimental wet-lab findings by [9] (standard metabolic rate function - see below) normalised for cod weight, time units and oxygen units by the linear body mass function (see above).
The combination of Oxygen Intake, Standard Metabolic Rate and Starvation rules means that the available metabolic scope for oxygen is determined by the active metabolic rate minus the standard metabolic rate. However, as several processes (digestion, movement, growth, etc.) are competing over this resources in a "stochastic race" the efficiency of the other processes can vary and is not deterministic.

**standard metabolic rate - function**

```
smr :: num -> num;
smr temp = (80.1*(1-(e)^(-0.185*(temp)^(0.79))));
```

Function provides standard metabolic rate taken from [9]

$$\mathrm{SMR} = 80.1(1 - \mathrm{e}^{-0.185\mathrm{T}^{0.79}}) \frac{\mathrm{mg\ O_2}}{\mathrm{h\ kg}}, \tag{3}$$

where T is the temperature in °C.

**Starvation**

```
//Starvation
GM(l,bm   ,ox   ,jpo   ,s,rc,hs,sta,p,v,ui)[gm?] ->
GM(l,bm-si,ox-1,jpo+4,s,rc,hs,sta,p,v,ui)[gm?]
@ if (ox>0) && (jpo <= 0.3*bm)
then smr(t) * lbmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Oxygen greater than 0 and insufficient free metabolites |
| End with: | Oxygen reduced by 1, body mass reduced by si - starvation increment (0.0005[g]) and metabolites reduced by 1 |
| Guard: | Oxygen must be higher than 0, metabolites lower or equal to 0.3 times body mass in [g] |
| Rate: | At standard metabolic rate (smr(t)) [9], o, t environmental oxygen and temperature respectively; scaled for time units and cod weight ((lbmf(bm))) |

This rule formalises the oxygen and metabolite expenditure for basic metabolic functions,

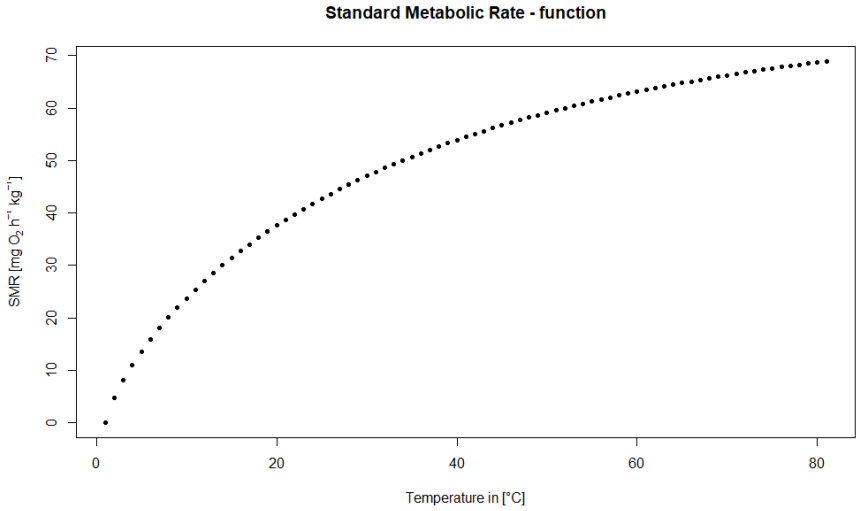



Figure 2: Standard metabolic rate taken from [9].

oxygen and metabolites are removed from the cod at the a basic metabolic rate, which is sensitive to temperature. The guard requires that insufficient free metabolites be available (below or equal 0.3 time body mass [g], 6) if this is not the case the Standard Metabolic Rate rule (see above) accounts for expenditure of basic metabolic functions. The rate for the process is based on experimental wet-lab findings by [9] (standard metabolic rate function - see below) normalised for cod weight, time units and oxygen units by the linear body mass function (see above).
The combination of Oxygen Intake, Standard Metabolic Rate and Starvation rules means that the available metabolic scope for oxygen is determined by the active metabolic rate minus the standard metabolic rate. However, as several processes (digestion, movement, growth, etc.) are competing over this resources in a "stochastic race" the efficiency of the other processes can vary and is not deterministic.
**Known further development requirements:** the starvation increment (si - 0.0005[g]) is currently a theoretical assumption and likely not one fixed number (dependence on the condition of the individual would be a probable line of investigation). Therefore further investigation and subsequent validation will be required, both for the absolute value and for a changing value dependent on physiological factors.

**Normal Growth**

```
//Normal Growth
GM(l ,bm ,ox ,jpo ,s,rc,hs,sta,p,v,ui)[gm?] ->
GM(l+lg(bm),bm+gi,ox-4,jpo-12,s,rc,hs,sta,p,v,ui)[gm?]
@ if ((ox > 27) && (jpo > 10*bm) && (cf(l,bm) > lgt))
then td(t,bm) * ebmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Cod has oxygen and metabolites removed and has gained mass (0.00587g) and length (calculated from the gained mass - (lg(bm))). |
| Guard: | Oxygen must be higher than 27, free metabolites must be above 10 times body mass and condition factor must be above length growth threshold (0.8 - loosely based on [10]) |
| Rate: | Rate is a product of temperature dependency (td(t,bm)), derived from [11] and (ebmf(bm)) |

This rule formalises "normal" growth, meaning increase of both length and mass (as opposed to compensatory growth which only increases mass, see below). Oxygen and free metabolites are removed and the mass of the cod is increased by the growth increment (gi=0.00587[g]) and the length is increased by the appropriate length - see length growth function below. The guard for this rule is sufficient oxygen, free metabolites ((see also tables for physiological priorities related to oxygen and free metabolites 5, 6)) and condition factor above the length growth threshold (lgt=0.8). The rate for this process is temperature dependent based on a function derived from [11] (temperature dependent function - see below) which accounts for different body-mass dependent temperature optimums and normalised with an exponential function accounting for size (exponential body mass function - see below).

**Known further development requirements:** together with the "Growth at low CF" rule these two rules set the boundaries for normal and compensatory growth. Here the thresholds are hard boundaries (no length growth below a condition factor of 0.8 (lgt - length growth threshold) and no compensatory growth above a condition factor of 1.2 (mgt - mass growth threshold)) this means that condition below 0.8 can only be reached through starvation and condition factors above 1.2 can not be reached. Also, these values are not validated and physiological priorities are likely guided by far more complex interactions which are currently not included in the model.

**length gain - function**

```
lg :: num -> num;
lg bm = (((bm)/aa)^(1/bb)-((bm-gi)/aa)^(1/bb))*10;
```

Length gain of individuals is based on "standard weight in fish" which formalises a typical mass-lengths-relationship for a given species/stock: $W = aL^b$, were W is weight, L is length and a and b are empirically determined factors. The value of a (aa=0.00743) and b (bb=3.06) are derived from current survey data (BITS). The "standard weight in fish" equation is transformed to calculate gain in length given a gain in mass (gi=0.00587[g]) for a specific original mass (see also Figure 3):

$$\Delta \mathrm{l} = \sqrt[\mathrm{b}]{\frac{bm}{\mathrm{a}}} - \sqrt[\mathrm{b}]{\frac{bm - \mathrm{gi}}{\mathrm{a}}}. \tag{4}$$

**condition factor - function**

```
cf :: num -> num -> num;
cf l bm = (100000*bm)/((l)^(3));
```

The condition factor provides a relation between length an mass in order to have one number to determine "plumpness" of fish ([12]). The range for healthy Baltic cod is 0.75-1.2 with individuals at 0.6 considered to have reached the threshold for starvation ([10]). Here it is calculated as:

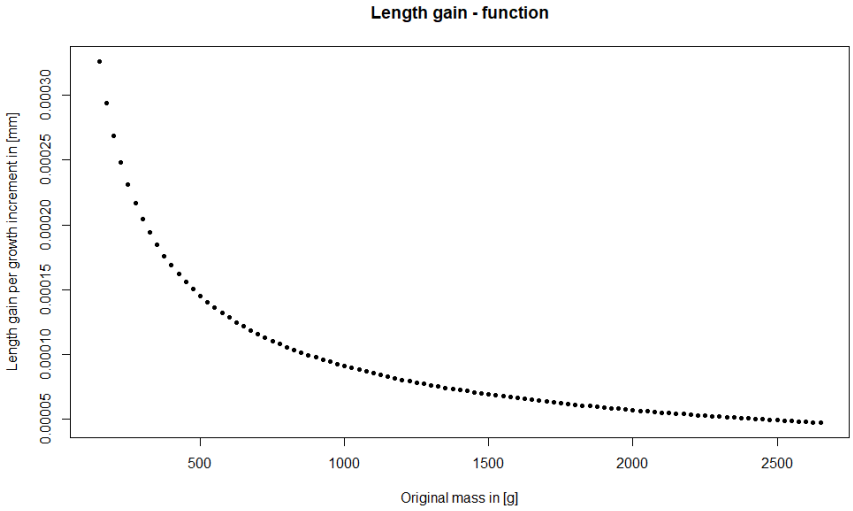



Figure 3: Length gain function, a growth increment is a gain in mass of 0.00587[g]

$$cf = 100000 \cdot \frac{bm}{l^3} \frac{[\mathrm{g}]}{[\mathrm{mm}^3]} \tag{5}$$

were $bm$ is the body mass in [g] and $l$ is the length in [mm]. For mature fish gonad weight is currently not taken into account.

**temperature dependence - function**

```
td :: num -> num -> num;
td temp bm = (-0.007*(temp-(22*(bm)^(-0.1507)))^(2) + 1);
```

This function is based on the findings of [11] and scales rates for body-mass dependent temperature optimum, this reflects that small individuals have a higher optimal temperature than larger ones.

$$\mathrm{td}(t, bm) = -0.007 \cdot (t - (22 \cdot bm^{-0.1507}))^2 + 1. \tag{6}$$

With $t$ as the temperature and $bm$ as the body mass of the cod.

**Growth at low CI**

```
//Growth at low CI
GM(l,bm   ,ox   ,jpo    ,s,rc,hs,sta,p,v,ui)[gm?] ->
GM(l,bm+gi,ox-4,jpo-12,s,rc,hs,sta,p,v,ui)[gm?]
@ if ((ox > 28) && (jpo > 10*bm) && (cf(l,bm) <= mgt))
then td(t,bm) * ebmf(bm) else 0;
```

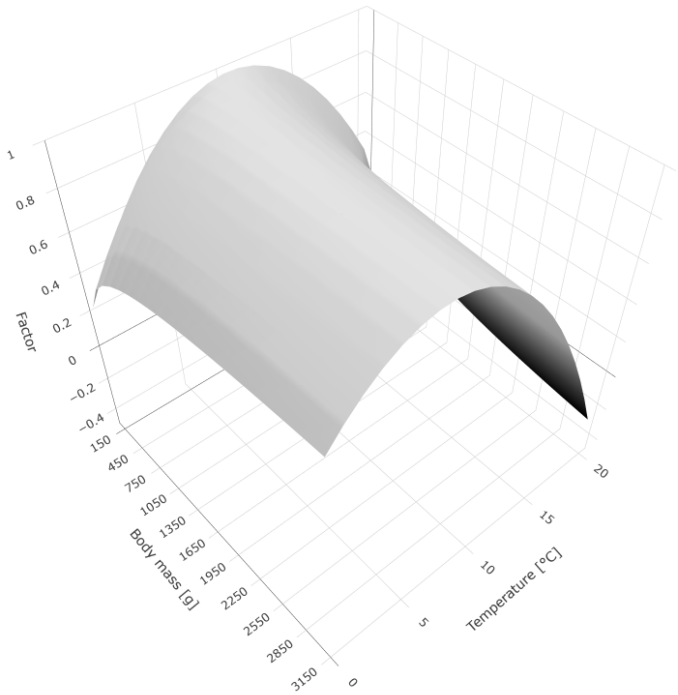



Figure 4: Temperature dependency for growth and feed conversion based on [11].

| | |
|---|---|
| Begin with: | Cod has sufficient oxygen, free metabolites and a condition factor below mass growth threshold (mgt=1.2) |
| End with: | Cod has increased mass and oxygen and free metabolites removed |
| Guard: | Oxygen must be higher than 27, free metabolites must be below 10 times body mass and condition factor must be below growth threshold (1.2 - based on [10]) |
| Rate: | Rate is a product of temperature dependency (td(t,bm)), derived from [11] and (ebmf(bm)) |

This rule formalises "compensatory" growth, meaning increase of only mass (as opposed to normal growth which increases both mass and length, see above). Oxygen and free metabolites are removed and the mass of the cod is increased by the growth increment (gi=0.00587[g]). The guard for this rule is sufficient oxygen, free metabolites ((see also tables for physiological priorities related to oxygen and free metabolites 5, 6)) and a condition factor below the mass growth threshold (mgt=1.2). The rate for this process is temperature dependent based on a function derived from [11] (temperature dependent function - see below) which accounts for different body-mass dependent temperature optimums and normalised with an exponential function accounting for size (exponential body mass function - see below).
**Known further development requirements:** together with the "Normal Growth" rule these two rules set the boundaries for normal and compensatory growth. Here the thresholds are hard boundaries (no length growth below a condition factor of 0.8 (lgt - length growth threshold) and

no compensatory growth above a condition factor of 1.2 (mgt - mass growth threshold)) this means that condition below 0.8 can only be reached through starvation and condition factors above 1.2 can not be reached. Also, these values are not validated and physiological priorities are likely guided by far more complex interactions which are currently not included in the model.

**Stomach Evacuation**

```
//Stomach Evacuation
GM(l,bm,ox___,jpo___,s,rc,hs,sta,p,v,ui)
[Stc(pc,va,uis)[AJPO(od,et,df) + sc?] + gm?] ->

GM(l,bm,ox-od,jpo+et,s,rc,hs,sta,p,v,ui)
[Stc(pc,va,uis)[_____    sc?] + gm?]
@ if ((ox > 25) && (jpo < 150*bm))
then 0.05 * td(t,bm) * ebmf(bm) * count(sc?,'AJPO') * df else 0;
```

| | |
|---|---|
| Begin with: | Cod has AJPO (prey pieces) in stomach |
| End with: | AJPO is transferred into free metabolites while using oxygen at a rate dependent on the properties of the prey |
| Guard: | Oxygen must be higher than 25, free metabolites must be above 10 times body mass |
| Rate: | Rate is a product of temperature dependency (td(t,bm)), derived from [11], (ebmf(bm)), number of prey pieces in stomach (resulting in an exponential relationship), the digestive factor (df) (a property of the prey) and a calibrating factor of 0.05 |

This rule formalises the digestion of food. After ingestion prey is formalised inside an individuals' stomach as pieces of prey (AJPO - 0.013[g] per piece) which maintain properties of the prey type informative to digestion, namely energy transfer to the digesting predator, oxygen requirement for digestion and a factor accounting for probable relative digestive speed. On execution of the rule the respective amount of metabolites are transferred to the cod and an the respective amount of oxygen and the piece of prey are removed. The guard for this rule is sufficient oxygen and metabolites (see also tables for physiological priorities related to oxygen and free metabolites 5, 6). The guard ensures that only with sufficient oxygen and available energy digestion takes place. The rate is a combination of the temperature dependency function (td(t,bm) - see above), exponential body mass function (ebmf(bm) - see above), the amount of the prey remaining in the stomach and a calibrating factor of 0.05.

**Ingestion**

```
//Ingestion
GM(l,bm,ox,jpo,s,rc,1,sta,p,v,ui)
 [Stc(pc,va,uis)[sc?] + gm?]
   + Prey(pn,ps,pod,pet,pdf,ast,cot):np  ->

GM(l,bm,ox,jpo,s,rc,0,sta,p,v,ui)
 [Stc(pc+(ps*0.013),va,uis)[(ps) AJPO(pod,pet,pdf) + sc?] + gm?]

@ if (jpo < 50*bm) &&
     ((ps+count(sc?,'AJPO') < (msv(bm)*oda(o)*td(t,bm))))
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod has been successful at hunting and in the same compartment as prey |
| End with: | Cod ingested suitable prey and returned to unsuccessful state |
| Guard: | Cod does not exceed free metabolites of 50 times body mass ('is hungry') and prey to be ingested plus stomach content does not exceed maximum stomach volume (msv(bm) - see below) which is further regulated by oxygen dependent appetite (oda(o) - see below) and temperature dependency (td(t,bm) - see above). |
| Rate: | Rate is 1 |

This rules formalises the process of ingestion. A cod has hunted successfully (see Hunting with success rule below) and is in the proximity of suitable (not to large) prey. The prey is transformed into food pieces (AJPO) which maintain the digestive properties (oxygen requirements (od - oxygen demand), energy transfer (et - energy transfer) and digestive speed (df - digestive factor)) of prey item inside the stomach upon execution of the rule. The rate is one as the regulation of the combined process of hunting and ingesting is covered in the Hunting rule (see below).

**Known further development requirements:** the oxygen dependent appetite function reinforces the wet-lab findings of oxygen dependent growth [9]. This is a stand-in for processes that actually regulate appetite. The rational is that sudden low oxygen availability might cause fish to regurgitate their stomach content establishing a connection between ambient oxygen saturation and that the wet-lab findings can inform the shape of the curve.

**maximum stomach volume - function**

```
//maximum stomach volume
msv:: num -> num;
msv bm = (bm*70/13);
```

Maximum stomach volume is calculated based on the mass of prey pieces (AJPO - 0.013[g]) resulting in 7% body mass.

**oxygen dependent appetite - function**

```
//oxygen dependent appetite
oda:: num -> num;
oda ox = 0.93*(1 - (e)^(-0.03*(ox - 10)));
```

This function is based on wet-lab findings of oxygen dependent growth [9] emulating the shape

of the curve for growth:

$$\text{oda}(o) = 0.93 * (1 - e^{(-0.03*(o-10))}), \tag{7}$$

where $o$ is the ambient oxygen saturation in [%].

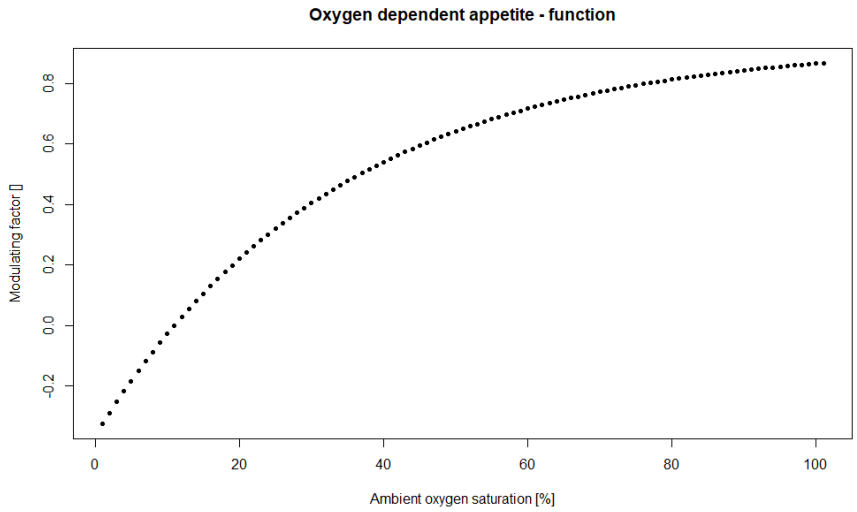



Figure 5: Oxygen dependency for appetite based on [9].

**Hunting with Success**

```
//Hunting with Success
GM(l,bm,ox,jpo,s,rc,0,sta,p,v,ui)[gm?] ->
GM(l,bm,ox,jpo,s,rc,1,sta,p,v,ui)[gm?]
@ if (jpo > 0.05*bm))
then (0.016*o-0.52)*10*lbmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Cod unsuccessfully at hunting |
| End with: | Cod successful at hunting; now able to ingest prey |
| Guard: | Cod has to have a minimal amount of free metabolites available (0.05 time body mass) |
| Rate: | Combination of a linear relationship between ambient oxygen saturation and rate shifted to set in at beginning of metabolic scope, the (lbmf(bm)) accounts for increase in hunting with increasing size and the resulting product is increased tenfold as a result of calibration |

This rule formalises a successful hunt, meaning the intent and action of hunting. The attribute of hunting success (hs) is changed from 0 to 1, this being a guard for the ingestion of prey (see ingestion of prey rule above). Please note that this Rule is not active in the validation experiments, as these are comparisons to wet-lab experiments and here cod are fed and their 'hunting' is always successful. This is realised by letting the cod be successful hunters during each feeding session (setting the hunting success attribute to 1 and not having it switch to 0). The guard for this rule is that there is sufficient metabolites available (6). The rate, which ties the behaviour to ambient oxygen is the same as for activity but with a different calibrating factor

(10). The combination of the two rules (Hunting with Success and Activity) allows to implement different relationships between the energy expended and the amount of prey an individual can catch.

