sOAR

A tool for modelling optimal animal life-history strategies in cyclic environments

Version 1.0
User Manual

Merlin Schäfer
Stephan Menz
Florian Jeltsch
Damaris Zurell

November 2016

Contents

Acknowledgement

We are very thankful to Karsten Isakovic for his efforts to improve code performance. We also thank Andrea Flack, Martin Wikelski, Michael Kaatz, Ran Nathan, Shay Rotich and Wolfgang Fiedler for helpful discussions on bird migration and

Jacint Tökölyi, John McNamara and Zoltan Barta for helpful OAR feedback. sOAR incorporates codes from Karsten Isakovic, the library Libconfig by Lindner et al. (http://www.hyperrealm.com/libconfig/) and a Brent optimizer based on Brent [Bre73]. We acknowledge the generous support of DIP grants (DFG) NA 846/1-1 and WI 3576/1-1 and DFG-GRK grants 2118/1 to FJ in the framework of the BioMove Research Training Group. DZ has received funding from the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement no. 624958.

1 Introduction

1.1 Background

The behaviour of many animals is influenced by rhythmic patterns of their environment like the solar day or the seasons [Kou06, NH14]. Migratory birds, for example, respond to seasonal environments by undertaking regular long-distance journeys between breeding and wintering habitat [Ale90, Ber12]. It can be expected that such behavioural responses to cycles of environmental conditions, food availability or predation risk are shaped by evolution and thus represent an evoutionarily optimized behavioural strategy [HM99, MHC01]. This assumption allows for theoretical studies of cyclic behavioural patterns with an optimization based approach. In particular, stochastic dynamic programming (SDP) can be used for identifying optimal life history strategies in cyclic environments and for simulating corresponding population dynamics [HM99, MHC01].

SDP is a well-known method for solving multistage decision processes [Ber05] and a framework using SDP to compute optimal annual routines (OAR) in behavioural ecology has been published by Houston and McNamara [HM99] a while ago. This framework is grounded in state-based life-history theory. Assuming that natural selection shapes the behaviour of organisms which in turn take behavioural decisions based on knowledge of their environment and their own state, the framework describes a dynamic programming model that determines an animal's behavior depending on its state and time, integrating periodically changing environmental conditions. The computational procedure consists of two steps: a backward iteration calculating the fitness-maximizing optimal behavioural strategy for a given scenario and a forward iteration simulating population development under the computed optimal strategy over time. [HM99, CM00]

A major advantage of the approach is that the full life history of an animal is taken into account, incorporating long-term consequences of events and decisions into behaviour [HM99, MHC01]. Hereby, possible actions usually do not exclude each other per se. Instead, behavioural rules emerge from the model during its solution [HM99, FSB$^+$08]. Similarly, the fitness consequences of a particular action arise from its defined effects on state variables. The approach thus readily includes temporal trade-offs as well as trade-offs between different behavioural options [HM99, MHC01, FSB$^+$08]. The mechanistic perspective further qualifies it for predictive modelling of behaviour and its evolution under new environmental regimes [BK13].

However, despite the outlined advantages of the OAR framework and its suitability for a broad range of research questions [FSB$^+$08], the number of actual implementations is low and focused mainly on small song birds. For example, it has been used to theoretically study the optimal timing of repro-

duction in non-migratory [HM99] and migratory birds [MWH98] or the timing of avian reproduction in unpredictable environments [TMHB12]. Additionally, many existing moult strategies of birds [BHM+06, BMH+08, HH95] could be reproduced by altering the local environmental regime in an avian OAR model. Employment of the optimal annual routine framework for other focal animals than birds is rare but has been performed for zooplankton by [VJTF07].

Reasons for the restricted application of the OAR framework might be the complexity of the modelling approach requiring time, mathematical knowledge and programming expertise when developing and implementing the SDP model [FSB+08, BK13]. Especially, since the so called "curse of dimensionality" exponentially increases computational intensity with any additional predictor of behaviour [Ber05]. Besides, large datasets can be required for parametrization, which often were not obtainable in the past [FSB+08, BK13]. Accordingly, to discover and exploit the full potential of the OAR framework in behavioural and evolutionary ecology, a flexible and user-friendly implementation of it is needed.

1.2 Aim and Purpose

With sOAR, we aim to provide a flexible and ready-to-use model that can easily be employed for investigating optimal animal life history strategies under cyclic environmental conditions. Since animals are increasingly facing anthropogenic changes that affect their environment, it is important to study and better understand the complete behavioural cycle of animals as embedded into the periodic environment from a mechanistic and evolutionary perspective [WW08, VCvO+10, HBSS+13]. The wellknown but infrequently used OAR framework by Houston and McNamara [HM99] represents a promising approach for doing so on a theoretical basis [FSB+08]. Thus, we wish to promote the method in ecology by offering an open-source, modular and fast implementation of the original OAR framework that is extended by useful features.

The presented software computes optimal animal life-history strategies within cyclic environments depending on the state of the animal and periodic environmental conditions (backward iteration), including demographic stochasticity. Subsequently, population dynamics under the computed optimal strategy can be simulated in sOAR (forward iteration). Considered behavioural options concern foraging, reproduction and optionally migration, while individuals are described through the state variables of energy reserves, experience, reproductive status, location, and optionally its state of migration and health condition. The environment in terms of food availability and optionally migratory costs (e.g. the probability of active flight as determined by wind conditions) is cyclic and defined by the user. User-defined parameters and response functions further capture the direct consequences of actions on state variables, e.g. through metabolism, reproductive efforts, growth of experience, predation, energy uptakes, flight energetics or immune response. Additional stochasticity can be set, for instance to account for environmental unpredictability.

At large, sOAR can be used in two different ways. First, general life history strategies can be computed with sOAR, for example when working with functional types and parameters that are theoretically derived from general biological principles. Second, sOAR can be employed to investigate a particular focal species in a particular environment using species-specific data. In both cases, the software can easily be adjusted to a variety of species or functional types for addressing questions regarding

- the timing of reproduction or migration,
- the effect of different environmental conditions on behavioural patterns and population dynamics,
- the relationship between reproductive value and the state of an animal or a particular time period,
- critical levels of state for behaviour.