**Activity**

```
//Activity
GM(l,bm,ox__,jpo__,s,rc,hs,sta,p,v,ui)[gm?] ->
GM(l,bm,ox-2,jpo-1,s,rc,hs,sta,p,v,ui)[gm?]
@ if ((ox > 26) && (jpo > 0.25*bm))
then (0.016*o-0.52) * 5 * lbmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Cod expended oxygen and metabolites |
| Guard: | Oxygen must be above 26 and metabolites must be above 0.25 time body mass |
| Rate: | Combination of a linear relationship between ambient oxygen saturation and rate shifted to set in at beginning of metabolic scope, the (lbmf(bm)) accounts for increasing expenditure with increasing size and the resulting product is quintupled as a result of calibration |

This Rule formalises the activity of cod - it currently is a proxy for a range of factors (general movement, vertical movement, migration, swimming and fleeing), bundling these to account for the use of energy and oxygen of these behaviours. This rule is an extreme case of one concept acting as proxy for a large number of processes. The cod expends energy and oxygen upon execution of the process. The guard fo this rule is sufficient oxygen and metabolites (see also tables for physiological priorities related to oxygen and free metabolites 5, 6). The rate, which ties the behaviour to ambient oxygen is the same as for Hunting with success but with a different calibrating factor (5). The combination of the two rules (Hunting with Success and Activity) allows to implement different relationships between the energy expended and the amount of prey an individual can catch.

**Death from starvation**

```
//Death from starvation
GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] ->
_____________________________________
@ if (cf(l,bm) < lcft)
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Removed (dead) Cod |
| Guard: | Condition factor of Cod must be below the lcft (lethal condition factor threshold - 0.6 [10]) |
| Rate: | Rate is 1 |

This rule formalises death from starvation. The Cod will die when its condition factor declines to 0.6 which is an empirically based number, surveys not recording live animals below this threshold (BITS). The rate of 1 reflects that at this point death is inevitable [10].

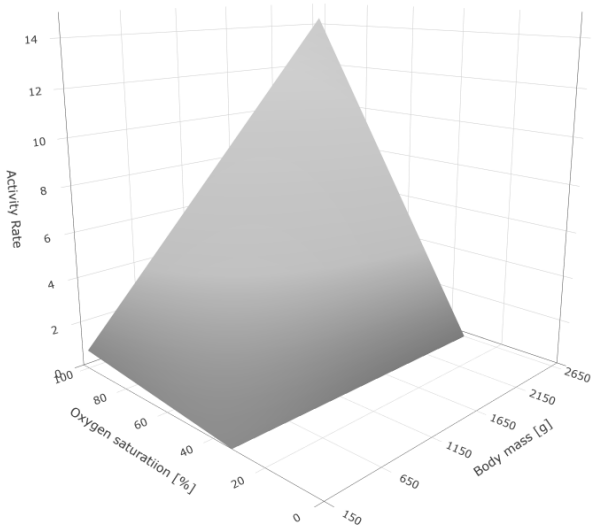



Figure 6: Activity Rate result of a product between a linear relationship between ambient oxygen saturation and rate shifted to set in at beginning of metabolic scope and the (lbmf(bm)) accounting for increasing expenditure with increasing size and quintupled as a result of calibration

**Death from asphyxiation**

```
//Death from Asphyxiation
GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] ->
_____________________________________________
@ if (ox < 1)
then 0.1 else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Removed (dead) Cod |
| Guard: | Cod contains no oxygen |
| Rate: | Lower than 1 (0.1) to account for short term survival in oxygen depleted state |

This Rule formalises the death of cod from asphyxiation. To account for short term survival when oxygen reserves are depleted the rate is set at 0.1. This rate was chosen arbitrarily but is validated by the asphyxiation experiments comparing modelled Cod behaviour to cod in wet-lab settings [13] - see also “asphyxiation experiment”.

Table 5: Physiological priorities by available oxygen:
Physiological functions by dependence on available oxygen (-: process is executed regardless of oxygen status, >x: process requires this amount of oxygen be present), Rank indicated order priority for processes when oxygen is limited.

| **Rule** | **Rank** | **Guard value** |
|---|---|---|
| Oxygen Intake | - | - |
| Standard Metabolic Rate | - | - |
| Starvation | - | - |
| Normal Growth | 3 | >28 |
| Growth at low CF | 3 | >28 |
| Stomach Evacuation | 1 | >25 |
| Ingestion | - | - |
| Hunting | - | - |
| Activity | 2 | >26 |
| Death from starvation | - | - |
| Death from asphyxiation | -1 | <1 |
| Gonad energy allocation | 1 | >25 |

Table 6: Physiological priorities by available metabolites:
Physiological functions by dependence on available metabolites (-: process is executed regardless of metabolite status, >x: process requires this amount of metabolites be present), Rank indicated order priority for processes when metabolites are limited.

| **Rule** | **Rank** | **Guard value** |
|---|---|---|
| Oxygen Intake | - | - |
| Standard Metabolic Rate | 3 | > 0.3*bm |
| Starvation | -1 | < 0.3*bm |
| Normal Growth | 4 | > 10*bm |
| Growth at low CF | 4 | > 10*bm |
| Stomach Evacuation | -3 | < 150*bm |
| Ingestion | -2 | < 50*bm |
| Hunting | 1 | > 0.05*bm |
| Activity | 2 | > 0.25*bm |
| Death from starvation | - | - |
| Death from asphyxiation | - | - |
| Gonad energy allocation | 4 | 10*bm |

### 2.7.2 Rule by Rule: Variable Appetite

To allow for and, if desired, control the variability in food intake the basic model, specifically the "Food intake" - Rule was augmented. Conceptually appetite here is a proxy for several aspects of food intake, such as hunting proximity of prey items, hunting success, appetite regulated by metabolic need (meaning impact of nutritional status of the individual) and appetite regulated by metabolic capacity (meaning impact of detrimental effects on digestive systems). Impact of temperature on appetite is formalised via the temperature sensitivity function. Impact of oxygen saturation on food ingestion is formalised indirectly through the required oxygen guard of the Hunting with Success Rule and directly through the oxygen dependant appetite function.

**Ingestion**

```
//Ingestion
GM(l,bm,ox,jpo,s,rc,1
,sta,p,v,ui)
[Stc(pc,va,uis)[ sc?] + gm?]
 + Prey(pn,ps,pod,pet,pdf,ast,cot):np  ->

GM(l,bm,ox,jpo,s,rc,fil(count(sc?,'AJPO'),va,bm,o,t)
,sta,p,v,ui)
[Stc(pc+(ps*0.013),va,uis)[(ps) AJPO(pod,pet,pdf) + sc?] + gm?]

@ if (jpo < 50*bm) &&
((ps+count(sc?,'AJPO') < (msv(bm)*oda(o)*td(t,bm))))
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod has been successful at hunting in the same environment at suitably small prey |
| End with: | Cod ingested suitable prey, depending on food intake level function (fil(n,para,v,bm,o,t) - see below) cod is either still successful at hunting or returned to unsuccessful state |
| Guard: | Cod does not exceed free metabolites of 50 times body mass ('is hungry') and prey to be ingested plus stomach content does not exceed maximum stomach volume (msv(bm) - see below) which is further regulated by oxygen dependent appetite (oda(o) - see below) and temperature dependency (td(t,bm) - see above). |
| Rate: | Rate is 1 |

This rules formalises the process of ingestion. A cod has hunted successfully (see Hunting with success rule above) and is in the proximity of suitable (not to large) prey. The prey is transformed into food pieces (AJPO) which maintain the digestive properties (oxygen requirements (od - oxygen demand), energy transfer (et - energy transfer) and digestive speed (df - digestive factor)) of prey item inside the stomach upon execution of the rule. The rate is one as the regulation of the combined process of hunting and ingesting is covered in the Hunting rule (see above).
Introducing the food intake level function allows for control of food intake in experiments. It also allows for foraging sessions on smaller prey without interfering with the ratio of unsuccessful hunts to energy expenditure through activity.
**Known further development requirements:** the oxygen dependent appetite function reinforces the wet-lab findings of oxygen dependent growth [9]. This is a stand-in for processes that

actually regulate appetite. The rational is that sudden low oxygen availability might cause fish to regurgitate their stomach content establishing a connection between ambient oxygen saturation and that the wet-lab findings can inform the shape of the curve.
The food intake level function is a proxy for a number of factors influencing success and duration of foraging and hunting these will have to determined and functional relationships need to be introduced.

**food intake level - function**

```
//food intake level
fil :: num -> num -> num -> num -> num -> num;
fil xn xv xbm xo xt = if (xn < (xv * 0.8 * (xbm*70/13) *
    (0.93*(1-(e)^(-0.03*(xo-10)))) *
    (-0.007*(xt-(22*(xbm)^(-0.1507)))^(2)+1))) then 1 else 0;
```

$$\text{fil}(sc, var, bm, ox, t) = \begin{cases} 1 & \text{if } sc < (var \cdot 0.8 \cdot \frac{bm \cdot 70}{13}) \cdot (0.93 \cdot (1 - \mathrm{e}^{-0.03 \cdot (ox-10)})) \\ & \quad \cdot (-0.007 \cdot (t - (22 \cdot w^{-0.1507}))^2 + 1) \\ 0 & \text{if } sc \geq (var \cdot 0.8 \cdot \frac{bm \cdot 70}{13}) \cdot (0.93 \cdot (1 - \mathrm{e}^{-0.03 \cdot (ox-10)})) \\ & \quad \cdot (-0.007 \cdot (t - (22 \cdot w^{-0.1507}))^2 + 1) \end{cases} \tag{8}$$

### 2.7.3 Rule by Rule: Infestation with *Contracaecum osculatum*

Please note that no mechanism for the recovery from parasites was implemented, therefore cod that are infected will never fall below the point of their highest infection. To extrapolate for population results such a mechanism would need to be added.

Roughly based on the findings of [14] the following concepts were formalised into the model "Liver.mlrj":
Increase of parasite infestation of the liver, in this model the increase is hypothesised to follow a sigmoid curve:

$$N_{parasite} = \frac{20}{1 + \mathrm{e}^{-ln - \frac{N_{parasite-1}}{N_{parasite-1}-20} + 1}}$$

where N is the number of parasites (for now as a continuous number) with an upper boundary of 20. The function is calibrated to level out at 20 parasites in the liver at the ingestion of 100 infested items of prey.

Parasitisation level of the liver impacts standard metabolic rate, decreasing it as found by [14].

Parasitisation level of the liver impacts appetite accounting for the observation of decreased intestinal functionality.

**Interpretation**
The model performs as expected and delivers decreased growth for increased levels of parasitisation. It conforms to the findings of [14].

As stated by [14] there is a "chicken or egg" question to be answered. Meaning that it could not yet be determined if increase in parasites leads to decreased physiological performance or if individuals with a low condition index are more prone to high parasite infestation. To aid in the investigation both suggested hypothesis need to be formalised.

**sigmoidal parasite increase - function version I**

```
//sigmoidal increase of parasites
//input variables: number of parasites in Liver, parasitation of
    prey (yes/no)
// -> experimental function; carrying capacity at 20 total reached
    after ingestion of 100 infected fish
spi :: num -> num -> num;
spi 0 0 = 0;
spi 0 1 = 0.14783;
spi y x = 20/(1+(e)^(-0.1*(((10*log(-y/(y-20))+50)+x)-50)));
```

$$\text{spi}(co, cot) = \begin{cases} 0 & \text{if } co = 0 \ \& \ cot = 0 \\ 0.14783 & \text{if } co = 0 \ \& \ cot = 1 \\ \frac{20}{1+e^{-0.1*((10*ln(\frac{-y}{y-20})+50)+x)-50)}} & \text{if } co = x \ \& \ cot = y \end{cases} \quad (9)$$

The variables co and cot are the number of parasites in the liver and weather or not (0,1) the prey is infected respectively.

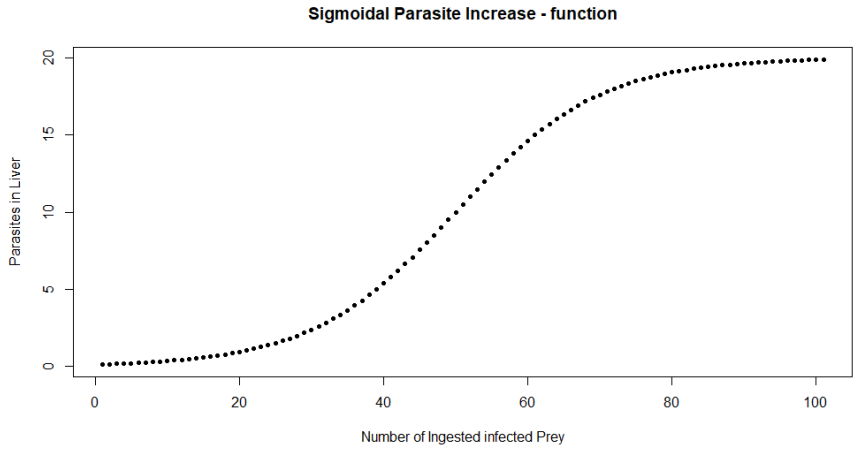


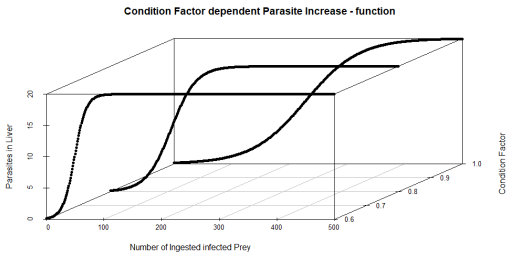

Figure 7: Shape of curve for sigmoidal parasite increase.

**condition factor-based parasite increase - function version II**

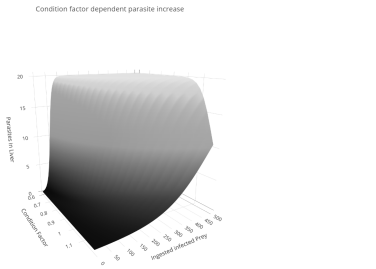
```
//condition factor-based parasite increase
//input variables: body mass, length, cod parasitation, prey
    parasitation
cfdpi :: num -> num -> num -> num -> num;
cfdpi bm l 0 0 = 0;
cfdpi bm l 0 1 = 0.14783 * 10*e^(-3.838*(100000*bm/(l^(3))));
cfdpi bm l y x = 20/(1+(e)^(-0.1*(((10*log(-y/(y-20))+50) +
        (x*(10*e^(-3.838*(100000*bm/l^(3)))))))-50)));
```

$$\text{cfdpi}(bm, l, co, cot) = \begin{cases} 0 & \text{if } co = 0 \ \& \ cot = 0 \\ 0.14783 & \text{if } co = 0 \ \& \ cot = 1 \\ \frac{20}{1+e^{-0.1*((10*ln(\frac{-y}{y-20})+50)+x)-50)}} & \text{if } co = x \ \& \ cot = y \end{cases} \tag{10}$$

The variables bm and l are the body mass and length respectively and are used to calculate the cods condition and the variables co and cot are the number of parasites in the liver and weather or not (0,1) the prey is infected respectively.




(a) Three examples of curves for the condition factor based parasite increase function




(b) Resulting surface of the condition factor based parasite increase function

Figure 8: Representations of the condition factor based parasite increase function

### 2.7.4 Rule by Rule: Habitat

The habitat consists of boxes which have their abiotic conditions as attributes. These are currently updated monthly based on survey data. This means that environmental conditions are being forced and are conceptually part of the experimental setup.

**Habitat (example)**

```
Box(1,t  ,o  ,sal,d,z,substr,pa,ar,mind,mand)[rest?] ->
Box(1,4  ,64,35 ,d,z,substr,pa,ar,mind,mand)[rest?]
@EXACT(219000);//April
```

| | |
|---|---|
| Begin with: | Environmental box as a defined temperature, oxygen saturation and salinity |
| End with: | Environmental box has a new temperature, oxygen saturation and salinity |
| Guard: | no Guard |
| Rate: | no Rate, changes are forced at specific point (monthly intervals) |

The monthly timing was calculated by multiplying the TU for a day (2400) by the number of days in a year (365) and dividing the result by 12, then multiplying this by the number of the ending month in a calender year ($\frac{2400*365}{12} \cdot 3$).

**Day**

```
Box(1,t1,o1,sal1,d1   ,z1,substr1,pa1,ar1,mind1,max1)[r1?]+
Box(13,t13,o13,sal2,d2   ,z13,substr13,pa13,ar13,mind13,max13)[r13?]+
Box(25,t25,o25,sal3,d3   ,z25,substr25,pa25,ar25,mind25,max25)[r25?]+
Box(37,t37,o37,sal4,d4   ,z37,substr37,pa37,ar37,mind37,max37)[r37?]->

Box(1,t1,o1,sal1,d1+1,z1,substr1,pa1,ar1,mind1,max1)[r1?]+
Box(13,t13,o13,sal2,d2+1,z13,substr13,pa13,ar13,mind13,max13)[r13?]+
Box(25,t25,o25,sal3,d3+1,z25,substr25,pa25,ar25,mind25,max25)[r25?]+
Box(37,t37,o37,sal4,d4+1,z37,substr37,pa37,ar37,mind37,max37)[r37?]

@EACH2400;
```

| | |
|---|---|
| Begin with: | Box has a day of the year as attribute |
| End with: | Day of the year is increased by one |
| Guard: | no Guard |
| Rate: | no Rate, changes are scheduled every 2400TU (each day) |

This rule formalises the passing of days. Every 2400TU (every day) the counter is increased by one. This allows to access environmental information such a day length or time of year (see below). For fragmented habitats rule is expanded to all Boxes.

**Environmental conditions 4 boxes**

```
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,6,83,36,d,z,substr,pa,ar,0,1)[rest?]@EXACT(1);//Januar
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,4,73,36,d,z,substr,pa,ar,0,1)[rest?]@EXACT(73000);//Februar
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,3,76,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(146000);//Maerz
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,4,64,35,d,z,substr,pa,ar,0,1)[rest?]@EXACT(219000);//April
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,5,62,32,d,z,substr,pa,ar,0,1)[rest?]@EXACT(292000);//Mai
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,8,61,38,d,z,substr,pa,ar,0,1)[rest?]@EXACT(365000);//Juni
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,10,98,35,d,z,substr,pa,ar,0,1)[rest?]@EXACT(438000);//Juli
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,15,99,35,d,z,substr,pa,ar,0,1)[rest?]@EXACT(511000);//August
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,18,67,35,d,z,substr,pa,ar,0,1)[rest?]@EXACT(584000);//September
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,11,95,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(657000);//Oktober
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,10,76,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(730000);//November
Box(1,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(1,8,80,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(803000);//Dezember
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,6,99,38,d,z,substr,pa,ar,0,1)[rest?]@EXACT(13);//Januar
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,62,34,d,z,substr,pa,ar,0,1)[rest?]@EXACT(72988);//Februar
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,81,34,d,z,substr,pa,ar,0,1)[rest?]@EXACT(145988);//Maerz
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,83,31,d,z,substr,pa,ar,0,1)[rest?]@EXACT(218988);//April
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,84,35,d,z,substr,pa,ar,0,1)[rest?]@EXACT(291988);//Mai
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,64,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(364988);//Juni
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,6,91,38,d,z,substr,pa,ar,0,1)[rest?]@EXACT(437988);//Juli
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,70,34,d,z,substr,pa,ar,0,1)[rest?]@EXACT(510988);//August
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,93,35,d,z,substr,pa,ar,0,1)[rest?]@EXACT(583988);//September
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,6,88,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(656988);//Oktober
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,5,77,33,d,z,substr,pa,ar,0,1)[rest?]@EXACT(729988);//November
Box(13,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(13,7,84,34,d,z,substr,pa,ar,0,1)[rest?]@EXACT(802988);//Dezember
```

```
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,70,36,d,z,substr,pa,ar,0,1)[rest?]@EXACT(25);//Januar
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,65,32,d,z,substr,pa,ar,0,1)[rest?]@EXACT(72976);//Februar
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,67,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(145976);//Maerz
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,72,33,d,z,substr,pa,ar,0,1)[rest?]@EXACT(218976);//April
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,75,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(291976);//Mai
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,67,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(364976);//Juni
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,71,32,d,z,substr,pa,ar,0,1)[rest?]@EXACT(437976);//Juli
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,72,32,d,z,substr,pa,ar,0,1)[rest?]@EXACT(510976);//August
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,72,33,d,z,substr,pa,ar,0,1)[rest?]@EXACT(583976);//September
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,89,34,d,z,substr,pa,ar,0,1)[rest?]@EXACT(656976);//Oktober
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,77,32,d,z,substr,pa,ar,0,1)[rest?]@EXACT(729976);//November
Box(25,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(25,7,84,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(802976);//Dezember
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,8,67,37,d,z,substr,pa,ar,0,1)[rest?]@EXACT(37);//Januar
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,7,69,37,d,z,substr,pa,ar,0,1)[rest?]@EXACT(72964);//Februar
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,5,91,38,d,z,substr,pa,ar,0,1)[rest?]@EXACT(145964);//Maerz
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,4,86,32,d,z,substr,pa,ar,0,1)[rest?]@EXACT(218964);//April
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,5,78,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(291964);//Mai
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,6,68,37,d,z,substr,pa,ar,0,1)[rest?]@EXACT(364964);//Juni
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,5,63,38,d,z,substr,pa,ar,0,1)[rest?]@EXACT(437964);//Juli
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,6,85,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(510964);//August
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,7,91,31,d,z,substr,pa,ar,0,1)[rest?]@EXACT(583964);//September
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,8,88,31,d,z,substr,pa,ar,0,1)[rest?]@EXACT(656964);//Oktober
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,9,64,34,d,z,substr,pa,ar,0,1)[rest?]@EXACT(729964);//November
Box(37,t,o,sal,d,z,substr,pa,ar,mind,mand)[rest?]->
Box(37,9,71,39,d,z,substr,pa,ar,0,1)[rest?]@EXACT(802964);//Dezember
```

| | |
|---|---|
| Begin with: | Environmental boxes |
| End with: | Environmental boxes with changed attributes |
| Guard: | no guard |
| Rate: | Rules are executed for every month at given simulation times using the EXACT function |

Taken from survey data environmental conditions are updated on a monthly basis.

### 2.7.5 Rule by Rule: Reproduction

Reproduction is interlinked with physiology, behaviour and environment: drain on physiological resources to develop gonads, behaviour such as moving to spawning grounds and impact on both from environmental (abiotic) conditions. To ground different parts of reproduction a formalised reproductive cycle (rc) was introduced, this assigns a "reproductive status" for each relevant phase: 'spent', 'preparing', 'ready to spawn' and 'spawning'.
The physiological tie-in is formalised in the following three rules:

**Gonad energy allocation**

```
//Gonad energy allocation
GM(l,bm,ox ,jpo  ,s,rc,hs,sta,p,v,ui)[
Gon(s,n,mog,ec  ,gw  ,uig) + gm?] ->

GM(l,bm,ox-4,jpo-12,s,rc,hs,sta,p,v,ui)[
Gon(s,n,mog,ec+gi,gw+gi,uig) + gm?]
@ if ((rc==2) && (ox > 27) && (jpo > 10*bm))
then td(t,bm) * ebmf(bm) else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Cod with reduced oxygen and metabolites and gonad energy content and gonad weight increased |
| Guard: | Cod reproductive cycle status must be at 'preparing' (2), have sufficient oxygen (above 27) and sufficient metabolites (above 10*bm) |
| Rate: | Rate is a product of temperature dependency - function (td(t,bm)) derived from [11] and the exponential body mass function (ebmf(bm)) |

This rule formalises the energy allocation for the growth and development of gonads. Oxygen and free metabolites are removed and energy content (ec [g]) and mass of gonads (gw [g]) are increased. Where energy content keeps track of resources allocated per developmental stage, of which there are eight and mass of gonads keeps track of total resources allocated to gonads. The energy-mass-balance considerations are the same as with growth (see above) using the same growth increment (gi=0.00587[g]) is used for the same amount of resources (4 units of oxygen and 12 units of metabolites). The guard for this rule requires the cod to have a reproductive cycle status of 'preparing', sufficient oxygen and metabolites. By formally separating energy allocation and growth of gonads (see below) further differentiation of the two processes, for example the introduction of skipped spawning or low quality of eggs can be implemented separately form the energy-mass-balance considerations implemented here.

**Gonad growth females**

```
//Gonad growth f
GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[
Gon(s,n,mog__,ec,gw,uig) + gm?] ->

GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[
Gon(s,n,mog+1,0__,gw,uig) + gm?]
@ if ((mog<8) && (s==0) && (0.08*0.15*bm < ec))
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Gonad energy content set to 0 and maturity of gonads increased increased by 1 (8 phases in total) |
| Guard: | Gonad maturity is below 8, sex is female and gonad energy content (in [g]) is below 0.08*0.15 times body mass |
| Rate: | Rate is 1 |

This rule formalises the development of gonads for females. The energy content (ec) of the gonads accumulated so far is reset to zero and the maturity of gonads (mog) is increased by one. The guard requires that the gonads are not fully mature (mog below 8), the sex of the individual is female and the energy content (ec) of the gonads has risen above 0.08*0.15*body mass of the cod. Here the 0.08 represents the assumption that female gonads are about 8% of the individuals body mass. To account for growth of the individual parallel to gonad development and inefficiencies of the process one eights of 120% rather than 100% is the threshold per developmental step: 0.15. The rate is one.

**Gonad growth males**

```
//Gonad growth m
GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[
Gon(s,n,mog__,ec,gw,uig) + gm?] ->

GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[
Gon(s,n,mog+1,0__,gw,uig) + gm?]
@ if ((mog<8) && (s==1) && (0.05*0.15*bm < ec))
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod |
| End with: | Gonad energy content set to 0 and maturity of gonads increased increased by 1 (8 phases in total) |
| Guard: | Gonad maturity is below 8, sex is male and gonad energy content is below 0.05*0.15 times body mass |
| Rate: | Rate is 1 |

This rule formalises the development of gonads for males. Currently the only difference to development of female gonads is the assumption that these are roughly 5% of body mass.

The transition between the different states of the reproductive cycle is formalised in the following four rules:

**Spent to preparing**

```
//Spent to prepering
GM(l,bm,ox,jpo,s,1,hs,sta,p,b,ui)[
Gon(s,n,mog,ec,gw,uig) + gm?] ->

GM(l,bm,ox,jpo,s,2,hs,sta,p,b,ui)[
Gon(s,4,0___,ec,gw,uig) + gm?]
@ if
     (cf(l,bm) > 0.65)   //condition index is high enough
&&   (dl(d)>10)          //AND daylength higher ten
&&   (dl(d)<12.5)        //AND daylength lower
&&   (dldt(d)>0)         //AND first half of the year
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod with reproductive cycle status of 'spent' (1) |
| End with: | Cod with reproductive cycle status of 'preparing' (2) |
| Guard: | Condition index above 0.65, daylength longer than 10 hours and shorter than 12.5 hours and it is between the winter and summer solstice - |
| Rate: | Rate is 1 |

This rules formalises the transition in reproductive cycle status from "spent" and "preparing". The guard requires a condition index above 0.65, a daylength between ten and twelve and a half hours, between the winter and summer solstice (meaning with this setup transition takes place beginning of march).

**Known further development requirements:** the guards/conditions formalised here are merely one preliminary hypothesis. Implicitly it assumed that onset of gonadal development is photo-periodic and additionally conditional to sufficient body condition. Further input from domain experts and different hypothesis should be explored.

**daylength - function**

```
dl :: num -> num;
dl day = 5.13*sin(0.017*day-1.377)+12;
```

The daylength is calculated based on the latitude of the Bornholm Island (55°08'35.0"N):

$$\mathrm{dl(d)} = 5.13 \cdot \sin(0.017 \cdot \mathrm{d} - 1.377) + 12, \tag{11}$$

where d is the calendar day starting on the first of January (note that day number above 365 will continue to produce proper results), (see Figure 9).

**derivative of daylight hours - function**

```
dldt :: num -> num;
dldt day = 0.08721*cos(0.017*day-1.377);
```

The derivative of daylength is calculated based on the daylength function (see above):

$$\mathrm{dldt(d)} = 0.08721 \cdot \cos(0.017 \cdot \mathrm{d} - 1.377), \tag{12}$$

were d is the calendar day starting on the first of January (note that day number above 365 will continue to produce proper results), (see Figure 10).

**Preparing to ready to spawn**

```
Box(bn,t,o,sal,d,z,su,pa,ar,mind,mand)[
GM(l,bm,ox,jpo,s,2,hs,sta,p,b,ui)[
Gon(s,n,mog,ec,gw,uig) + gm?] + box?] ->

Box(bn,t,o,sal,d,z,su,pa,ar,mind,mand)[
GM(l,bm,ox,jpo,s,3,hs,sta,p,b,ui)[
Gon(s,n,mog,ec,gw,uig) + gm?] + box?]
@ if     (mog==8)                    //gonads are mature
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod with reproductive status of 'preparing' |
| End with: | Cod with reproductive status of 'ready to spawn' |
| Guard: | Gonads are mature (maturity of gonads: mog=8) |
| Rate: | Rate is 1 |

This rule formalises the transition form reproductive status of "preparing" to "ready to spawn". The guard requires that the gonads be mature and the rate is one. The status of "ready to spawn" has physiological consequences: no more resources are allocated to gonad growth; and behavioural consequences: movement to spawning grounds begins.

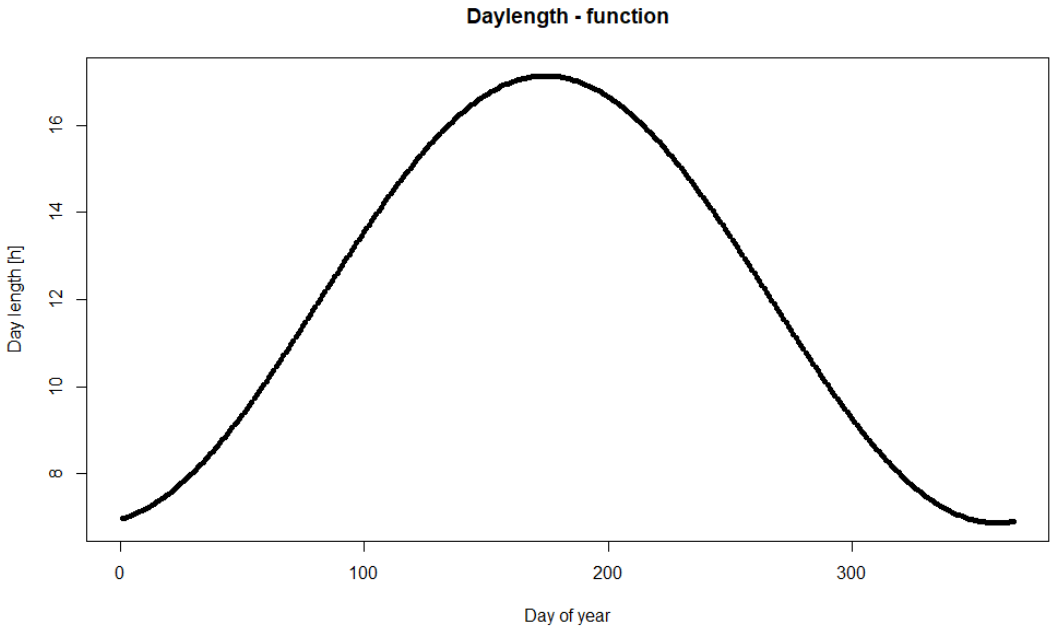



Figure 9: Length of day calculated from calendar day for the latitude of the Bornholm Island (55°08'35.0"N).

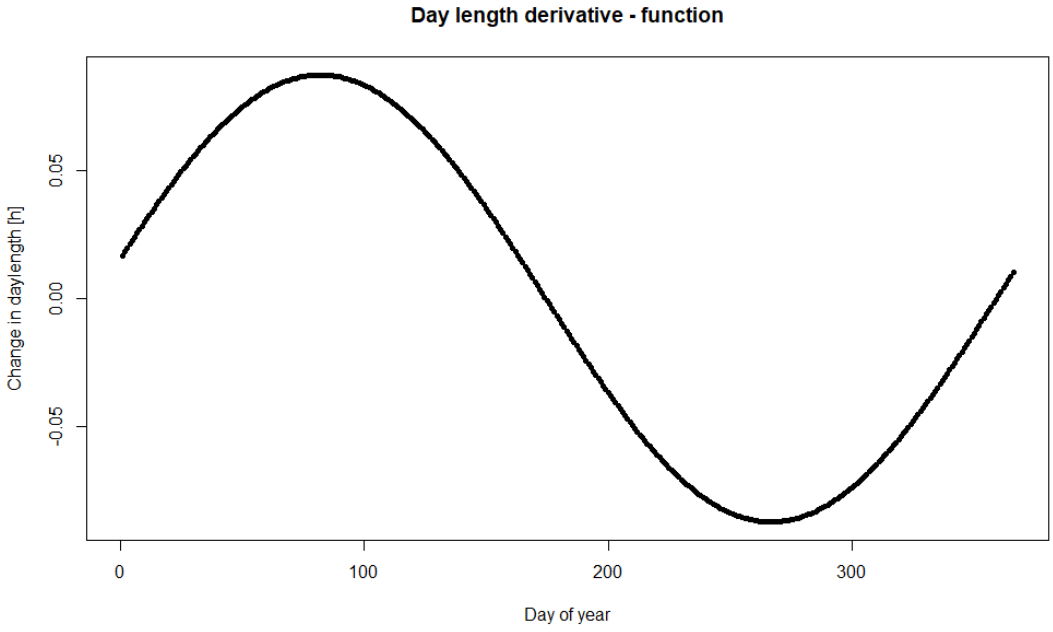



Figure 10: Change in length of day calculated from calendar day for the latitude of the Bornholm Island (55°08'35.0"N).

**Spawning and Mating**