Additionally, sOAR can facilitate the development of new cyclic SDP models by serving as an implemented starting point for specialised research questions involving for instance different behavioural options, additional locations, other consequences of behaviour or a modified optimization criterion. However, coding expertise and comprehensive knowledge of SDP are required in this case.

1.3 Strengths and Limitations

In the following, the main strengths and limitations of the current version of sOAR (V1.0), as we see them, are compiled. Lesser advantages and restrictions are indicated throughout the remainder of this manual, whereas a short general evaluation of the underlying OAR framework can be found in the accompanying article.

The key strengths of sOAR are:

1. It provides a powerful and flexible open-source implementation of the SDP algorithm for modelling cyclic behavioural routines as introduced by Houston and McNamara [HM99], overcoming many of the challenges that hindered a wide-spread application of this well-reckognized framework in the past [FSB+08].
2. The strict seperation of computational routines from parameter settings and modularity ensures high user-friendliness of sOAR and make it applicable also to users who have only little coding experience.
3. The application field of the original framework [HM99, MWH98] has been broadened by extending the original framework to differentiate between active and passive flight costs and allowing for periodic weather conditions acting on migration. For example, it is now also possible to study the timing of migration in soaring birds which depend on the availability of thermals, for example, or to look at ontogenetic differences in birds where the use of flapping versus soaring flight is age dependent.

4. sOAR is freely available from *https://sourceforge.net/projects/soar-animal-behaviour* under the terms of the *GNU General Public License*, Version 3 (*GPLv3*) or any later version, meaning the program can freely be used, modified and distributed in compliance with the license (see *https://www.gnu.org/licenses/licenses.en.html*).

5. By offering the source code, users are free (but do not have) to further adapt sOAR to their needs, e.g. integrating more advanced density dependence as described by [BMH$^+$08], using the current code as basic framework. Users can thereby rely and build upon the computational routine and numerical optimizations implemented in sOAR, which are among the most complicated parts when implementing OARs.

6. Implementation happened in the widespread, fast and standardized C++ programming language and is additionally optimized for speed.

7. sOAR comes with a supporting library that performs comprehensive internal checks to catch erroneous or missing parameterizations by the user.

8. To facilitate the use of sOAR and familarization to it, the software package includes four replicable sample model configurations for which some major results are presented in the User Manual.

9. sOAR has been tested successfully on the published reproduction model by Houston and McNamara [HM99] as well as the migration model by McNamara et al. 1998 [MWH98] (see Chapter 4)

The current limitations of sOAR are:

1. Like in other SDP or IBM models, a relatively large dataset on the focal species is required for setting all required parameters - to the point that sometimes it might be challenging to parametrize sOAR for a particular species. However, data collection efforts are ever increasing as well as efforts to develop proper estimation methods so that this aspect should become less important in the near future.

2. Available behavioural options are restricted to foraging, reproduction and migration though others like molt might be of interest. However, the user is free to customize sOAR to his needs under the terms of the *GNU General Public License*, Version 3 (*GPLv3*) or any later version (see *https://www.gnu.org/licenses/licenses.en.html*).

3. Density-dependence is only implicitely implemented (see [BHM$^+$06, BMH$^+$08] for an example of the more complicated explicit integration of density-dependence into an OAR framework) and inter-specific interactions are excluded.

4. The implementation of migration is kept quite simple since there are only two locations but no staging sites included, the duration of migration is fixed and migratory costs can only depend on the level of energy reserves and time. A more advanced OAR version including staging sites and a flexible duration of migration was used by [BMH$^+$08] to study optimal moult strategies in migratory birds.

5. When the health condition is enabled, computing the probability to die from disease is currently restricted to the approach employed by McNamara et al. [MWH98].

6. The fast and open source implementation focusing on user-friendliness increases code complexity in the background, e.g. by integrating the *libconfig* library for processing configuration files (*http://www.hyperrealm.com/libconfig/*). This might make it more difficult to understand the program structure when tailoring sOAR code to specific research questions in an advanced stage of sOAR usage.

7. With the inherent complexity of the underlying model and the strict separation of computational routines from parameter settings the user might be tempted to employ sOAR without fully understanding the meaning of his or her settings. Here, the user is asked to thoroughly consult this manual and to familiarize with the the underlying OAR framework [MWH98, HM99] as well as the general SDP approach in behavioural ecology [HM99, CM00] when using sOAR.

2 Concept and Methods

The model description partly follows the ODD+D (ODD for human decision-making) protocol [MBD+13] which in turn is based on the ODD (Overview, Design concepts, Details) protocol [GBB+06, GBD+10].

2.1 Overview

2.1.1 Entities, state variables and scales

The model is developed for studying the behaviour and population dynamics of animals in cyclic environments. Animals are hereby described through state variables and can perform certain behavioural actions depending on their state and time. To characterize the animal, six state variables are implemented: *energy reserves*, *health condition*, *experience*, *age of offspring*, *location* and *migratory state*. The state of the animal and the environment as well as time influence the animal's behaviour.

Figure 2.1 provides an overview of the model as implemented in sOAR, capturing the relationships between state variables, considered behavioural activities, the environment and other user-defined parameters. The following subsection introduces the state variables while their dynamics are described in section 2.1.2. Details on how to set these variables and other parameters characterizing the focal animal and its environment is provided in Chapter 3.2. Contingent upon the selected setting, some parameters may be optional. Available actions are described in section 2.2.2.

Space is implicitly included in the model such that an animal's location is part of its state. Currently, the model works for up to two locations between which migration can occur (optional). The temporal resolution is an arbitrary periodic time-span which is discretized by a user-defined number of stages or decision epochs at which a behavioural decision needs to be taken (e.g. a period of one year consisting of 52 weeks). These are characterised by user-defined cyclic environmental conditions that drive the model since they determine potential energy intake and movement costs (optional) which in turn are major influencing factors of behaviour.