```
//RC spawning - mating
Box(bnf,tf,of,salf,df,zf,suf,paf,arf,mindf,mandf)[
GM(lf,bmf,oxf,jpof,0,3,hsf,staf,pf,bf,uif)[
Gon(sf,nf,mogf,ecf,gwf,uigf) + gmff?] +
                                box?] +
Box(bnm,tm,om,salm,dm,zm,sum,pam,arm,mindm,mandm)[
GM(lm,bmm,oxm,jpom,1,3,hsm,stam,pm,bm,uim)[
Gon(sm,nm,mogm,ecm,gwm,uigm) + gmmm?] + box?]->

Box(bnf,tf,of,salf,df,zf,suf,paf,arf,mindf,mandf)[
GM(lf,bmf,oxf,jpof,0,4,hsf,staf,pf,bf,uif)[
Gon(sf,0_,0___,ecf,0__,uigf) + gmff?] +
nf GME(0,stacomb(staf,stam)) + box?] +
Box(bnm,tm,om,salm,dm,zm,sum,pam,arm,mindm,mandm)[
GM(lm,bmm,oxm,jpom,1,4,hsm,stam,pm,bm,uim)[
Gon(sm,0_,0___,ecm,0__,uigm) + gmmm?] + box?]
@ if
   (((bnf==25) ||(bnf==26) ||(bnf==27) ||(bnf==28) ||(bnf==29)
     ||(bnf==30) ||(bnf==31) ||(bnf==32) ||(bnf==33) ||(bnf==34)
     ||(bnf==35) ||(bnf==36))
&& ((bnm==25) ||(bnm==26) ||(bnm==27) ||(bnm==28) ||(bnm==29)
     ||(bnm==30) ||(bnm==31) ||(bnm==32) ||(bnm==33) ||(bnm==34)
     ||(bnm==35) ||(bnm==36)))
&&  (staf==stam) //male and female have the same stock affiliation
&&  (nm>0)       //the male has gonads
&&  (nf>0)       //the female has gonads
&&  (mogm==8)    //the male gonads are mature
&&  (mogf==8)    //the female gonads are mature
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod with fully developed gonads one female and one male |
| End with: | Cod with empty and immature gonads, one female and one male and fertilised eggs in the amount of the females eggs |
| Guard: | Both individuals are in the same functional habitat, they have the same stock affiliation (staf=stam), both have (nm and nf above 0) mature (mogm and mogf is 8) gonads |
| Rate: | Rate is 1 |

This rule formalises spawning/mating. This version of the rule takes fragmentation of functional habitats into account and is therefore slightly more elaborate. A female and a male cod with the reproductive status of 'ready to spawn' are in the same functional habitat, but may be in different boxes. Currently no batch spawning is formalised but implementing a separate reproductive status of 'spawning' provides the infrastructure of implementing several spawnings taking place before the reproductive status of an individual transitions to 'spent'.

**stacomb - function**

```
stacomb :: string -> string -> string;
stacomb 'W' 'W' = 'W';
stacomb 'W' 'E' = 'H';
stacomb 'E' 'W' = 'H';
stacomb 'E' 'E' = 'E';
stacomb  x   y  = 'H';
```

The implicit requirements for the execution of this rule are a male and a female cod with fully developed gonads. Further conditions will probably be needed.

This rule formalises the act of mating which is directly linked to spawning. In this rule a female and a male with sufficiently developed gonads will produce fertilised eggs of the amount of eggs in the female. This assumes that the amount of semen from the male will always be sufficient to fertilize all the eggs.

ToDo: find out which percentage should be excluded from becoming fertilised in the first place.

The rule requires the function stacomb which assigns the correct stock affiliation to the fertilised eggs. The function reads as:

this means that if both parents have the same stock affiliation the offspring will be classed the same. If the parents have different stock affiliations or one or both of them are hybrids the offspring will be classed as a hybrid.

**Spawning to spent**

```
Box(bn,t,o,sal,d,z,su,pa,ar,mind,mand)[
GM(l,bm,ox,jpo,s,4,hs,sta,p,b,ui)[
Gon(s,n,mog,ec,gw,uig) + gm?] + box?] ->

Box(bn,t,o,sal,d,z,su,pa,ar,mind,mand)[
GM(l,bm,ox,jpo,s,1,hs,sta,p,b,ui)[
Gon(s,n,mog,ec,gw,uig) + gm?] + box?]
@ if     (n==0)                    //gonads are empty
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Cod with reproductive status 'spawning' (4) |
| End with: | Cod with reproductive status 'spent' (1) |
| Guard: | Gonads have been emptied (n=0) |
| Rate: | Rate is 1 |

This rules formalises the transition from reproductive status of "spawning" to "spent".

The basis is the 2017 physiological model which has been updated in the following ways:

**Transfer of entities to attributes:**
In the original model there was a strong focus on expressiveness of the models dynamics. To this end performance had been sacrificed by using entities when they where not conceptually required to be. In the updated version oxygen, free energy and other similar concepts have been switched from being entities to being attributes of the respective suitable entities.

**Simplification:**
During the development of the successive composition method [5] the complete development

process was repeated in a highly structured fashion. As a result some of the originally calibrated factors could be removed as they turned out not to improve the usefulness of the model.
The above described changes resulted in the "Basic.mlrj" model which is the basis for all further models presented here.

**Variability:**
To aid in comparability of results variability of the Basic metabolism was introduced. This was implemented by constraining food intake. On the level of the ingestion rule the possible size of ingestible prey is varied including zero, which represents an unsuccessful hunting event. The corollary is that the hunting/activity rule was augmented to establish an oxygen dependent, minimum energy expenditure between successful hunts. This is the model "Basic_variable.mlrj"

### 2.7.6 Rule by Rule: Movement

Since there is no explicit spatial relation of the environmental boxes to each other this relation is implyed by the movements between boxes formalised in the following rules.

**Movement from Bornholm Island Slope to Bornholm Pelagic**

```
//37::1 {bornholm slope -> bornholm pelagic}
Box(bn37,t37,o37,sal37,d37,z37,sub37,pa37,ar37,mind37,mand37)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] + ths?] +
Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[_____________________________________________one?] ->

Box(bn37,t37,o37,sal37,d37,z37,sub37,pa37,ar37,mind37,mand37)
[_____________________________________________ths?] +
Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,bn1,ui)[gm?] + one?]

@ if ((((bn37==37) ||(bn37==38) ||(bn37==39) ||(bn37==40)
       ||(bn37==41) ||(bn37==42) ||(bn37==43) ||(bn37==44)
       ||(bn37==45) ||(bn37==46) ||(bn37==47) ||(bn37==48))
    && ((bn1==1)    ||(bn1==2)    ||(bn1==3)    ||(bn1==4)
       ||(bn1==5)    ||(bn1==6)    ||(bn1==7)    ||(bn1==8)
       ||(bn1==9)    ||(bn1==10)   ||(bn1==11)   ||(bn1==12)))
       //assign functional box-groups
&& ((rc==3)
  //transition likely: if reproductive cycle at 'wanting to spawn'
    (3)
|| ((rc==1) && (dl(d1)<7))
  //OR reproductive cycle at 'spent' (1) AND daylength is shorter
    than 7 hours
|| ((rc==2) && (dl(d1)>7))))
  //OR reproductive cycle at 'preparing' (2) AND daylength is
    greater than 7 hours
then 0.00006 else 0;
```

| | |
|---|---|
| Begin with: | Cod in the Bornholm Slope environmental Box |
| End with: | Cod in the Bornholm Pelagic environmental Box |
| Guard: | The Guard requires the Boxes be the exact Boxes or their derivatives and that one of the following three conditions are met: 1) the reproductive cycle of the cod is in a state of 'wanting to spawn' OR 2) the reproductive cycle is in a state of 'spent' and the daylength is shorter than 7 hours (winter solstice ± 3 days) OR 3) the reproductive cycle is in a state of 'preparing' and the daylength is greater than 7 hours (rest of the year) |
| Rate: | The rate reflects possible weekly movement between boxes based, on expert knowledge and narratives |

The assignment of functional boxes accounts for the modelling of habitat fragmentation. The additional guards describe hypothesised behaviour.

**Movement from Bornholm Pelagic to Bornholm Basin Slope**

```
//1::13 {bornholm pelagic -> bornholm basin slope}
Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] + one?] +
Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[_____________________________________________thi?] ->

Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[_____________________________________________one?] +
Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,bn13,ui)[gm?] + thi?]

@ if ((((bn1==1)   ||(bn1==2)   ||(bn1==3)   ||(bn1==4)
      ||(bn1==5)   ||(bn1==6)   ||(bn1==7)   ||(bn1==8)
      ||(bn1==9)   ||(bn1==10)  ||(bn1==11)  ||(bn1==12))
&& ((bn13==13) ||(bn13==14) ||(bn13==15) ||(bn13==16)
   ||(bn13==17) ||(bn13==18) ||(bn13==19) ||(bn13==20)
   ||(bn13==21) ||(bn13==22) ||(bn13==23) ||(bn13==24)))
   //assign functional box-groups
&& ((rc==3)
   // transition likely: if sexual cycle at 'wanting to spawn' (3)
|| ((rc==1) && (dl(d1)<7))
   // OR sexual cycle at 'spent' (1) AND daylength is shorter than 7
      hours
|| ((rc==2) && (dl(d1)>7))))
   //OR reproductive cycle at 'preparing' (2) AND daylength is
      greater than 7 hours
then 0.00006 else 0;
```

| | |
|---|---|
| Begin with: | Cod in the Bornholm Pelagic environmental Box |
| End with: | Cod in the Bornholm basin Slope environmental Box |
| Guard: | The Guard requires the Boxes be the exact Boxes or their derivatives and that one of the following three conditions are met: 1) the reproductive cycle of the cod is in a state of 'wanting to spawn' OR 2) the reproductive cycle is in a state of 'spent' and the daylength is shorter than 7 hours (winter solstice ± 3 days) OR 3) the reproductive cycle is in a state of 'preparing' and the daylength is greater than 7 hours (rest of the year) |
| Rate: | The rate reflects possible weekly movement between boxes based, on expert knowledge and narratives |

The assignment of functional boxes accounts for the modelling of habitat fragmentation. The additional guards describe hypothesised behaviour.

**Movement from Bornholm Basin Slope to Bornholm Basin Deep**

```
//13::25 {bornholm basin -> basin deep}
Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] + thi?] +
Box(bn25,t25,o25,sal25,d25,z25,sub25,pa25,ar25,mind25,mand25)
[_____twf?] ->

Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[_____thi?] +
Box(bn25,t25,o25,sal25,d25,z25,sub25,pa25,ar25,mind25,mand25)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,bn25,ui)[gm?] + twf?]

@ if ((((bn13==13) ||(bn13==14) ||(bn13==15) ||(bn13==16)
       ||(bn13==17) ||(bn13==18) ||(bn13==19) ||(bn13==20)
       ||(bn13==21) ||(bn13==22) ||(bn13==23) ||(bn13==24))
&& ((bn25==25) ||(bn25==26) ||(bn25==27) ||(bn25==28)
   ||(bn25==29) ||(bn25==30) ||(bn25==31) ||(bn25==32)
   ||(bn25==33) ||(bn25==34) ||(bn25==35) ||(bn25==36)))
   //assign functional box-groups
&& (rc==3))
   //transition likely: if reproductive cycle at 'wanting to spawn'
    (3)
then 0.00006 else 0;
```

| | |
|---|---|
| Begin with: | Cod in the Bornholm Basin environmental Box |
| End with: | Cod in the Bornholm basin Deep environmental Box |
| Guard: | The Guard requires the Boxes be the exact Boxes or their derivatives and that the reproductive cycle of the cod is in a state of 'wanting to spawn' |
| Rate: | The rate reflects possible weekly movement between boxes based, on expert knowledge and narratives |

The assignment of functional boxes accounts for the modelling of habitat fragmentation. The additional guards describe hypothesised behaviour.

**Movement from Bornholm Basin Deep to Bornholm Basin Slope**

```
//25::13 {basin deep -> basin slope}
Box(bn25,t25,o25,sal25,d25,z25,sub25,pa25,ar25,mind25,mand25)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] + twf?] +
Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[_____thi?] ->

Box(bn25,t25,o25,sal25,d25,z25,sub25,pa25,ar25,mind25,mand25)
[_____twf?] +
Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,bn13,ui)[gm?] + thi?]

@ if ((((bn13==13) ||(bn13==14) ||(bn13==15) ||(bn13==16)
       ||(bn13==17) ||(bn13==18) ||(bn13==19) ||(bn13==20)
       ||(bn13==21) ||(bn13==22) ||(bn13==23) ||(bn13==24))
&& ((bn25==25) ||(bn25==26) ||(bn25==27) ||(bn25==28)
   ||(bn25==29) ||(bn25==30) ||(bn25==31) ||(bn25==32)
   ||(bn25==33) ||(bn25==34) ||(bn25==35) ||(bn25==36)))
   //assign functional box-groups
&& (rc==1))
   //transition likely: if reproductive cycle at 'spent' (1)
then 0.00006 else 0;
```

| | |
|---|---|
| Begin with: | Cod in the Bornholm Basin Deep environmental Box |
| End with: | Cod in the Bornholm Basin Slope environmental Box |
| Guard: | The Guard requires the Boxes be the exact Boxes or their derivatives and that the reproductive cycle of the cod is in a state of 'spent' |
| Rate: | The rate reflects possible weekly movement between boxes based, on expert knowledge and narratives |

The assignment of functional boxes accounts for the modelling of habitat fragmentation. The additional guards describe hypothesised behaviour.

**Movement from Bornholm Basin Slope to Bornholm Pelagic**

```
//13::1 {bornholm basin -> bornholm pelagic}
Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] + thi?] +
Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[_____________________________________________one?] ->

Box(bn13,t13,o13,sal13,d13,z13,sub13,pa13,ar13,mind13,mand13)
[_____________________________________________thi?] +
Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,bn1,ui)[gm?] + one?]

@ if ((((bn13==13) ||(bn13==14) ||(bn13==15) ||(bn13==16)
       ||(bn13==17) ||(bn13==18) ||(bn13==19) ||(bn13==20)
       ||(bn13==21) ||(bn13==22) ||(bn13==23) ||(bn13==24))
&& ((bn1==1)   ||(bn1==2)   ||(bn1==3)   ||(bn1==4)
   ||(bn1==5)   ||(bn1==6)   ||(bn1==7)   ||(bn1==8)
   ||(bn1==9)   ||(bn1==10)  ||(bn1==11)  ||(bn1==12)))
  //assign functional box-groups
&& (((rc==2) && (dl(d1)>10))
  //transition likely: if reproductive cycle at 'preparing' (2) AND
    daylength longer 10 hours
||  ((rc==1) && (dl(d1)>8.13))
  //OR: reproductive cycle at 'spent' (1) AND daylength is longer
    8.13 hours
||  ((rc==3) && (dl(d1)<8.13))))
  //OR: reproductive cycle at 'wanting to spawn' (3) AND daylength
    is shorter than 8.13 hours
then 0.00006 else 0;
```

| | |
|---|---|
| Begin with: | Cod in the Bornholm Basin environmental Box |
| End with: | Cod in the Bornholm Pelagic environmental Box |
| Guard: | The Guard requires the Boxes be the exact Boxes or their derivatives and that one of the following three conditions are met: 1) the reproductive cycle of the cod is in a state of 'preparing' AND the daylength is greater than 10 hours (April to mid-October) OR 2) the reproductive cycle is in a state of 'spent' AND the daylength is greater than 8.13 hours (February to mid-November) OR 3) the reproductive cycle is in a state of 'wanting to spawn' and the daylength is shorter than 8.13 hours (mid-November to February) |
| Rate: | The rate reflects possible weekly movement between boxes based, on expert knowledge and narratives |

Note that this includes conditions which will not be met: wanting to spawn does not coincide with the time span of mid-November to February!

**Movement from Bornholm Pelagic to Bornholm Island Slope**

```
//1::37 {bornholm pelagic -> bornholm slope}
Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,v,ui)[gm?] + one?] +
Box(bn37,t37,o37,sal37,d37,z37,sub37,pa37,ar37,mind37,mand37)
[_____ths?] ->