2.1.1.1 State variables

Energy reserves, x: Energy reserves are discretized in user-defined steps from a critical minimum value x_{min} at which the individual and its dependent brood will die to a maximum capacity x_{max}:

$$x_{min} \leq x \leq x_{max}, \quad x \in \mathbb{R}.$$

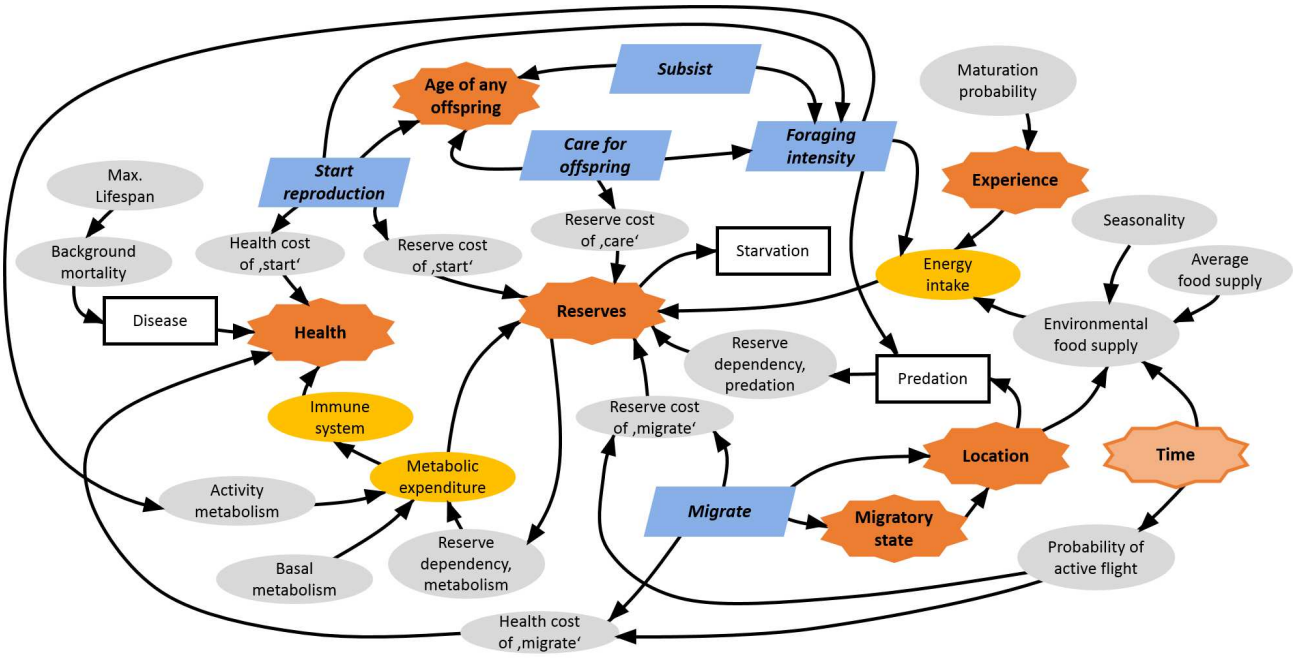

Figure 2.1: Model overview for sOAR depicting the implemented interdependencies (arrows) between state variables (orange stars), behavioural activities (blue parallelograms), user-defined input parameters (grey ellipses), internal submodels (yellow ellipses) and sources of mortality (white rectangles).

Depending on the context, model energy reserves x could be interpreted as metabolic energy, fat deposits or body weight.

Health condition, y (optional): Health condition varies in a user-defined number of steps between a critical minimum value y_{min} and an upper limit y_{max}, such that

$$y_{min} \leq y \leq y_{max}, \quad y \in \mathbb{R}.$$

The variable can be interpreted as the state of the immune system of the animal and its susceptibility to disease (cf. [HM99]). The minimum value y_{min} leads to death of the animal due to disease. In this case, any dependent offspring will die, as well. The health variable may be switched off.

Experience, e: This state variable increases with age from the minimum level of zero for newly independent offspring to the maximum experience e_{max} of a fully matured animal:

$$0 \leq e \leq e_{max}, \quad e \in \mathbb{N}.$$

The associated probability p_{exp} that an animal's experience grows by one unit per decision epoch can be defined such that full experience will on average be reached by the typical maturation age of the focal species. Experience may affect actual energy intake from foraging so that immature young can

be less efficient foragers than adults (see Section 2.3.1 for details).

Age of any offspring, a: The variable indicates the reproductive status of the animal. Animals that do not care for any offspring posess a value of $a = -1$, while for a parent animal the value of this variable represents the age of its offspring in number of decision epochs. In the model,

$$-1 \leq a \leq a_{max}, \quad a \in \mathbb{Z},$$

where a_{max} is the age at which offspring becomes independent from its parent. A potential period of incubation or pregnancy is included in this time frame whereby different costs can be defined for the phases of incubation/pregnancy parental care. Note that dependent offspring is treated as part of the parent's state. Thus, dependent offspring has no experience variable yet but its ageing corresponds to a potential change of its value to the parent.

Location, o: The model considers up to two locations between which individuals may migrate (optional):

$$1 \leq o \leq 2, \quad o \in \mathbb{N}.$$

If the migration option is switched off, only one location is considered in the model. Each activated location needs to be characterized by its food availability and potentially the probability of active flight at that location.

State of migration, s (optional): This variable only comes into effect if the option to migrate is enabled. It indicates whether or not an animal is migrating, and gives the current week of migration in case it is. In the model

$$0 \leq s \leq s_{max}, \quad s \in \mathbb{Z},$$

where $s = 0$ means that the animal is not migrating, s_{max} corresponds to the user-defined duration of migration and $s \in [1, ..., s_{max}]$ defines the stage of migration. Note that the probability of active flight as determined e.g. by wind conditions can be defined for each decision epoch and may thus change over the course of migration.