Box(bn1,t1,o1,sal1,d1,z1,sub1,pa1,ar1,mind1,mand1)
[_____one?] +
Box(bn37,t37,o37,sal37,d37,z37,sub37,pa37,ar37,mind37,mand37)
[GM(l,bm,ox,jpo,s,rc,hs,sta,p,bn37,ui)[gm?] + ths?]
@ if ((((bn1==1)  ||(bn1==2)  ||(bn1==3)  ||(bn1==4)
      ||(bn1==5)  ||(bn1==6)  ||(bn1==7)  ||(bn1==8)
      ||(bn1==9)  ||(bn1==10) ||(bn1==11) ||(bn1==12))
&& ((bn37==37) ||(bn37==38) ||(bn37==39) ||(bn37==40)
  ||(bn37==41) ||(bn37==42) ||(bn37==43) ||(bn37==44)
  ||(bn37==45) ||(bn37==46) ||(bn37==47) ||(bn37==48)))
  //assign functional box-groups
&& (((rc==2) && (dl(d1)>10))
  //transition likely: if sexual cycle at 'preparing' (2) AND
    daylength longer 10 hours
||  ((rc==1) && (dl(d1)>8.13))
  //OR: sexual cycle at 'spent' (1) AND daylength is longer 8.13
    hours
||  ((rc==3) && (dl(d1)<8.13))))
  //OR: reproductive cycle at 'wanting to spawn' (3) AND daylength
    is shorter than 8.13 hours
then 0.00006 else 0;
```

| | |
|---|---|
| Begin with: | Cod in the Bornholm Pelagic environmental Box |
| End with: | Cod in the Bornholm Slope environmental Box |
| Guard: | The Guard requires the Boxes be the exact Boxes or their derivatives and that one of the following three conditions are met: 1) the reproductive cycle of the cod is in a state of 'preparing' AND the daylength is greater than 10 hours (April to mid-October) OR 2) the reproductive cycle is in a state of 'spent' AND the daylength is greater than 8.13 hours (February to mid-November) OR 3) the reproductive cycle is in a state of 'wanting to spawn' and the daylength is shorter than 8.13 hours (mid-November to February) |
| Rate: | The rate reflects possible weekly movement between boxes based, on expert knowledge and narratives |

Note that this includes conditions which will not be met: wanting to spawn does not coincide with the time span of mid-November to February!

### 2.7.7 Rule by Rule: Prey

Prey has been designed based on [15] both in terms of types as well as physiologically relevant properties. An overview is given in table 7.

To provide a realistic distribution of prey functional habitats will be populated with known likely composition of prey types. These will be replenished on demand ensuring diet composition observed in surveys.

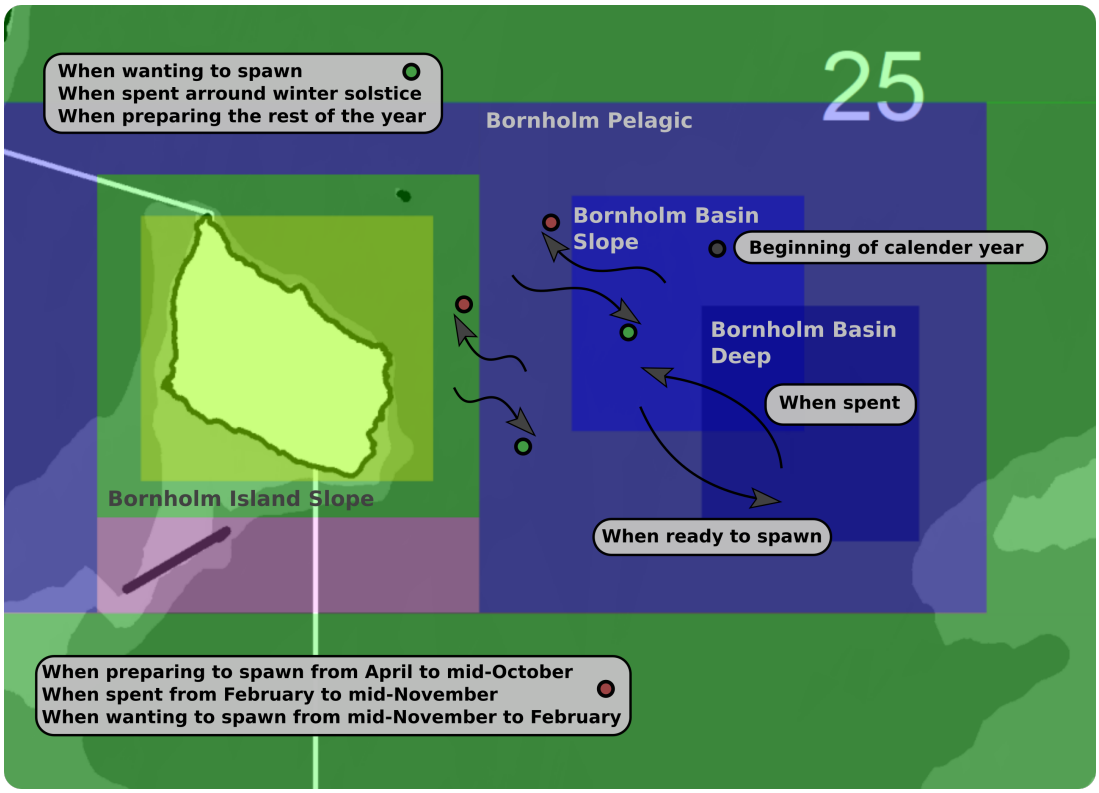



Figure 11: Overview of possible movements through the environmental boxes and the respective guards for these movements. Note that this includes conditions which will not be met: wanting to spawn does not coincide with the time span of mid-November to February!

**Replenish prey**

```
//P01 - 01 replenish prey - Bornholm Pelagic
Box(bn,t,o,sal,d,z,subs,pa      ,ar,mind,mand)[box?] ->
Box(bn,t,o,sal,d,z,subs,pa+2317,ar,mind,mand)[box?
+ 1743 Prey ('ringworm'  ,toInt(unif(1,2))      ,6,17,1    ,0  ,0)
+ 2    Prey ('crab'      ,toInt(unif(30,300))   ,6,17,1.23 ,0  ,0)
+ 22   Prey ('herring'   ,toInt(unif(115,300)) ,8,24,1.13 ,0
    ,paraass(lop))
+ 253  Prey ('crangon'   ,toInt(unif(38,300))  ,6,17,1    ,0
    ,paraass(lop))
+ 1    Prey ('cod'       ,toInt(unif(115,300)) ,5,14,1.45 ,0
    ,paraass(lop))
+ 23   Prey ('goby'      ,toInt(unif(115,300)) ,5,14,1.45 ,0
    ,paraass(lop))
+ 114  Prey ('mollusc'   ,toInt(unif(15,54))   ,6,17,1.23 ,0  ,0)
+ 93   Prey ('peracarida',toInt(unif(38,300))  ,6,17,1    ,0  ,0)
+ 14   Prey ('flatfish'  ,toInt(unif(115,300)) ,5,14,1.45 ,0
    ,paraass(lop))
+ 7    Prey ('isopod'    ,toInt(unif(15,300))  ,6,17,1    ,0
    ,paraass(lop))
+ 45   Prey ('sprat'     ,toInt(unif(115,300)) ,8,24,1.13 ,0
    ,paraass(lop))]
@ if (bn==1) && (count(box?,'Prey') <= 1)
then 1 else 0;
```

| | |
|---|---|
| Begin with: | Environmental box |
| End with: | Environmental box with increased prey, prey type distribution taken from [15] |
| Guard: | correct type of environmental box and prey is depleted |
| Rate: | Rate is 1 |

This rule formalises the distribution of prey types. By having prey be replenished only in the assumed proportions of prey types and only after it has been consumed entirely the observed stomach contents for different depth strata [15] can be reproduced. The guard for this rule is that it can only be applied to the correct box, controlling for depth strata, and when all prey is consumed disallowing overlaps. Please note that for some prey species the option of parasitation is included.

# 3 Data evaluation

---

This **TRACE** element provides supporting information on:
The quality and sources of numerical and qualitative data used to parametrise the model, both directly and inversely via calibration, and of the observed patterns that were used to design the overall model structure. This critical evaluation will allow model users to assess the scope and the uncertainty of the data and knowledge on which the model is based.

---

The quality of the input data has a broad span ranging from narratives (highly informed guesses) to precise high quality wet-lab and survey data. Again it should be noted that the model/submodels are not meant to provide quantitative results (although quantitative validation experiments should be undertaken frequently to keep the model grounded) but rather to grasp dynamic concepts and interdependencies.
All specific sources, rationales for educated guesses and procedures for calibrations are given with the description of each process in section 2 as part of the ODD protocol. The main data sources are also listed below:

Active Metabolic Rate
The rate for oxygen uptake is based on the findings of Claireaux et al. [9], who determined the active metabolic rate of cod dependent on temperature and oxygen saturation of the surrounding water in wet-lab experiments. The function for the active metabolic rate is given as:

$$\mathrm{AMR}(\mathrm{T}, \mathrm{S}_{\mathrm{O}_2}) = (17.29\mathrm{T}^{-0.015\mathrm{T}+1.062} + 30.01) \cdot (1 - \mathrm{e}^{-0.035\mathrm{S}_{\mathrm{O}_2}+0.34}) \ \frac{\mathrm{mg\ O_2}}{\mathrm{h\ kg}}, \tag{13}$$

where T is the temperature in °C and $\mathrm{S}_{\mathrm{O}_2}$ is the oxygen saturation given in %.

Maximum Oxygen Content
The maximum oxygen content a modelled cod can have was derived from the combined findings of [7] and [8], where the former had determined the blood oxygen capacity dependent on body weight and the latter determined the blood volume of cod. It was found to be justifiable to set the the upper threshold for oxygen to 3mg.

Standard metabolic rate
Equation also taken from [9] wet-lab experiments

$$\mathrm{SMR} = 80.1(1 - \mathrm{e}^{-0.185\mathrm{T}^{0.79}}) \ \frac{\mathrm{mg\ O_2}}{\mathrm{h\ kg}}, \tag{14}$$

where T is the temperature in °C.

'Joule per Oxygen'
JPO, the central energy unit was based on the findings of [16] and defined as $\mathbf{JPO} = 2.7112\frac{J}{2\mathbf{O}}$, meaning that there is a turnover of 2.7112 Joule per 2 **units** of oxygen which have been defined as 0.1mg.

Starvation Increment and Growth Increment

ToDo: Tabbelle mit parametern

# 4 Conceptual model evaluation

---

This **TRACE** element provides supporting information on:
The simplifying assumptions underlying a model's design, both with regard to empirical knowledge and general, basic principles. This critical evaluation allows model users to understand that model design was not ad hoc but based on carefully scrutinized considerations.

---

Overall updating the underlying understanding of the modelled system is the goal of this simulation study and the conceptual model therefore is undergoing constant updated and must therefore be viewed on a iteration-by-iteration basis.
For each process the underlying assumptions, subsequent simplifications and introduction of proxies is described in the ODD-protocol in section 2.

# 5 Implementation verification

---

This **TRACE** element provides supporting information on:
(1) Whether the computer code implementing the model has been thoroughly tested for programming errors, (2) whether the implemented model performs as indicated by the model description, and (3) how the software has been designed and documented to provide necessary usability tools (interfaces, automation of experiments, etc.) and to facilitate future installation, modification, and maintenance.

**1)** In the approach of this study model and simulator are explicitly separated and a formal modelling language is used. The modelling language is **ML-Rules** which is an external, stochastic, domain-specific modelling language developed for describing cell-biology at multiple levels of organizations, i.e., from intra-cellular, inter-cellular dynamics, to the dynamics of entire cell populations. ML-Rules has a well defined formal semantics [17] and its implementation of has been tested for programming errors. It has subsequently been used in several published modelling and simulation studies which have exposed it to continuous scrutiny [18, 19]
**2)** Based on the formal semantic of the modelling language the interpretation of the model code is unambiguous and transparent. In addition all models have been validated or analysed to confirm that they behave as would be expected from the model description. If further confirmation is required all tools and models are available for reproduction.
**3)** All models and experiments have been thoroughly tested for reproducibility and are available for use and augmentation. For instructions of software installation and use see the GIT repository. For experiments a binding to **SESSL**, an internal domain-specific DSL for describing and executing various simulation experiments specified in Scala [20] was used. Brief documentation is given in the models, experiments and scripts. Thorough documentation is given in this document. For instructions of software installation see the GIT repository. Full reproducibility is provided at `https://github.com/Baltic-Cod/EBC_IBM`. Here Software for the two DSL's all models and all experiments are available. Additionally R-Scripts which were used for evaluation and example results are provided.

# 6 Model output verification

This **TRACE** element provides supporting information on:
(1) How well model output matches observations and (2) how much calibration and effects of environmental drivers were involved in obtaining good fits of model output and data.

Continued validation of oxygen and temperature dependent physiological performance was ensured by successive validation ([6]) with recreated wet-lab experiments ([21], [11] and [13]).
In keeping with the model purpose (to be a 'thinking tool') calibration was used during model development. For these experiments strategic proxies (e.g.: food intake levels) were calibrated to match observations.
Environmental conditions were forced based on field-data and not calibrated.

ToDo: Model calibration: traditional parameter calibration
instead of additional parameter values we added, adapted and removed rules/processes (iterative expansion of precesses to fit to field data)

### 6.0.1 Experiments with Labyrinth_xxx.mlrj

These are simulation experiments conducted in the Bornholm Basin environment. The simplest version consist of four environmental boxes providing abiotic conditions and prey. To include more variation in abiotic conditions, the very large volumes were each fragmented into six and twelve environmental boxes with slight variations in temperature and oxygen saturations.

**1 year at liberty experiment in 4 boxes**

```
// Experiment for modelled cod physiology at liberty in 4 envrionmental boxes for one
    year (876000 time units) 25 pairs of male and female
// Model file: Labyrinth_4.mlrj"
// Output folder: result_labyrinth_4_year
// Duration example: 3h 23min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {
  model = "./Labyrinth_4.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1
  stopTime = 876000
  replications = 1

  observeAt(range(0,16800,876000))
  scan("group" <~ (
"GM(442,715,30,150,1,1,0,'E',0,13,1001)[Stc(0,1,1001)[] + Liver(0,0,1001) +
    Gon(1,0,0,0,0,1001)]+ GM(442,715,30,150,0,1,0,'E',0,13,1002)[Stc(0,1,1002)[] +
    Liver(0,0,1002) + Gon(0,0,0,0,0,1002)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1003)[Stc(0,1,1003)[] + Liver(0,0,1003) +
    Gon(1,0,0,0,0,1003)]+ GM(442,715,30,150,0,1,0,'E',0,13,1004)[Stc(0,1,1004)[] +
    Liver(0,0,1004) + Gon(0,0,0,0,0,1004)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1005)[Stc(0,1,1005)[] + Liver(0,0,1005) +
    Gon(1,0,0,0,0,1005)]+ GM(442,715,30,150,0,1,0,'E',0,13,1006)[Stc(0,1,1006)[] +
    Liver(0,0,1006) + Gon(0,0,0,0,0,1006)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1007)[Stc(0,1,1007)[] + Liver(0,0,1007) +
    Gon(1,0,0,0,0,1007)]+ GM(442,715,30,150,0,1,0,'E',0,13,1008)[Stc(0,1,1008)[] +
    Liver(0,0,1008) + Gon(0,0,0,0,0,1008)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1009)[Stc(0,1,1009)[] + Liver(0,0,1009) +
    Gon(1,0,0,0,0,1009)]+ GM(442,715,30,150,0,1,0,'E',0,13,1010)[Stc(0,1,1010)[] +
    Liver(0,0,1010) + Gon(0,0,0,0,0,1010)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1011)[Stc(0,1,1011)[] + Liver(0,0,1011) +
    Gon(1,0,0,0,0,1011)]+ GM(442,715,30,150,0,1,0,'E',0,13,1012)[Stc(0,1,1012)[] +
    Liver(0,0,1012) + Gon(0,0,0,0,0,1012)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1013)[Stc(0,1,1013)[] + Liver(0,0,1013) +
    Gon(1,0,0,0,0,1013)]+ GM(442,715,30,150,0,1,0,'E',0,13,1014)[Stc(0,1,1014)[] +
    Liver(0,0,1014) + Gon(0,0,0,0,0,1014)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1015)[Stc(0,1,1015)[] + Liver(0,0,1015) +
    Gon(1,0,0,0,0,1015)]+ GM(442,715,30,150,0,1,0,'E',0,13,1016)[Stc(0,1,1016)[] +
    Liver(0,0,1016) + Gon(0,0,0,0,0,1016)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1017)[Stc(0,1,1017)[] + Liver(0,0,1017) +
    Gon(1,0,0,0,0,1017)]+ GM(442,715,30,150,0,1,0,'E',0,13,1018)[Stc(0,1,1018)[] +
    Liver(0,0,1018) + Gon(0,0,0,0,0,1018)]"
```

```
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1019)[Stc(0,1,1019)[] + Liver(0,0,1019) +
    Gon(1,0,0,0,0,1019)]+ GM(442,715,30,150,0,1,0,'E',0,13,1020)[Stc(0,1,1020)[] +
    Liver(0,0,1020) + Gon(0,0,0,0,0,1020)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1021)[Stc(0,1,1021)[] + Liver(0,0,1021) +
    Gon(1,0,0,0,0,1021)]+ GM(442,715,30,150,0,1,0,'E',0,13,1022)[Stc(0,1,1022)[] +
    Liver(0,0,1022) + Gon(0,0,0,0,0,1022)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1023)[Stc(0,1,1023)[] + Liver(0,0,1023) +
    Gon(1,0,0,0,0,1023)]+ GM(442,715,30,150,0,1,0,'E',0,13,1024)[Stc(0,1,1024)[] +
    Liver(0,0,1024) + Gon(0,0,0,0,0,1024)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1025)[Stc(0,1,1025)[] + Liver(0,0,1025) +
    Gon(1,0,0,0,0,1025)]+ GM(442,715,30,150,0,1,0,'E',0,13,1026)[Stc(0,1,1026)[] +
    Liver(0,0,1026) + Gon(0,0,0,0,0,1026)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1027)[Stc(0,1,1027)[] + Liver(0,0,1027) +
    Gon(1,0,0,0,0,1027)]+ GM(442,715,30,150,0,1,0,'E',0,13,1028)[Stc(0,1,1028)[] +
    Liver(0,0,1028) + Gon(0,0,0,0,0,1028)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1029)[Stc(0,1,1029)[] + Liver(0,0,1029) +
    Gon(1,0,0,0,0,1029)]+ GM(442,715,30,150,0,1,0,'E',0,13,1030)[Stc(0,1,1030)[] +
    Liver(0,0,1030) + Gon(0,0,0,0,0,1030)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1031)[Stc(0,1,1031)[] + Liver(0,0,1031) +
    Gon(1,0,0,0,0,1031)]+ GM(442,715,30,150,0,1,0,'E',0,13,1032)[Stc(0,1,1032)[] +
    Liver(0,0,1032) + Gon(0,0,0,0,0,1032)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1033)[Stc(0,1,1033)[] + Liver(0,0,1033) +
    Gon(1,0,0,0,0,1033)]+ GM(442,715,30,150,0,1,0,'E',0,13,1034)[Stc(0,1,1034)[] +
    Liver(0,0,1034) + Gon(0,0,0,0,0,1034)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1035)[Stc(0,1,1035)[] + Liver(0,0,1035) +
    Gon(1,0,0,0,0,1035)]+ GM(442,715,30,150,0,1,0,'E',0,13,1036)[Stc(0,1,1036)[] +
    Liver(0,0,1036) + Gon(0,0,0,0,0,1036)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1037)[Stc(0,1,1037)[] + Liver(0,0,1037) +
    Gon(1,0,0,0,0,1037)]+ GM(442,715,30,150,0,1,0,'E',0,13,1038)[Stc(0,1,1038)[] +
    Liver(0,0,1038) + Gon(0,0,0,0,0,1038)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1039)[Stc(0,1,1039)[] + Liver(0,0,1039) +
    Gon(1,0,0,0,0,1039)]+ GM(442,715,30,150,0,1,0,'E',0,13,1040)[Stc(0,1,1040)[] +
    Liver(0,0,1040) + Gon(0,0,0,0,0,1040)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1041)[Stc(0,1,1041)[] + Liver(0,0,1041) +
    Gon(1,0,0,0,0,1041)]+ GM(442,715,30,150,0,1,0,'E',0,13,1042)[Stc(0,1,1042)[] +
    Liver(0,0,1042) + Gon(0,0,0,0,0,1042)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1043)[Stc(0,1,1043)[] + Liver(0,0,1043) +
    Gon(1,0,0,0,0,1043)]+ GM(442,715,30,150,0,1,0,'E',0,13,1044)[Stc(0,1,1044)[] +
    Liver(0,0,1044) + Gon(0,0,0,0,0,1044)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1045)[Stc(0,1,1045)[] + Liver(0,0,1045) +
    Gon(1,0,0,0,0,1045)]+ GM(442,715,30,150,0,1,0,'E',0,13,1046)[Stc(0,1,1046)[] +
    Liver(0,0,1046) + Gon(0,0,0,0,0,1046)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1047)[Stc(0,1,1047)[] + Liver(0,0,1047) +
    Gon(1,0,0,0,0,1047)]+ GM(442,715,30,150,0,1,0,'E',0,13,1048)[Stc(0,1,1048)[] +
    Liver(0,0,1048) + Gon(0,0,0,0,0,1048)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1049)[Stc(0,1,1049)[] + Liver(0,0,1049) +
    Gon(1,0,0,0,0,1049)]+ GM(442,715,30,150,0,1,0,'E',0,13,1050)[Stc(0,1,1050)[] +
    Liver(0,0,1050) + Gon(0,0,0,0,0,1050)]"))
```