2.1.1.2 Stage variable

As in most dynamic programming problems the stage variable is time t. Accordingly, the stage variable indicates the current time step within the user-defined cycle of periodic environmental conditions. This cycle is divided into T consecutive decision epochs $t = 0, 1, 2..., T - 1$:

$$t = 0, 1, 2, ..., T - 1, \quad t \in \mathbb{N},$$

where time T marks the end of one cycle as well as the beginning of the subsequent cycle. Additionally, the maximum number n of complete time cycles through which the SDP algorithm may run to compute the optimal strategy needs to be defined. However, once the backward iteration converged towards an optimal strategy, only time t within the cycle is important for determining the optimal behavior for a given state and time.

2.1.2 Dynamic programming equations

In the following, the mathematical background of sOAR is provided (also see [HM99]). In particular, the underlying dynamic programming problem of the model is formulated in terms of strategies, state equations, a criterion function and an optimal value function. The terminal condition is presented, as well as the wrap-around condition to stitch the end of one cycle to the beginning of the next cycle during computations. Besides, the reproductive value and the evolutionary fitness of the behavioral strategy are defined in this section. Details on the integrated submodels can be found in section 2.3. In chapter 3, the user learns which parameters to set and how to.

2.1.2.1 Strategies

sOAR optimizes behavioral strategies which map the state of an animal into behavioral actions that stem from a state dependent set of actions and involve a corresponding optimal foraging intensity. Potential actions in the model are (cf. 2.2.2):

- Initiate reproduction ("start")
- Care for offspring ("care")
- Migrate ("migrate") (optional)
- Do none of the above ("subsist")

It is assumed that the animal selects the optimal foraging intensity $u^* \in [0, 1]$ for each behavioural option, i.e. the foraging intensity u which maximizes the long-term number of descendants, given a particular behavioral activity is performed. For simplicity, a fixed energy intake through foraging is assumed during migration which the user has to prorate with the migration costs.

2.1.2.2 State equations

State equations describe the development of state variables between time t and $t + 1$, given the animal has not died by that time. Depending on the selected behavioural action these equations will take different forms.

Dynamics of energy reserves, x: The change of energy reserves of an animal that is not migrating is described by

$$x_{action,t+1} = \min(x_{max}, \max(x_{min}, x_t + \gamma(e, o, u, t) - c(u, x) - \Delta x_{action})), \tag{2.1}$$

where u is the foraging intensity, the function $\gamma(e, o, u, t)$ gives the energy intake, the function $c(u, x)$ represents the energy expenditure through metabolism when subsisting and Δx is the energetic cost of any demanding extra activity (reproductive efforts, migration). Since offspring is abandoned or becomes independent at the beginning of a decision epoch, the "subsist" option is selected in that decision epoch and no extra energetic costs will occur. Working with the minimum and maximum

function ensures that the new value of reserves x_{t+1} lies within the allowed range. The implemented submodels of actual energy intake and energetic expenditure are described in section 2.3.

The extra metabolic cost Δx_{action} takes the value

$$\Delta x = \begin{cases} \Delta x_{subsist} = 0 & \text{if animal subsists,} \\ \Delta x_{start} & \text{if reproduction is initiated,} \\ \Delta x_{incubate} & \text{if eggs are incubated,} \\ \Delta x_{care} & \text{if offspring is provisioned,} \end{cases} \tag{2.2}$$

where Δ_{start}, $\Delta_{incubate}$ and Δ_{care} are user-defined constants.

The new mean level of reserves during migration is

$$\begin{aligned} x_{migrate,t+1} = \min(x_{max}, \max(x_{min}, x_t - (p_{act}\Delta x_{act}(1 + f_{x,act}(x)) \\ + (1 - p_{act})\Delta x_{pas}(1 + f_{x,pas}(x))))), \end{aligned} \tag{2.3}$$

where p_{act} is the probability of active flight and Δx_{act} and Δx_{pas} are user-defined constants defining the energetic cost of active and passive flight. The user-defined functions $f_{x,act}(x)$ and $f_{x,pas}(x)$ capture the influence of reserves on flight costs during active and passive flight (see chapter 3.2 for allowed functional types).

Dynamics of health condition, y: The dynamics of health condition are described by

$$y_{action,t+1} = \min(y_{max}, \max(y_{min}, y_t + \alpha(c) - \Delta y_{action})), \tag{2.4}$$

where $\alpha(c)$ is a user-defined function describing the recovery or deterioration of health depending on metabolism (see section 2.3) and Δy are any extra health costs associated to special events like migration or reproduction. Implemented extra health costs Δy are

$$\Delta y = \begin{cases} \Delta y_{start} & \text{if reproduction is initiated,} \\ \Delta y_{migrate} & \text{if migrating,} \\ \Delta y_{subsist} = 0 & \text{if not reproducing or migrating.} \end{cases} \tag{2.5}$$

Dynamics of experience, e: An animal's experience grows by one unit from time step t to $t + 1$ with probability p_{exp} but cannot be higher than the maximum value of experience e_{max} so that

$$e_{t+1} = \begin{cases} e' = \min(e + 1, e_{max}) & \text{with probability } p_{exp} \\ e & \text{with probability } (1 - p_{exp}). \end{cases} \tag{2.6}$$

When calculating the optimal strategy, the weighted average of the number of descendants left by an animal with either experience e or e' at time $t + 1$ is applied.

Dynamics of reproductive state, a: The state variable a indicates the reproductive state of an individual. Individuals can either have no offspring as indicated by a value of $a = -1$ or have offspring of age $0 < a < a_{max}$ where a_{max} is the age at which offspring becomes independent and is abandoned by the parent. The new value of reproductive state at time $t + 1$ is

$$a_{t+1} = \begin{cases} -1 & \text{if migrating or subsisting,} \\ 0 & \text{if starting reproduction,} \\ a_t + 1 & \text{if caring for offspring and } u_{crit} \leq u \leq 1. \end{cases} \tag{2.7}$$

The decision of a parent individual to care for its offspring depends on whether its foraging intensity u lies above the critical value u_{crit} to fulfill the offspring's energy needs. If $u_{crit} \leq u \leq 1$, the parent can provision enough food for the young to survive and may select the "care" option. Otherwise, the offspring needs to be abandoned by the parent which is forced to choose the "subsist" option. Note that care for offspring includes any incubation duties.