```
observe("boxnumber" ~ attribute("*/Box",0))
observe("temperature" ~ attribute("*/Box",1))
observe("oxygen" ~ attribute("*/Box",2))
observe("day" ~ attribute("*/Box",4))
observe("prey_availability" ~ attribute("*/Box",7))

observe("length" ~ attribute("*/GM",0))
observe("mass" ~ attribute("*/GM",1))
observe("sex" ~ attribute("*/GM",4))
observe("rep_cyc" ~ attribute("*/GM",5))
observe("res_box" ~ attribute("*/GM",9))
observe("identifyer" ~ attribute("*/GM",10))

observe("stc_prey_count" ~ attribute("*/Stc",0))
observe("stc_variable" ~ attribute("*/Stc",1))
observe("stc_identifyer" ~ attribute("*/Stc",2))

observe("liver_co" ~ attribute("*/Liver",1))
observe("liver_identifyer" ~ attribute("*/Liver",2))

observe("sex_gon" ~ attribute("*/Gon",0))
observe("eggnumber" ~ attribute("*/Gon",1))
observe("gonad_maturity" ~ attribute("*/Gon",2))
observe("JPO_egg_counter" ~ attribute("*/Gon",3))
observe("gonad_weight" ~ attribute("*/Gon",4))
observe("gonad_identifyer" ~ attribute("*/Gon",5))

observe("Eggs" ~ count("*/GME"))

observe("Swimming" ~ count("*/ACT"))
observe("Hunting" ~ count("*/HUNT"))
observe("Eating" ~ count("*/EAT"))
observe("Mating" ~ count("*/MATE"))

csvOutputDirectory(() => "result_labyrinth_4_year")
withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Labyrinth_4.mlrj |
| Duration: | 876000TU $\hat{=}$ 12 weeks |
| Scanning for: | 25 pairs (female and male) of cod |
| Monitoring: | boxes and their temperature, oxygen saturation, day of year and prey availability; cod and their length, mass, sex, status of reproductive cycle, residing box and unique identifier; stomach and their prey count feed intake level status, unique identifier; liver with *Contracaecum osculatum* load and unique identifier; gonads with sex, amount of eggs, gonad maturity, gonad energy count, gonad weight, unique identifier; eggs in the water; entity counters for: swimming, hunting, eating and mating |
| Interval: | every 16800TU $\hat{=}$ every week |
| Replications | 1 - (note: each pair of cod can be understood as a replication) |

**1 year at liberty experiment in 4 boxes at liberty with uptake of liver parasite *Contracaecum osculatum* based on ingestion**

```
// Experiment for modelled cod physiology at liberty in 4 envrionmental boxes for one
    year (876000 time units) 25 pairs of male and female
// Model file: Labyrinth_4_pp_liver.mlrj"
// Output folder: result_labyrinth_4_pp_liver_year
// Duration example: 9h 26min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {
  model = "./Labyrinth_pp_liver_4.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1
  stopTime = 876000
  replications = 1

  observeAt(range(0,16800,876000))
  scan("lop" <~ (0,0.2,0.8,1))
  scan("group" <~ (
"GM(442,715,30,150,1,1,0,'E',0,13,1001)[Stc(0,1,1001)[] + Liver(0,0,1001) +
    Gon(1,0,0,0,0,1001)]+ GM(442,715,30,150,0,1,0,'E',0,13,1002)[Stc(0,1,1002)[] +
    Liver(0,0,1002) + Gon(0,0,0,0,0,1002)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1003)[Stc(0,1,1003)[] + Liver(0,0,1003) +
    Gon(1,0,0,0,0,1003)]+ GM(442,715,30,150,0,1,0,'E',0,13,1004)[Stc(0,1,1004)[] +
    Liver(0,0,1004) + Gon(0,0,0,0,0,1004)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1005)[Stc(0,1,1005)[] + Liver(0,0,1005) +
    Gon(1,0,0,0,0,1005)]+ GM(442,715,30,150,0,1,0,'E',0,13,1006)[Stc(0,1,1006)[] +
    Liver(0,0,1006) + Gon(0,0,0,0,0,1006)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1007)[Stc(0,1,1007)[] + Liver(0,0,1007) +
    Gon(1,0,0,0,0,1007)]+ GM(442,715,30,150,0,1,0,'E',0,13,1008)[Stc(0,1,1008)[] +
    Liver(0,0,1008) + Gon(0,0,0,0,0,1008)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1009)[Stc(0,1,1009)[] + Liver(0,0,1009) +
    Gon(1,0,0,0,0,1009)]+ GM(442,715,30,150,0,1,0,'E',0,13,1010)[Stc(0,1,1010)[] +
    Liver(0,0,1010) + Gon(0,0,0,0,0,1010)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1011)[Stc(0,1,1011)[] + Liver(0,0,1011) +
    Gon(1,0,0,0,0,1011)]+ GM(442,715,30,150,0,1,0,'E',0,13,1012)[Stc(0,1,1012)[] +
    Liver(0,0,1012) + Gon(0,0,0,0,0,1012)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1013)[Stc(0,1,1013)[] + Liver(0,0,1013) +
    Gon(1,0,0,0,0,1013)]+ GM(442,715,30,150,0,1,0,'E',0,13,1014)[Stc(0,1,1014)[] +
    Liver(0,0,1014) + Gon(0,0,0,0,0,1014)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1015)[Stc(0,1,1015)[] + Liver(0,0,1015) +
    Gon(1,0,0,0,0,1015)]+ GM(442,715,30,150,0,1,0,'E',0,13,1016)[Stc(0,1,1016)[] +
    Liver(0,0,1016) + Gon(0,0,0,0,0,1016)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1017)[Stc(0,1,1017)[] + Liver(0,0,1017) +
    Gon(1,0,0,0,0,1017)]+ GM(442,715,30,150,0,1,0,'E',0,13,1018)[Stc(0,1,1018)[] +
    Liver(0,0,1018) + Gon(0,0,0,0,0,1018)]"
```

```
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1019)[Stc(0,1,1019)[] + Liver(0,0,1019) +
    Gon(1,0,0,0,0,1019)]+ GM(442,715,30,150,0,1,0,'E',0,13,1020)[Stc(0,1,1020)[] +
    Liver(0,0,1020) + Gon(0,0,0,0,0,1020)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1021)[Stc(0,1,1021)[] + Liver(0,0,1021) +
    Gon(1,0,0,0,0,1021)]+ GM(442,715,30,150,0,1,0,'E',0,13,1022)[Stc(0,1,1022)[] +
    Liver(0,0,1022) + Gon(0,0,0,0,0,1022)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1023)[Stc(0,1,1023)[] + Liver(0,0,1023) +
    Gon(1,0,0,0,0,1023)]+ GM(442,715,30,150,0,1,0,'E',0,13,1024)[Stc(0,1,1024)[] +
    Liver(0,0,1024) + Gon(0,0,0,0,0,1024)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1025)[Stc(0,1,1025)[] + Liver(0,0,1025) +
    Gon(1,0,0,0,0,1025)]+ GM(442,715,30,150,0,1,0,'E',0,13,1026)[Stc(0,1,1026)[] +
    Liver(0,0,1026) + Gon(0,0,0,0,0,1026)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1027)[Stc(0,1,1027)[] + Liver(0,0,1027) +
    Gon(1,0,0,0,0,1027)]+ GM(442,715,30,150,0,1,0,'E',0,13,1028)[Stc(0,1,1028)[] +
    Liver(0,0,1028) + Gon(0,0,0,0,0,1028)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1029)[Stc(0,1,1029)[] + Liver(0,0,1029) +
    Gon(1,0,0,0,0,1029)]+ GM(442,715,30,150,0,1,0,'E',0,13,1030)[Stc(0,1,1030)[] +
    Liver(0,0,1030) + Gon(0,0,0,0,0,1030)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1031)[Stc(0,1,1031)[] + Liver(0,0,1031) +
    Gon(1,0,0,0,0,1031)]+ GM(442,715,30,150,0,1,0,'E',0,13,1032)[Stc(0,1,1032)[] +
    Liver(0,0,1032) + Gon(0,0,0,0,0,1032)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1033)[Stc(0,1,1033)[] + Liver(0,0,1033) +
    Gon(1,0,0,0,0,1033)]+ GM(442,715,30,150,0,1,0,'E',0,13,1034)[Stc(0,1,1034)[] +
    Liver(0,0,1034) + Gon(0,0,0,0,0,1034)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1035)[Stc(0,1,1035)[] + Liver(0,0,1035) +
    Gon(1,0,0,0,0,1035)]+ GM(442,715,30,150,0,1,0,'E',0,13,1036)[Stc(0,1,1036)[] +
    Liver(0,0,1036) + Gon(0,0,0,0,0,1036)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1037)[Stc(0,1,1037)[] + Liver(0,0,1037) +
    Gon(1,0,0,0,0,1037)]+ GM(442,715,30,150,0,1,0,'E',0,13,1038)[Stc(0,1,1038)[] +
    Liver(0,0,1038) + Gon(0,0,0,0,0,1038)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1039)[Stc(0,1,1039)[] + Liver(0,0,1039) +
    Gon(1,0,0,0,0,1039)]+ GM(442,715,30,150,0,1,0,'E',0,13,1040)[Stc(0,1,1040)[] +
    Liver(0,0,1040) + Gon(0,0,0,0,0,1040)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1041)[Stc(0,1,1041)[] + Liver(0,0,1041) +
    Gon(1,0,0,0,0,1041)]+ GM(442,715,30,150,0,1,0,'E',0,13,1042)[Stc(0,1,1042)[] +
    Liver(0,0,1042) + Gon(0,0,0,0,0,1042)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1043)[Stc(0,1,1043)[] + Liver(0,0,1043) +
    Gon(1,0,0,0,0,1043)]+ GM(442,715,30,150,0,1,0,'E',0,13,1044)[Stc(0,1,1044)[] +
    Liver(0,0,1044) + Gon(0,0,0,0,0,1044)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1045)[Stc(0,1,1045)[] + Liver(0,0,1045) +
    Gon(1,0,0,0,0,1045)]+ GM(442,715,30,150,0,1,0,'E',0,13,1046)[Stc(0,1,1046)[] +
    Liver(0,0,1046) + Gon(0,0,0,0,0,1046)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1047)[Stc(0,1,1047)[] + Liver(0,0,1047) +
    Gon(1,0,0,0,0,1047)]+ GM(442,715,30,150,0,1,0,'E',0,13,1048)[Stc(0,1,1048)[] +
    Liver(0,0,1048) + Gon(0,0,0,0,0,1048)]"
,"1 GM(442,715,30,150,1,1,0,'E',0,13,1049)[Stc(0,1,1049)[] + Liver(0,0,1049) +
    Gon(1,0,0,0,0,1049)]+ GM(442,715,30,150,0,1,0,'E',0,13,1050)[Stc(0,1,1050)[] +
    Liver(0,0,1050) + Gon(0,0,0,0,0,1050)]" ))
```

```
observe("boxnumber" ~ attribute("*/Box",0))
observe("temperature" ~ attribute("*/Box",1))
observe("oxygen" ~ attribute("*/Box",2))
observe("day" ~ attribute("*/Box",4))
observe("prey_availability" ~ attribute("*/Box",7))

observe("length" ~ attribute("*/GM",0))
observe("mass" ~ attribute("*/GM",1))
observe("sex" ~ attribute("*/GM",4))
observe("rep_cyc" ~ attribute("*/GM",5))
observe("res_box" ~ attribute("*/GM",9))
observe("identifyer" ~ attribute("*/GM",10))

observe("stc_prey_count" ~ attribute("*/Stc",0))
observe("stc_variable" ~ attribute("*/Stc",1))
observe("stc_identifyer" ~ attribute("*/Stc",2))

observe("liver_co" ~ attribute("*/Liver",1))
observe("liver_identifyer" ~ attribute("*/Liver",2))

observe("sex_gon" ~ attribute("*/Gon",0))
observe("eggnumber" ~ attribute("*/Gon",1))
observe("gonad_maturity" ~ attribute("*/Gon",2))
observe("JPO_egg_counter" ~ attribute("*/Gon",3))
observe("gonad_weight" ~ attribute("*/Gon",4))
observe("gonad_identifyer" ~ attribute("*/Gon",5))

observe("Eggs" ~ count("*/GME"))

observe("Swimming" ~ count("*/ACT"))
observe("Hunting" ~ count("*/HUNT"))
observe("Eating" ~ count("*/EAT"))
observe("Mating" ~ count("*/MATE"))

csvOutputDirectory(() => "result_labyrinth_4_pp_liver_year")
withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Labyrinth_pp_liver_4.mlrj |
| Duration: | 876000TU $\hat{=}$ 12 weeks |
| Scanning for: | 25 pairs (female and male) of cod and level of parasitation of prey (lop) (0,0.2,0.8,1)) |
| Monitoring: | boxes and their temperature, oxygen saturation, day of year and prey availability; cod and their length, mass, sex, status of reproductive cycle, residing box and unique identifier; stomach and their prey count feed intake level status, unique identifier; liver with *Contracaecum osculatum* load and unique identifier; gonads with sex, amount of eggs, gonad maturity, gonad energy count, gonad weight, unique identifier; eggs in the water; entity counters for: swimming, hunting, eating and mating |
| Interval: | every 16800TU $\hat{=}$ every week |
| Replications | 1 - (note: each pair of cod can be understood as a replication) |

### Condition at liberty

```
// Experiment for modelled cod physiology at liberty in 4 envrionmental boxes for one
    year (876000 time units) 25 pairs of male and female parasitation based on
    condition
// Model file: Labyrinth_4_pp_cond.mlrj"
// Output folder: result_labyrinth_4_pp_cond_year
// Duration example: 11h 19min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {

  model = "./Labyrinth_4_pp_cond.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1
  stopTime = 876000
  replications = 1

  observeAt(range(0,16800,876000))

  scan("lop" <~ (0,0.2,0.8,1))

  scan("group" <~ (
"GM(442,605,30,150,1,1,0,'E',0,13,1001)[Stc(0,1,1001)[] + Liver(0,0,1001) +
    Gon(1,0,0,0,0,1001)]+ GM(442,605,30,150,0,1,0,'E',0,13,1002)[Stc(0,1,1002)[] +
    Liver(0,0,1002) + Gon(0,0,0,0,0,1002)]"
,"1 GM(442,605,30,150,1,1,0,'E',0,13,1003)[Stc(0,1,1003)[] + Liver(0,0,1003) +
    Gon(1,0,0,0,0,1003)]+ GM(442,605,30,150,0,1,0,'E',0,13,1004)[Stc(0,1,1004)[] +
    Liver(0,0,1004) + Gon(0,0,0,0,0,1004)]"
,"1 GM(442,605,30,150,1,1,0,'E',0,13,1005)[Stc(0,1,1005)[] + Liver(0,0,1005) +
    Gon(1,0,0,0,0,1005)]+ GM(442,605,30,150,0,1,0,'E',0,13,1006)[Stc(0,1,1006)[] +
    Liver(0,0,1006) + Gon(0,0,0,0,0,1006)]"
,"1 GM(442,605,30,150,1,1,0,'E',0,13,1007)[Stc(0,1,1007)[] + Liver(0,0,1007) +
    Gon(1,0,0,0,0,1007)]+ GM(442,605,30,150,0,1,0,'E',0,13,1008)[Stc(0,1,1008)[] +
    Liver(0,0,1008) + Gon(0,0,0,0,0,1008)]"
,"1 GM(442,605,30,150,1,1,0,'E',0,13,1009)[Stc(0,1,1009)[] + Liver(0,0,1009) +
    Gon(1,0,0,0,0,1009)]+ GM(442,605,30,150,0,1,0,'E',0,13,1010)[Stc(0,1,1010)[] +
    Liver(0,0,1010) + Gon(0,0,0,0,0,1010)]"
,"1 GM(442,691,30,150,1,1,0,'E',0,13,1011)[Stc(0,1,1011)[] + Liver(0,0,1011) +
    Gon(1,0,0,0,0,1011)]+ GM(442,691,30,150,0,1,0,'E',0,13,1012)[Stc(0,1,1012)[] +
    Liver(0,0,1012) + Gon(0,0,0,0,0,1012)]"
,"1 GM(442,691,30,150,1,1,0,'E',0,13,1013)[Stc(0,1,1013)[] + Liver(0,0,1013) +
    Gon(1,0,0,0,0,1013)]+ GM(442,691,30,150,0,1,0,'E',0,13,1014)[Stc(0,1,1014)[] +
    Liver(0,0,1014) + Gon(0,0,0,0,0,1014)]"
,"1 GM(442,691,30,150,1,1,0,'E',0,13,1015)[Stc(0,1,1015)[] + Liver(0,0,1015) +
    Gon(1,0,0,0,0,1015)]+ GM(442,691,30,150,0,1,0,'E',0,13,1016)[Stc(0,1,1016)[] +
    Liver(0,0,1016) + Gon(0,0,0,0,0,1016)]"
,"1 GM(442,691,30,150,1,1,0,'E',0,13,1017)[Stc(0,1,1017)[] + Liver(0,0,1017) +
    Gon(1,0,0,0,0,1017)]+ GM(442,691,30,150,0,1,0,'E',0,13,1018)[Stc(0,1,1018)[] +
    Liver(0,0,1018) + Gon(0,0,0,0,0,1018)]"
```