Dynamics of migratory state, s:

Depending on the selected behaviour the new value of this state variable will either indicate a period of stay or the length of the migratory period up to now:

$$s_{t+1} = \begin{cases} 0 & \text{if not migrating or } s_t = s_{max}, \\ s_t + 1 & \text{if migrating and } s_t < s_{max}. \end{cases} \tag{2.8}$$

where s_{max} represents the user-defined duration of migration after which a migratory individual will reach its destination. Accordingly, the state of migration at the next time epoch s_{t+1} will equal -1 when the animal is currently not migrating or is in the last epoch of its migration reaching its destination by the next time step. Whereas s_{t+1} will increase by one unit when migrating but not reaching the other location yet - indicating the length of migration until then. Note that this state variable will only change if the migration option is enabled and that individuals are forced to finish their journey once started.

Dynamics of location, o:

If the migration option is not enabled in the general settings, location will be constant over time. Otherwise, the animal's location may change between two sites due to migration:

$$o_{t+1} = \begin{cases} o_t & \text{if } 0 \leq s_t < s_{max}, \\ 2 - o_t + 1 & \text{if } s_t = s_{max}. \end{cases} \tag{2.9}$$

So if the animal is in its final week of migration at time t (and does not die during that decision epoch), it will be at the other location by the next time step $t + 1$. Note that the value of location will only be updated once migration is completed so that this state variable additionally indicates the origin of an individual that is migrating.

2.1.2.3 Criterion function

The criterion function H_{action} specifies the reproductive payoff from performing a particular behaviour and is used as a criterion when comparing different behavioural options. In particular, it gives the number of descendants left into the far future by an individual being in a certain state at a particular time which is assumed to follow the optimal strategy from that time onward. Here, the criterion function H_{action} is a state-dependent nested function of other functions incorporating mortality by predation and disease, metabolic expenditure and energy intake which in turn are influenced by the environment (see Section 2.3 for details on these functions).

To define the criterion function, we denote the reproductive value of an animal in state (x, y, e, a, o, s, t) at the beginning of time epoch t and n cycles before the target time cycle by $V_n(x, y, e, a, o, s, t)$. Further, the state and activity dependent survival probability is denoted as $S(y, o, s, t, u)$. Then, the reproductive payoff H_{action} from a particular behaviour received by a subsisting individual is

$$\begin{aligned} H_{subsist}(u) =& S(y, o, s, t, u)[(1 - p_{exp})V_n(x_{nocare}, y_{subsist}, e, -1, o, s, t + 1) \\ &+ p_{exp}V_n(x_{nocare}, y_{subsist}, e', -1, o, s, t + 1)], \end{aligned} \tag{2.10}$$

whereas an animal that starts reproduction receives the payoff

$$\begin{aligned} H_{start}(u) =& S(y, o, s, t, u)[(1 - p_{exp})V_n(x_{start}, y_{start}, e, 0, o, s, t + 1) \\ &+ p_{exp}V_n(x_{start}, y_{start}, e', 0, o, s, t + 1)], \end{aligned} \tag{2.11}$$

the payoff of caring for existing offspring is

$$\begin{aligned} H_{care}(u) =& S(y, o, s, t, u)[(1 - p_{exp})V_n(x_{care}, y_{subsist}, e, a + 1, o, s, t + 1) \\ &+ p_{exp}V_n(x_{care}, y_{subsist}, e', a + 1, o, s, t + 1)] \end{aligned} \tag{2.12}$$

and migrating results in the payoff

$$\begin{aligned} H_{migrate}(u) =& S(y, o, s, t, u)[(1 - p_{exp})V_n(x_{migrate}, y_{migrate}, e, -1, o, s, t + 1) \\ &+ p_{exp}V_n(x_{migrate}, y_{migrate}, e', -1, o, s + 1, t + 1)]. \end{aligned} \tag{2.13}$$

Please note that additional stochasticity has been introduced at this step as described in Section 2.3.5 and Houston and McNamara [HM99] to avoid potential grid effects and to increase biological realism. Thus, the reproductive value $V_n(x_{action}, y_{action}, e, a, o, s, t)$ of a particular behavioural activity is actually a weighted sum of the reproductive value of eight different combinations of energy reserves and health condition. These combinations are variants of the deterministic level of reserves and health at the next time step and shall reflect variability in food availability and individual performance.

2.1.2.4 Optimal value function

The optimal value function assigns the optimal reproductive pay-off to a state and hereby identifies the best behavioural action. It is used to compute the optimal strategy and the reproductive value at time t. To obtain the optimal payoff, we first compute the optimal foraging intensity for each behavioral option (except for migration, see Section 2.3.4), i.e. the foraging intensity u that maximizes the (expected) long-term number of offspring for that behavioural option:

$$\begin{aligned} H^*_{nocare} &= \max_{0\leq u\leq 1} H_{nocare}(u), \\ H^*_{start} &= \max_{0\leq u\leq 1} H_{start}(u), \\ H^*_{care} &= \max_{u_{\text{crit}}\leq u\leq 1} H_{care}(u), \\ H^*_{migrate} &= H_{migrate}\ . \end{aligned} \tag{2.14}$$

Then, in order to gain the optimal behavioural decision and the corresponding optimal reproductive value for the particular state-time combination, the admissible behavioural option with the maximum reproductive payoff is selected.
If the animal is alive, has no offspring and is not migrating ($x > 0$, $y > 0$, $a = -1$, $s = 0$), the optimal payoff is

$$V_n(x, y, e, a, o, s, t) = \max(H^*_{\text{nocare}}, H^*_{\text{start}}, H^*_{\text{migrate}}). \tag{2.15}$$

An alive individual with dependent offspring ($x > 0$, $y > 0$, $0 \leq a \leq a_{max} - 1$, $s = 0$) has the optimal payoff

$$V_n(x, y, e, a, o, s, t) = \max(H^*_{nocare}, H^*_{care}). \tag{2.16}$$

When the offspring becomes independent ($x > 0$, $y > 0$, $a = a_{max}$, $s = 0$) and is thus abandoned by the parent, the parent receives a payoff that includes the reproductive value of its young:

$$V_n(x, y, e, a, o, s, t) = H^*_{nocare} + n_{offspring}\, V_n(x_{indep}, y_{indep}, 0, -1, o, 0, t), \tag{2.17}$$

where x_{indep} and y_{indep} note the state of reserves and health condition of newly independent young and $n_{offspring}$ the number of offspring which becomes independent per parent per successful reproduction attempt.