```
,"1 GM(442,691,30,150,1,1,0,'E',0,13,1019)[Stc(0,1,1019)[] + Liver(0,0,1019) +
    Gon(1,0,0,0,0,1019)]+ GM(442,691,30,150,0,1,0,'E',0,13,1020)[Stc(0,1,1020)[] +
    Liver(0,0,1020) + Gon(0,0,0,0,0,1020)]"
,"1 GM(442,777,30,150,1,1,0,'E',0,13,1021)[Stc(0,1,1021)[] + Liver(0,0,1021) +
    Gon(1,0,0,0,0,1021)]+ GM(442,777,30,150,0,1,0,'E',0,13,1022)[Stc(0,1,1022)[] +
    Liver(0,0,1022) + Gon(0,0,0,0,0,1022)]"
,"1 GM(442,777,30,150,1,1,0,'E',0,13,1023)[Stc(0,1,1023)[] + Liver(0,0,1023) +
    Gon(1,0,0,0,0,1023)]+ GM(442,777,30,150,0,1,0,'E',0,13,1024)[Stc(0,1,1024)[] +
    Liver(0,0,1024) + Gon(0,0,0,0,0,1024)]"
,"1 GM(442,777,30,150,1,1,0,'E',0,13,1025)[Stc(0,1,1025)[] + Liver(0,0,1025) +
    Gon(1,0,0,0,0,1025)]+ GM(442,777,30,150,0,1,0,'E',0,13,1026)[Stc(0,1,1026)[] +
    Liver(0,0,1026) + Gon(0,0,0,0,0,1026)]"
,"1 GM(442,777,30,150,1,1,0,'E',0,13,1027)[Stc(0,1,1027)[] + Liver(0,0,1027) +
    Gon(1,0,0,0,0,1027)]+ GM(442,777,30,150,0,1,0,'E',0,13,1028)[Stc(0,1,1028)[] +
    Liver(0,0,1028) + Gon(0,0,0,0,0,1028)]"
,"1 GM(442,777,30,150,1,1,0,'E',0,13,1029)[Stc(0,1,1029)[] + Liver(0,0,1029) +
    Gon(1,0,0,0,0,1029)]+ GM(442,777,30,150,0,1,0,'E',0,13,1030)[Stc(0,1,1030)[] +
    Liver(0,0,1030) + Gon(0,0,0,0,0,1030)]"
,"1 GM(442,864,30,150,1,1,0,'E',0,13,1031)[Stc(0,1,1031)[] + Liver(0,0,1031) +
    Gon(1,0,0,0,0,1031)]+ GM(442,864,30,150,0,1,0,'E',0,13,1032)[Stc(0,1,1032)[] +
    Liver(0,0,1032) + Gon(0,0,0,0,0,1032)]"
,"1 GM(442,864,30,150,1,1,0,'E',0,13,1033)[Stc(0,1,1033)[] + Liver(0,0,1033) +
    Gon(1,0,0,0,0,1033)]+ GM(442,864,30,150,0,1,0,'E',0,13,1034)[Stc(0,1,1034)[] +
    Liver(0,0,1034) + Gon(0,0,0,0,0,1034)]"
,"1 GM(442,864,30,150,1,1,0,'E',0,13,1035)[Stc(0,1,1035)[] + Liver(0,0,1035) +
    Gon(1,0,0,0,0,1035)]+ GM(442,864,30,150,0,1,0,'E',0,13,1036)[Stc(0,1,1036)[] +
    Liver(0,0,1036) + Gon(0,0,0,0,0,1036)]"
,"1 GM(442,864,30,150,1,1,0,'E',0,13,1037)[Stc(0,1,1037)[] + Liver(0,0,1037) +
    Gon(1,0,0,0,0,1037)]+ GM(442,864,30,150,0,1,0,'E',0,13,1038)[Stc(0,1,1038)[] +
    Liver(0,0,1038) + Gon(0,0,0,0,0,1038)]"
,"1 GM(442,864,30,150,1,1,0,'E',0,13,1039)[Stc(0,1,1039)[] + Liver(0,0,1039) +
    Gon(1,0,0,0,0,1039)]+ GM(442,864,30,150,0,1,0,'E',0,13,1040)[Stc(0,1,1040)[] +
    Liver(0,0,1040) + Gon(0,0,0,0,0,1040)]"
,"1 GM(442,950,30,150,1,1,0,'E',0,13,1041)[Stc(0,1,1041)[] + Liver(0,0,1041) +
    Gon(1,0,0,0,0,1041)]+ GM(442,950,30,150,0,1,0,'E',0,13,1042)[Stc(0,1,1042)[] +
    Liver(0,0,1042) + Gon(0,0,0,0,0,1042)]"
,"1 GM(442,950,30,150,1,1,0,'E',0,13,1043)[Stc(0,1,1043)[] + Liver(0,0,1043) +
    Gon(1,0,0,0,0,1043)]+ GM(442,950,30,150,0,1,0,'E',0,13,1044)[Stc(0,1,1044)[] +
    Liver(0,0,1044) + Gon(0,0,0,0,0,1044)]"
,"1 GM(442,950,30,150,1,1,0,'E',0,13,1045)[Stc(0,1,1045)[] + Liver(0,0,1045) +
    Gon(1,0,0,0,0,1045)]+ GM(442,950,30,150,0,1,0,'E',0,13,1046)[Stc(0,1,1046)[] +
    Liver(0,0,1046) + Gon(0,0,0,0,0,1046)]"
,"1 GM(442,950,30,150,1,1,0,'E',0,13,1047)[Stc(0,1,1047)[] + Liver(0,0,1047) +
    Gon(1,0,0,0,0,1047)]+ GM(442,950,30,150,0,1,0,'E',0,13,1048)[Stc(0,1,1048)[] +
    Liver(0,0,1048) + Gon(0,0,0,0,0,1048)]"
,"1 GM(442,950,30,150,1,1,0,'E',0,13,1049)[Stc(0,1,1049)[] + Liver(0,0,1049) +
    Gon(1,0,0,0,0,1049)]+ GM(442,950,30,150,0,1,0,'E',0,13,1050)[Stc(0,1,1050)[] +
    Liver(0,0,1050) + Gon(0,0,0,0,0,1050)]" ))
```

```
observe("boxnumber" ~ attribute("*/Box",0))
observe("temperature" ~ attribute("*/Box",1))
observe("oxygen" ~ attribute("*/Box",2))
observe("day" ~ attribute("*/Box",4))
observe("prey_availability" ~ attribute("*/Box",7))

observe("length" ~ attribute("*/GM",0))
observe("mass" ~ attribute("*/GM",1))
observe("sex" ~ attribute("*/GM",4))
observe("rep_cyc" ~ attribute("*/GM",5))
observe("res_box" ~ attribute("*/GM",9))
observe("identifyer" ~ attribute("*/GM",10))

observe("stc_prey_count" ~ attribute("*/Stc",0))
observe("stc_variable" ~ attribute("*/Stc",1))
observe("stc_identifyer" ~ attribute("*/Stc",2))

observe("liver_co" ~ attribute("*/Liver",1))
observe("liver_identifyer" ~ attribute("*/Liver",2))

observe("sex_gon" ~ attribute("*/Gon",0))
observe("eggnumber" ~ attribute("*/Gon",1))
observe("gonad_maturity" ~ attribute("*/Gon",2))
observe("JPO_egg_counter" ~ attribute("*/Gon",3))
observe("gonad_weight" ~ attribute("*/Gon",4))
observe("gonad_identifyer" ~ attribute("*/Gon",5))

observe("Eggs" ~ count("*/GME"))

observe("Swimming" ~ count("*/ACT"))
observe("Hunting" ~ count("*/HUNT"))
observe("Eating" ~ count("*/EAT"))
observe("Mating" ~ count("*/MATE"))

csvOutputDirectory(() => "result_labyrinth_4_pp_cond_year")
withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Labyrinth_pp_cond_4.mlrj |
| Duration: | 876000TU $\hat{=}$ 12 weeks |
| Scanning for: | 25 pairs (female and male) of cod |
| Monitoring: | boxes and their temperature, oxygen saturation, day of year and prey availability; cod and their length, mass, sex, status of reproductive cycle, residing box and unique identifier; stomach and their prey count feed intake level status, unique identifier; liver with *Contracaecum osculatum* load and unique identifier; gonads with sex, amount of eggs, gonad maturity, gonad energy count, gonad weight, unique identifier; eggs in the water; entity counters for: swimming, hunting, eating and mating |
| Interval: | every 16800TU $\hat{=}$ every week |
| Replications | 1 - (note: each pair of cod can be understood as a replication) |

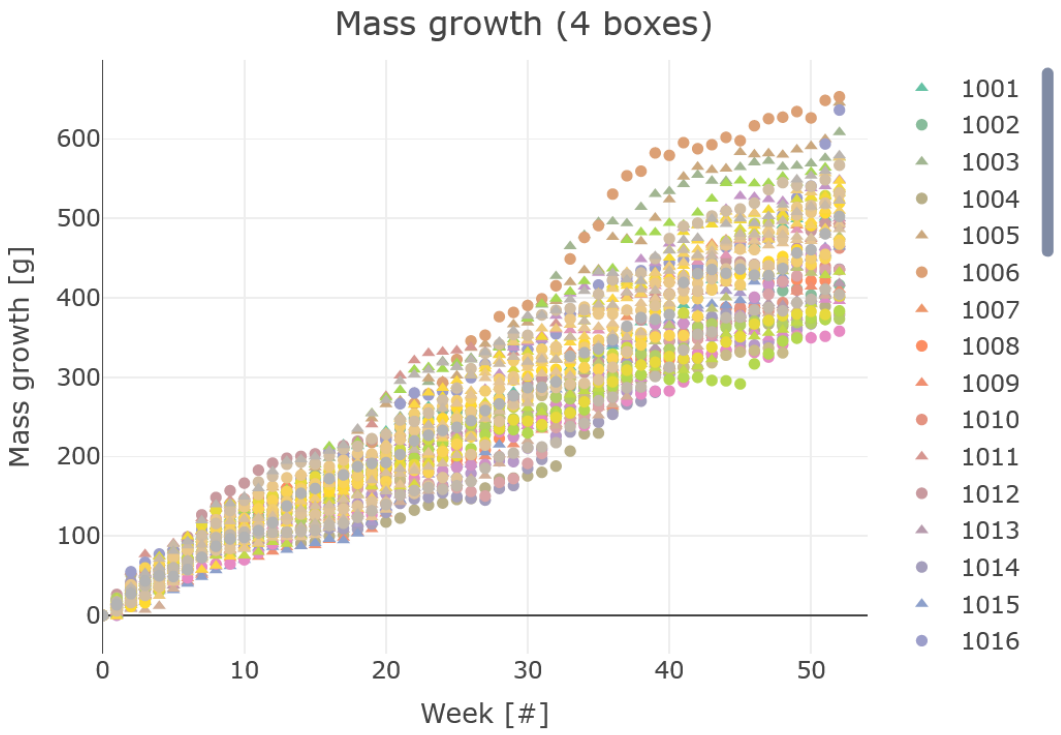



Figure 12

# 7 Model analysis

---

This **TRACE** element provides supporting information on:
(1) How sensitive model output is to changes in model parameters (sensitivity analysis), and (2) how well the emergence of model output has been understood.

---

The physiological models are sensitive, as validated, to oxygen saturation and temperature. Order of physiological priority both for oxygen and metabolite requirement has a high impact on physiological behaviour. In relation to oxygen, temperature and parasitation digested food currently has the highest impact on physiological performance in terms of growth (see section 2 - variable appetite).
During development factors with low or competing sensitivity were omitted, having low explanatory power.
Weak emergence could be observed with implementation of wet-lab results: using wet-lab results for metabolic scopes directly and without calibration produced valid behaviour for oxygen and temperature dependent growth.
The multi-level nature of the model, technically allows for strong emergence in model behaviour, but non has so far been identified.

ToDo: expand on parameter scans and visualisations

| Type | Name | O2 | JPO | Factor | Sizes [AJPO] |
|---|---|---|---|---|---|
| Annelida | 'ringworm' | 6 | 14 | 1 | 1-2 |
| Brachyura | 'crab' | 6 | 14 | 1.23 | 30-1460 |
| Clupea harengus | 'herring' | 8 | 20 | 1.13 | 115-2500 |
| Crangon crangon | 'crangon' | 6 | 14 | 1 | 38-354 |
| Gadus morhua | 'cod' | 5 | 12 | 1.45 | 115-2500 |
| Gobiidae | 'goby' | 5 | 12 | 1.45 | 115-2500 |
| Mollusca | 'mollusc' | 6 | 14 | 1.23 | 15-54 |
| Paracarida | 'peracarida' | 6 | 14 | 1 | 38-354 |
| Pleuronectiformes | 'flatfish' | 5 | 12 | 1.45 | 115-2500 |
| Saduria entomon | 'isopod' | 6 | 14 | 1 | 15-770 |
| Sprattus sprattus | 'sprat' | 8 | 20 | 1.13 | 115-1150 |

Table 7: Exploratory prey design. Sizes are limited to 2500 AJPO since cod used for experiment are not large enough to ingest prey larger than this. Properties based on [15]

# 8 Model output corroboration

---

This **TRACE** element provides supporting information on:
How model predictions compare to independent data and patterns that were not used, and preferably not even known, while the model was developed, parametrized, and verified. By documenting model output corroboration, model users learn about evidence which, in addition to model output verification, indicates that the model is structurally realistic so that its predictions can be trusted to some degree.

---

For the validated parts of the model (oxygen and temperature dependent physiological performance and hypoxia tolerance) independent data was used: [21], [11] and [13] for each behaviour respectively.
Further development of the model was explicitly aimed at speculation and no validity is claimed. However, taking into account the introduced suitable proxies and calibration some simulation results could be partially face validated by suitable domain experts.

ToDo: add validation experiments; Important predicting data that the model has not seen before

## 8.1 Experiments

All experiments, example results and visualisations can be found at `https://github.com/Baltic-Cod/EBC_IBM`

### 8.1.1 Experiments with Basic.mlrj

The Basic model contains an updated version of the juvenile physiology of modelled cod.

Expansion with the feeding regime is the Basic_ex model. The feeding regime offers items of prey for one hour each Monday, Wednesday and Friday.