If reserves are depleted ($x = x_{min}$) so that the animal is virtually dead, the optimal payoff value is set to

$$V_n(x_{min}, y, e, a, o, s, t) = 0, \tag{2.18}$$

2.1.2.5 Terminal condition

The terminal condition represents the long-term reward at the end of the considered time horizon. It is the starting point for the backward iteration when computing the optimal behavioural strategy. The optimal strategy is determined such that the population is assumed to be of stable size so that

$$V_0(x, y, e, a, o, s, t) = \begin{cases} 0, & x = x_{min}, \\ 1, & x > x_{min}. \end{cases} \tag{2.19}$$

2.1.2.6 Wrap-around condition

The wrap-around condition ensures temporal continuity setting payoff from the final time epoch of one cycle equivalent to payoff from the first time epoch of the next cycle. It is given by

$$V_n(x, y, e, a, o, s, T) = V_{n-1}(x, y, e, a, o, s, 0)\,. \tag{2.20}$$

Note that n represents the number of cycles back from the terminal point of time.

2.1.2.7 Reproductive value

The reproductive value is a relative measure that can be used to compare the value of different individuals or rather state combinations. Here, it determines the reproductive value of any animal relative to that of a particular animal with maximum energy reserves, in top health condition and with offspring of maximum age at the end of the year:

$$V_n(x, y, e, a, o, s, T) = \frac{V_n(x, y, e, a, o, s, T)}{V_n(x_{max}, y_{max}, e_{max}, a_{max}, 1, 0, T)}\,. \tag{2.21}$$

2.1.2.8 Convergence

Under the optimal (evolutionary stable) strategy the reproductive value of being in a certain state only depends on the decision epoch within the environmental cycle and becomes independent of the cycle number on convergence. Thus, the convergence factor λ_n is defined in sOAR, which is computed at the end of each cycle n as

$$\lambda_n(x, y, e, a, o, s, 0) = \frac{V_n(x, y, e, a, o, s, 0)}{V_{n-1}(x, y, e, a, o, s, 0)} \tag{2.22}$$

for all state combinations which had a reproductive value V_{n-1} larger than zero at the end of the previous cycle. Iterations are stopped when

$$\max_{\vec{x}} |\lambda_n(x, y, e, a, o, s, 0) - \lambda_{n-1}(x, y, e, a, o, s, 0)| < \epsilon \tag{2.23}$$

for these state combinations, where the criterion value ϵ is a small user-defined number so that the reproductive value of each particular state changes only marginally between years on convergence.

2.2 Design Concepts

2.2.1 Theoretical background

General concepts
sOAR is grounded in integrated life history theory, employing the mathematical method of stochastic dynamic programming to derive optimal life-history strategies for predicting the behaviour of animals depending on their state and prevailing environmental conditions. The optimal life history strategy represents a decision rule for choosing time- and state-dependent behavioural activities that maximizes the expected number of lifetime surviving offspring and is thus favoured by natural selection. Hereby, all sequential behavioural events in an animal's life are considered as a single instance of the optimal strategy [HM99, HG12]. The common currency for evaluating different behavioural actions is the expected number of offspring far into the future, which serves as a proxy for biological fitness. A population following the computed optimal strategy will be of stable size, given the model assumptions, since evolutionary stability is a side-constraint in the model.

Assumptions
A basic model assumption is that evolution optimizes animal behaviour with respect to temporal trade-offs and certain trade-offs between different behavioural options. It is further assumed that animals can perceive the state of their environment and their own body and that physiological, functional and developmental processes of the animal are linked to environmental influences and each other as described in Sections 2.1.2 and 2.3. The environment is supposed to be cyclic. Using SDP, it is presumed that the optimal decision in a particular state-time combination solely depends on the state at that time step and is independent of previous states or decisions. Finally, since an optimal evolutionary stable strategy is computed, behaviour in the modelled system is strictly speaking viewed as being at an evolutionary end-point.

2.2.2 Individual decision-making

The subjects of decision-making are animals of the species or of the functional type that is defined by the user through the biological parameters and functions in the settings file (see sample configuration file in Section 4.1.2 and Tables 3.2 and 3.1 in Section 3.2). The objects of decision-making are behavioural actions regarding foraging, reproduction and optionally migration. Hereby, two levels of decision-making are included. First, the optimal foraging intensity for each potential behavioural action is considered. Then, there is a selection process where the action with maximum benefit under the optimal foraging intensity is selected from the available set of actions.

The optimal behavioural action for any potential state-time combination, i.e. the optimal strategy or decision rule, is computed by backward iteration. Subsequently, a forward iteration can be run in which an individual, or a cohort of individuals respectively, follows this strategy.

At each decision epoch throughout the cyclic period, the model animal can choose between various

actions depending on its state. In case it is not caring for offspring, it can remain in the same location and start reproduction, it can remain in the same location and not start reproduction or, given the migration option is enabled, it can migrate to the other location. An animal that is currently caring for its offspring can choose between two actions. It can either decide to abandon the offspring or to continue parental care. If applicable, parental care also comprises incubation time. Abandoned offspring is destined to die in the model. Offspring of maximum age will immediately become independent so that the parental bird is forced to abandon them. If the migration option is available, a migrating individual must continue migration until it reaches the last week of its migration. Then it will be forced to stop migration by the next decision epoch.