**Asphyxiation experiment**

```
// Experiment for modelled cod physiology in a 4 day asphyxiation experiment at
    different weight classes
// Model file: Basic_asphyx.mlrj
// Output folder: result_basic_asphyx
// Duration example: 3 min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {
  model = "./Basic_asphyx.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1

  stopTime = 9600
  replications = 1

  observeAt(range(0, 9600, 9600))
  scan("init" <~ (
"20 GM(385,570 ,30,50,1,0,1,'E',0,1,1001)[Stc(0,1,1001)[]]",
"20 GM(560,1740,30,50,1,0,1,'E',0,1,1001)[Stc(0,1,1001)[]]",
"20 GM(445,890 ,30,50,1,0,1,'E',0,1,1001)[Stc(0,1,1001)[]]",
"20 GM(560,1790,30,50,1,0,1,'E',0,1,1001)[Stc(0,1,1001)[]]"))
scan("o" <~ (13.8,17.8,23.7,29.5,36.5,42.5))
scan("t" <~ (2,6))

observe("N_GM" ~ count("GM"))

csvOutputDirectory(() => "result_basic_asphyx")
withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Basic_asphyx.mlrj |
| Duration: | 9600TU $\hat{=}$ 4 days |
| Scanning for: | weight(570,1740,890,1790), oxygen(13.8,17.8,23.7,29.5,36.5,42.5), temperature(2,6) |
| Monitoring: | survival |
| Interval: | Observation only at end (9600TU $\hat{=}$ 4 days) |
| Replications | 1 (note: 20 animals are present per treatment) |

To validate the physiological response to persistent low levels of oxygen the wet-lab findings of [13] where reproduced in a 4 day asphyxiation experiment. Both real and simulated cod of different weight classes where kept at different temperatures and oxygen saturation in groups of twenty animals and the the resulting cumulative mortality was calculated. the simulated response of the updated model was, again, more regular than wet-lab observations but showed the same tipping point from 100% to 0% mortality (20 to 30 % oxygen saturation). When not aiming to gain insight on this exact threshold the model can therefore be deemed valid concerning death from asphyxiation.

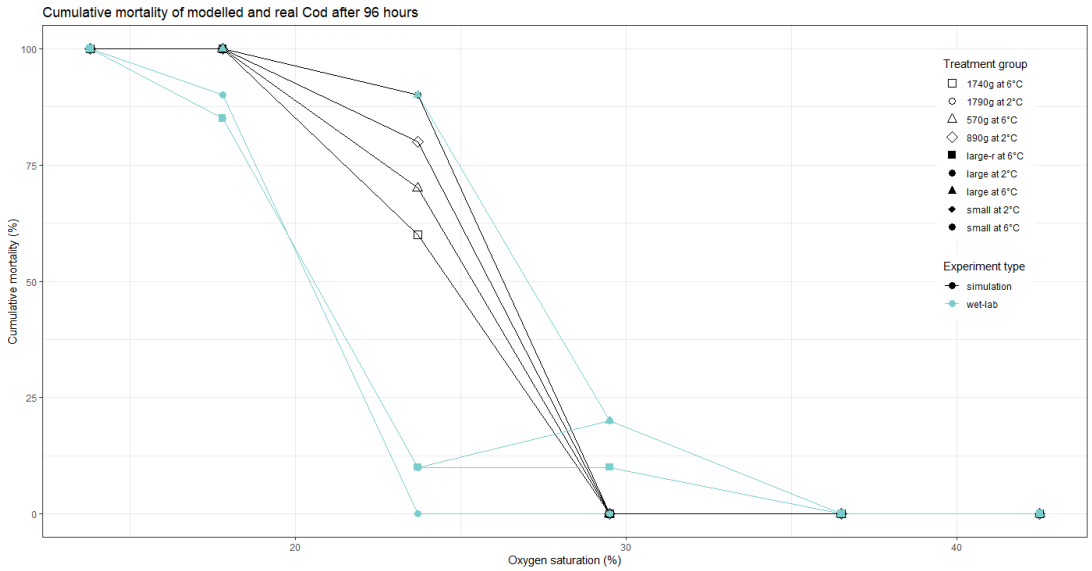



Figure 13: Results of a 4-day asphyxiation experiment comparing wet-lab ([13]) and simulation results.

**12 week growth experiment**

```
// Experiment for modelled cod physiology in a 12 week feeding experiment at different
    oxygen levels
// Model file: Basic.mlrj
// Output folder: result_basic_ex
// Duration example: 11 min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {
  model = "./Basic_ex.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1

  stopTime = 201600
  replications = 20

  scan("o" <~ (45,56,65,75,84,93))
  scan("t" <~ (10))

  observeAt(range(0, 201600, 201600))
  observe("length" ~ attribute("*/GM",0))
  observe("weight" ~ attribute("*/GM",1))

  csvOutputDirectory(() => "result_basic_ex")

  withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Basic_ex.mlrj |
| Duration: | 201600TU $\hat{=}$ 12 weeks |
| Scanning for: | oxygen (45,56,65,75,84,93) |
| Monitoring: | length and weight |
| Interval: | Observation only at end (201600TU $\hat{=}$ 12 weeks) |
| Replications | 20 |

To validate physiological performance in terms of growth at different oxygen saturation the wet-lab experiment conducted by [?] was replicated. For comparison of wet-lab and simulation results see Figures 14, 15, 16.

### 8.1.2 Experiments with Basic_variable.mlrj

The following experiment are conducted with the basic physiology expanded to control food intake level. This can be understood a proxy for processes such as: foraging success, stress, individual variance in metabolic performance, etc..

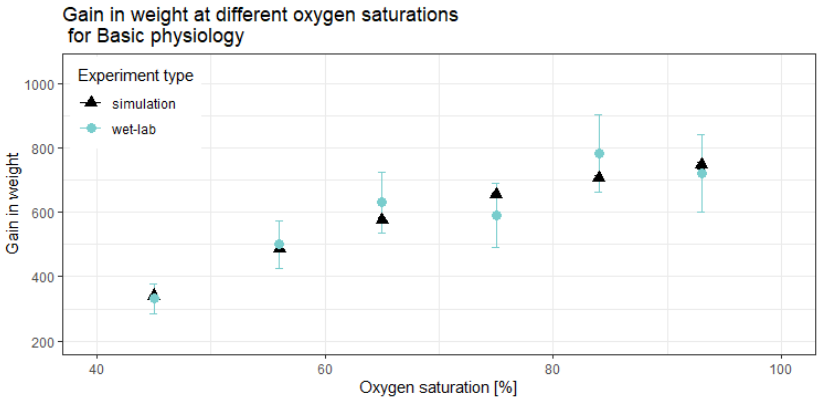



Figure 14: Results of 12 week growth experiment with Basic.mlrj

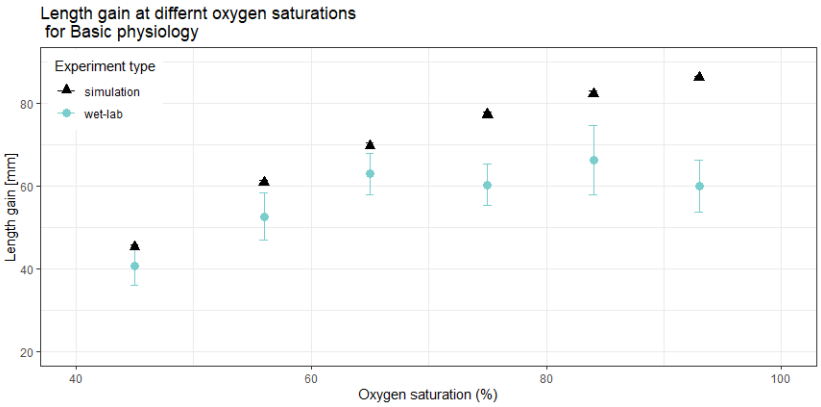



Figure 15: Results of 12 week growth experiment with Basic.mlrj

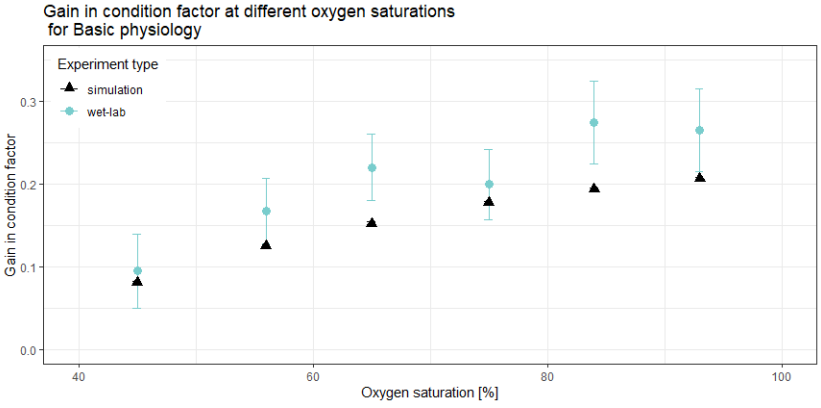



Figure 16: Results of 12 week growth experiment with Basic.mlrj

### 12 week growth experiment

```
// Experiment for modelled cod physiology in a 12 week feeding experiment at different
    oxygen levels with varied food intake levels
// Model file: Basic_variable_ex.mlrj
// Output folder: result_basic_variable_ex
// Duration example: 37 min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {

  model = "./Basic_variable_ex.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1

  stopTime = 201600
  replications = 20

  observeAt(range(0, 201600, 201600))
  scan("o" <~ (45,56,65,75,84,93))
  scan("t" <~ (10))
  scan("va" <~ (0.25,0.5,0.75,1))

  observe("length" ~ attribute("*/GM",0))
  observe("weight" ~ attribute("*/GM",1))

  csvOutputDirectory(() => "result_basic_variable_ex")

  withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Basic_variable_ex.mlrj |
| Duration: | 201600TU $\hat{=}$ 12 weeks |
| Scanning for: | oxygen (45,56,65,75,84,93) |
| Monitoring: | length and weight |
| Interval: | Observation only at end |
| Replications | 20 |

### 12 week growth experiment with variable appetite

```
// Experiment for modelled cod physiology in a 12 week feeding experiment at different
    oxygen levels with varied food intake levels
// Model file: Basic_variable_ex.mlrj
// Output folder: result_basic_variable_ex
// Duration example: 37 min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {
  model = "./Basic_variable_ex.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1

  stopTime = 201600
  replications = 20

  observeAt(range(0, 201600, 201600))
  scan("o" <~ (45,56,65,75,84,93))
  scan("t" <~ (10))
  scan("va" <~ (0.25,0.5,0.75,1))

  observe("length" ~ attribute("*/GM",0))
  observe("weight" ~ attribute("*/GM",1))

  csvOutputDirectory(() => "result_basic_variable_ex")

  withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Basic_variable_ex.mlrj |
| Duration: | 201600TU $\hat{=}$ 12 weeks |
| Scanning for: | oxygen (45,56,65,75,84,93) |
| Monitoring: | length and weight |
| Interval: | Observation only at end (201600TU $\hat{=}$ 12 weeks) |
| Replications | 20 |

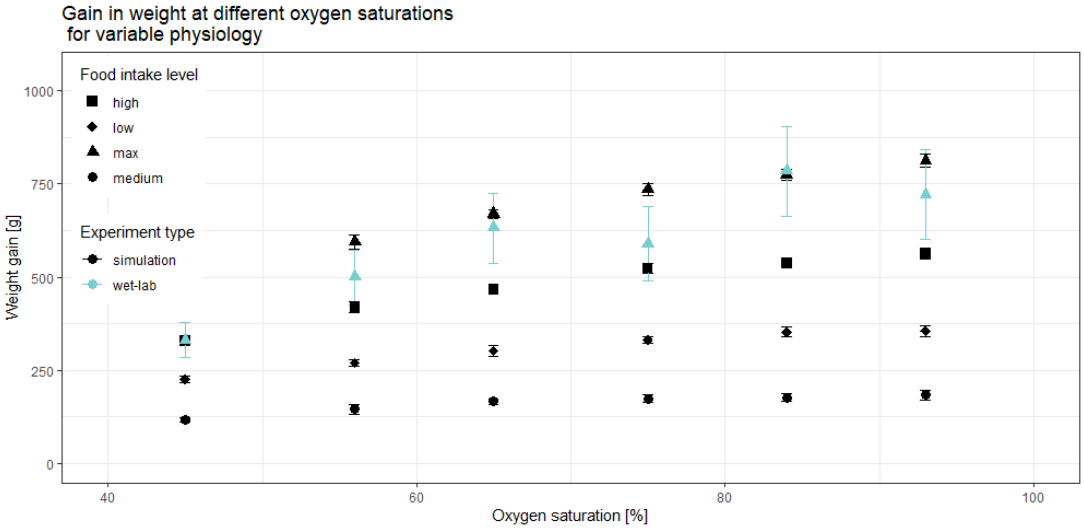

Gain in weight at different oxygen saturations
for variable physiology
Food intake level
high
low
max
medium
Experiment type
simulation
wet-lab
Weight gain [g]
1000
750
500
250
0
40
60
80
100
Oxygen saturation [%]

Figure 17

### 8.1.3 Experiments with Liver_ex.mlrj

**12 week growth experiment with the Liver model**

```
// Experiment for modelled cod physiology in a 12 week feeding experiment at different
    oxygen levels with ingestion driving parasitation of liver
// Model file: Liver_ex.mlrj
// Output folder: result_liver_ex
// Duration example: 49 min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {
  model = "./Liver_ex.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1
  stopTime = 201600
  replications = 20

  scan("o" <~ (45,56,65,75,84,93))
  scan("t" <~ (10))
  scan("lop" <~ (0,0.2,0.8,1))

  observeAt(range(0, 201600, 201600))
  observe("length" ~ attribute("*/GM",0))
  observe("weight" ~ attribute("*/GM",1))
  observe("prey_parasitation" ~ attribute("*/Prey",6))
  observe("Liver_oc_infest" ~ attribute("*/Liver",1))

  csvOutputDirectory(() => "result_liver_ex")

  withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Liver_ex.mlrj |
| Duration: | 201600TU $\hat{=}$ 12 weeks |
| Scanning for: | oxygen (45,56,65,75,84,93), level of parasitation (lop) (0,0.2,0.8,1) |
| Monitoring: | length and weight |
| Interval: | Observation only at end (201600TU $\hat{=}$ 12 weeks) |
| Replications | 20 |

### 8.1.4 Experiments with Condition_ex.mlrj

**12 week growth experiment with the Condition model**

```
// Experiment for modelled cod physiology in a 12 week feeding experiment at different
    oxygen levels with condition factor determining parasite infestation
// Model file: Condition_ex.mlrj
// Output folder: result_condition_ex
// Duration example: 129 min

package org.sessl
object ExampleExperiment extends App {

import sessl._
import sessl.mlrules._

execute {
 new Experiment with Observation with ParallelExecution with CSVOutput {

  model = "./Condition_ex.mlrj"
  simulator = StandardSimulator()
  parallelThreads = -1
  stopTime = 201600
  replications = 20

  observeAt(range(0, 201600, 201600))
  scan("o" <~ (45,56,65,75,84,93))
  scan("t" <~ (10))
  scan("lop" <~ (0,1))
  scan("group" <~ (
  "1 GM(442,605 ,30,150,1,1,0,'E',0,13,1002)[Stc(0,1,1002)[] + Liver(0,0,1002)]"
  ,"1 GM(442,691 ,30,150,1,1,0,'E',0,13,1003)[Stc(0,1,1003)[] + Liver(0,0,1003)]"
  ,"1 GM(442,777 ,30,150,1,1,0,'E',0,13,1004)[Stc(0,1,1004)[] + Liver(0,0,1004)]"
  ,"1 GM(442,864 ,30,150,1,1,0,'E',0,13,1005)[Stc(0,1,1005)[] + Liver(0,0,1005)]"
  ,"1 GM(442,950 ,30,150,1,1,0,'E',0,13,1006)[Stc(0,1,1006)[] + Liver(0,0,1006)]"
  ))

  observe("length" ~ attribute("*/GM",0))
  observe("weight" ~ attribute("*/GM",1))
  observe("prey_parasitation" ~ attribute("*/Prey",6))
  observe("Liver_oc_infest" ~ attribute("*/Liver",1))

  csvOutputDirectory(() => "result_condition_ex")

  withRunResult(writeCSV) }}}
```

| | |
|---|---|
| Model: | Condition_ex.mlrj |
| Duration: | 201600TU $\hat{=}$ 12 weeks |
| Scanning for: | oxygen (45,56,65,75,84,93), level of parasitation (lop) (0,1), condition (realised by varying weight) (0.7,0.8,0.9,1,1.1) |
| Monitoring: | length and weight |
| Interval: | Observation only at end (201600TU $\hat{=}$ 12 weeks) |
| Replications | 20 |

# References


[1] ICES. Cod (*Gadus morhua*) in subdivisions 24-32, eastern baltic stock (eastern baltic sea) (2019). Publisher: ICES.

[2] Eero, M., Cardinale, M. & Storr-Paulsen, M. Emerging challenges for resource management under ecosystem change: Example of cod in the Baltic Sea. *Ocean & Coastal Management* **198**, 105314 (2020).

[3] Grimm, V. *et al.* The ODD protocol: A review and first update. *Ecological Modelling* **221**, 2760–2768 (2010).

[4] Grimm, V. *et al.* The ODD Protocol for Describing Agent-Based and Other Simulation Models: A Second Update to Improve Clarity, Replication, and Structural Realism. *Journal of Artificial Societies and Social Simulation* **23**, 7 (2020).

[5] Pierce, M. E., Krumme, U. & Uhrmacher, A. M. BUILDING SIMULATION MODELS OF COMPLEX ECOLOGICAL SYSTEMS BY SUCCESSIVE COMPOSITION AND REUSING SIMULATION EXPERIMENTS. In *2018 Winter Simulation Conference (WSC)*, 2363–2374 (IEEE, Gothenburg, Sweden, 2018).

[6] Pierce, M. E. *et al.* Developing and validating a multi-level ecological model of eastern baltic cod (*Gadus morhua*) in the bornholm basin – a case for domain-specific languages **361**, 49–65 (2017).

[7] Burke, J. D. Vertebrate blood oxygen capacity and body weight. *Nature* **212**, 46–48 (1966).

[8] Ronald, K., Macnab, H. C., Stewart, J. E. & Beaton, B. Blood Properties of Aquatic Vertebrates: I. Total Blood Volume of the Atlantic Cod, *Gadus morhua* l. *Canadian Journal of Zoology* **42**, 1127–1132 (1964).

[9] Claireaux, G., Webber, D., Lagardère, J.-P. & Kerr, S. Influence of water temperature and oxygenation on the aerobic metabolic scope of atlantic cod (*Gadus morhua*) **44**, 257–265.

[10] Dutil, J.-D. & Lambert, Y. Natural mortality from poor condition in atlantic cod (*Gadus morhua*) **57**, 826–836 (2000).

[11] Bjornsson, B. Optimal temperature for growth and feed conversion of immature cod (*Gadus morhua* l.) **58**, 29–38.

[12] Nash, R. D., Valencia, A. H. & Geffen, A. J. The origin of fulton's condition factor—setting the record straight **31**, 236–238 (2006).

[13] Plante, S., Chabot, D. & Dutil, J.-D. Hypoxia tolerance in atlantic cod **53**, 1342–1356.

[14] Ryberg, M. P. *et al.* Physiological condition of eastern baltic cod, *Gadus morhua*, infected with the parasitic nematode Contracaecum osculatum. *Conservation Physiology* **8**, coaa093 (2020).

[15] Hornetz, P. *Spatio-temporal distribution and food intake of cod (Gadus morhua) along a depth gradient in the western Bornholm Sea.* Master's thesis, University of Hamburg (2020).

[16] .

[17] Maus, C., Rybacki, S. & Uhrmacher, A. M. Rule-based multi-level modeling of cell biological systems. *BMC Systems Biology* **5**, 166 (2011).

[18] Haack, F., Lemcke, H., Ewald, R., Rharass, T. & Uhrmacher, A. M. Spatio-temporal model of endogenous ROS and raft-dependent WNT/beta-catenin signaling driving cell fate commitment in human neural progenitor cells 28 (2015).

[19] Haack, F. *Exploring the spatio-temporal dynamics of lipid rafts and their role in signal transduction*. Dissertation (2015).

[20] Warnke, T., Helms, T. & Uhrmacher, A. M. Reproducible and flexible simulation experiments with ML-Rules and SESSL. *Bioinformatics* **34**, 1424–1427 (2018).

[21] Chabot, D. & Dutil, J.-D. Reduced growth of Atlantic cod in non-lethal hypoxic conditions. *Journal of Fish Biology* **55**, 472–491 (1999).