In addition to the above options of behavioral actions, any model animal also has certain foraging options. An animal that is not migrating can choose a level of foraging intensity $0 \leq u \leq 1, u \in \mathbb{R}$ which represents the proportion of time spent on collecting food. For migrating animals, a constant energy intake during migration needs to be prorated with the user-defined migration costs instead (see also Section 2.3.4).

Animals adapt their behaviour to changing endogenous and exogenous state variables in accordance with the computed optimal strategy. At the beginning of each time-step, the own state is evaluated and then the behavioural action performed that results in the maximum expected number of offspring far into the future under the given environment as determined by the optimal strategy.

Spatial aspects only play a role if the migration option is available to the animals. Then, the animal can decide to stay in one of two locations or to migrate between them. The locations might differ in environmental food availability and prevailing wind conditions. However, the environment can only vary within a period but not between periods. Therefore, though the time epoch within a period matters in the decision process, the period itself does not. As [HM99] point out, using this approach is reasonable if fluctuations between periods are small or if the studied organism is long-lived.

Individual decision-making is influenced by uncertainty regarding condition dependent mortality, predation dependent mortality, growth of experience and environment dependent energetic intake and migratory costs.

When determining the optimal strategy in the backward iteration, actions are evaluated with respect to their expected fitness value. For details, also see Section 2.1.2.

2.2.3 Learning

Decisions in the model depend on the state variable of experience e which is potentially increasing. Such an increase of experience could be interpreted as learning, though growth can be modelled in the same way. In the model, experience e determines the foraging efficiency and thus maximum possible energetic uptake of young animals compared to more experienced ones (also see Section 2.3.1). Newly independent offspring has the minimum level of experience ($e = 0$) while fully matured individuals

posses a user-defined maximum experience ($e = e_{max}$). In the model, experience will take values within the defined range:

$$0 \leq e \leq e_{max}, e \in \mathbb{Z}. \tag{2.24}$$

Hereby, experience grows from zero experience $e = 0$ of newly independent offspring by one unit per decision epoch with the user-defined probability p_{exp} until the maximum experience e_{max} is reached. The probability p_{exp} could be set such that on average the individual will reach full experience by the maturation age of the focal species. No collective learning or other form of individual learning is implemented in the model.

2.2.4 Individual sensing

Animals can only perceive their own state and the state of the environment (food availability and optionally wind conditions) but not the state of other animals. If migration is enabled, an inherent knowledge on the environmental conditions at the destination and along the route is assumed. Potential errors in the sensing process are neglected. Mechanisms by which animals obtain information are not modelled explicitely. Neither are any costs for cognition or for collecting information included in the model.

2.2.5 Individual prediction

In the model, animals use the stage of time and corresponding environmental conditions, their own state and the defined dynamics of state variables to predict future conditions and the consequences of their actions. The prediction process comprises some uncertainty regarding growth of experience, predation, development of health condition and actual energy uptake from feeding. Model animals know the probability distribution of these events but not the actual outcome (also see Sections 2.2.9 and 2.3.5).

2.2.6 Interaction

There are no explicit interactions among animals implemented in the model. However, by setting the parameter θ in Equation 2.25 in Section 2.3.1 to a value less than one, experienced individuals will gain more energy from the same intensity of foraging activity than less experienced individuals, which could be interpreted as some form of implicite density-dependence acting on the young.

2.2.7 Collectives

Collectives of individuals are not part of the model.

2.2.8 Heterogenity

All animals in a population are homogeneous with respect to the describing state variables, potential actions and the processes governing their behaviour. Also, all animals follow the same behavioural strategy. However, animals can be heterogeneous with respect to actual state values and therefore perform different optimal behavioural actions at a certain point of time.

2.2.9 Stochasticity

In the model, growth of experience, predation, development of health condition and actual energy uptake from feeding are assumed to be partly random processes. Details on how stochasticity is implemented for these processes can be found in Sections 2.1.2.2 and 2.3 where the development of state variables and the corresponding submodels are described.

2.3 Submodels

We recommend to compare the submodels outlined below with Figure 2.1.

2.3.1 Dynamics of reserves

The dynamics of reserves are determined by the gross energetic gain as well as the energy expenditure in each decision epoch. The gross energetic intake depends on food availability, foraging intensity and experience while energetic expenditure is influenced by metabolism and energetic costs of demanding activities like offspring production, care for offspring or migration. Additionally implemented stochasticity serves to increase biological realism and to avoid potential grid effects (see Section 2.3.5 for details). It shall reflect variability in food supply and foraging success.

2.3.1.1 Gross energetic gain

The gross energetic gain $\gamma(e, o, u, t)$ depends on environmental food availability $g(o, t)$, foraging intensity u and experience e as follows:

$$\gamma(e, o, u, t) = \theta^{e_{max}-e} \cdot g(o, t) \cdot u, \tag{2.25}$$

where e_{max} is the value of maximum experience. The parameter θ with $0 \leq \theta \leq 1$ implicitly quantifies the effect of density dependence acting on the foraging success of juveniles. In particular, θ determines the foraging efficiency of less experienced juveniles compared to fully experienced adults. A higher θ means that experience has lower effect on foraging efficiency and therefore energy intake. As energetic intake cannot be less than zero and as less experienced birds should not gain more energy through foraging than most experienced birds, θ must be from the interval $[0, 1]$. If $\theta = 1$, there is no density dependence acting on juveniles. If $\theta < 1$, young are assumed to be less efficient in foraging than adults resulting in a relatatively lower energy uptake which will increase with growing experience.

Typically, θ is calibrated during computation such that the population follows an evolutionarily stable strategy with $\lambda^* = 1$ and has a stable size (also see [HM99]). This is done within sOAR using the same Brent algorithm as for finding the optimal foraging intensity for each behavioural action.

2.3.1.2 Environmental food availability

Environmental food availability depends on location and time. In sOAR, environmental food supply is cyclic and can either be read in via a CSV file or be defined as a sine curve variant (as e.g. in Figure 4.1). In the latter case, food availability $g(o,t)$ varies according to

$$g(o,t) = \left(\epsilon(o) \cdot \sin\left[\frac{2\pi(t - t_{max}/4)}{t_{max}}\right] + A_{food}(o)\right) \cdot (x_{max} - x_{min}). \tag{2.26}$$

Here, x_{max} and x_{min} denote the maximum and minimum possible level of energy reserves that can be possessed by a model animal being alive. Thus, environmental food availability is expressed from the animal perspective, i.e. as a multiple of the potential maximum energetic uptake by the animal, or of its specific theoretical range of reserves, respectively. The parameter $A_{food} \in \mathbb{R}$ represents the average environmental food availability at a location over the year. It is specific for a location and if it has a value of one, local food availability covers 100% of the animal's potential range of energy reserves. The location dependent parameter $\epsilon \in [-1, 1]$ measures the degree of seasonality in a location. The higher the absolute value of ϵ, the steeper is the food supply curve swinging around its local mean $A_{food}(o)$ and the more do winter and summer differ in food availability. To summarize, the yearly average food availability can be different in the two locations and local food supply can differ between seasons.

2.3.1.3 Energetic expenditure

Energetic expenditure is governed by basal metabolism and energy consuming activities. In the model, energetic costs for basal metabolism and foraging activity, which will always apply to an individual, are separated from the extra energetic costs for straining activities like reproduction or migration.

The energetic expenditure $c(u,x)$ of a subsisting individual is

$$c(u,x) = c_{bmr} \cdot \left(1 + f_{c_{bmr},x}\left(\frac{x}{x_{max}}\right)\right) + f_{c_f,u}(u) \cdot \left(1 + f_{c_f,x}\left(\frac{x}{x_{max}}\right)\right). \tag{2.27}$$

Here, c_{bmr} is the basal metabolic expenditure and $f_{c_f,u}(u)$ is a function describing metabolic costs arising from foraging activity. Typically, this function increases and accelerates with higher foraging activity. Both, basal and activity metabolic costs, might depend on the level of energy reserves, as well. For example, basal metabolic rate has been shown to increase with body mass within species [AP70, DMSV89] and Glazier [Gla05] lists several studies, which found that intraspecific metabolic scaling depends on activity level. Therefore, the function $f_{c_{bmr},x}(x)$ allows to define additional dependencies of basal metabolism on reserves. Additionally, the relationship between activity metabolism

and reserves can be captured with the function $f_{c_f,x}(x)$ (see Chapter 3.2 for supported functional types). If this is not required or wished, these functions need to be set to constant zero.

Considered energy demanding events in an animal's life are the initiation of reproduction (e.g. production of eggs), caring for offspring and migration, which decrease energy reserves but can also affect condition (see also Section 2.3.2). The energetic costs of these events are set through the user-defined constants Δx_{start} for initiating reproduction, $\Delta x_{incubate}$ for incubation, Δx_{care} for caring for offspring, and Δx_{act} and Δx_{pas} for migratory costs of active and passive flight. The constants for these additional metabolic costs apply per decision epoch and need to be defined accordingly. Any migratory energy costs should include potential energy intake during migratory stopovers (also see Chapter 2.3.4). To account for "increased flight costs with progressively heavier fuel burdens" [Ale91], the functions $f_{x,act}$ and $f_{x,pas}$ have been introduced in sOAR to define the relationship between reserves and the energetic costs of active and passive flight (see Chapter 3.2 for supported functional types). Constant extra metabolic costs Δx take the values

$$\Delta x = \begin{cases} \Delta x_{subsist} = 0 \quad \text{if subsisting,} \\ \Delta x_{start} \quad \text{if reproduction is initiated,} \\ \Delta x_{incubate} \quad \text{if eggs are incubated,} \\ \Delta x_{care} \quad \text{if offspring is provisioned,} \end{cases} \tag{2.28}$$

where Δ_{start}, $\Delta_{incubate}$ and Δ_{care} are user-defined.

2.3.2 Dynamics of condition (optional)

In the model, health condition depends on energy expenditure (see section 2.3.1) and potential extra costs of special activities like reproduction or migration as described by Equation 2.4. For example, reproductive effort has been shown to impair the immune system [DADB97, LD00, NE00, SV96]. The immediate effect ($\alpha(c(u,x))$) of energy expenditure through metabolism and foraging activity on health condition is defined per decision epoch and will follow a user-defined function that can take any form of the supported functional types (see Table 3.1 in Chapter 3). However, it should be ensured that health condition can increase under low metabolic expenditure and decrease under high metabolic expenditure, e.g. by including a sufficiently high constant in the formula for $\alpha(c(u,x))$ (see Chapter 4 and [MWH98] for concrete examples).

2.3.3 Mortality

There are three sources of mortality in the model: predation, death due to low reserves and death due to disease (optional). Predation risk $M_{loc}(x,o,u)$ depends on the animal's level of reserves x, its location o and foraging activity u as follows:

$$M_{loc}(x,o,u) = f_{M_{loc},u}(o,u) \cdot \left(1 + f_{M_{loc},x}\left(\frac{x}{x_{max}}, o\right)\right), \tag{2.29}$$

where $f_{M_{loc},u}(o,u)$ is a user-defined function describing how predation M_{loc} depends on foraging activity u at location o and $f_{M_{loc},x}(x,o)$ is a user-defined function giving the dependency of predation M on the level of reserves x at the location. If the migration option is enabled, these function need to be defined for both locations. The reasoning behind associating predation risk to foraging activity is that higher activity levels lead to enhanced exposure to predation. Additionally, a high level of energy reserves or fat might be hindering in potential escape movements. Please note that the function $f_{M_{loc},x}(x,o)$ can also be set to constant zero (see Table 3.1 for other supported functional types).

If the state variable for health condition is enabled, mortality that is connected to the state of the immune system or health condition will occur in the model. In particular, lower condition will lead to higher probability of death from disease and vice versa. When health condition reaches its critical minimum level y_{min}, the animal will die. McNamara et al. [MWH98] describe the interdependence between health status and disease trough a decreasing power function. In the model, we keep the shape of this function but allow to adjust its parameters to a different background mortality reflecting another life expectancy of the animal. Then, the average probability $D(y)$ of death due to disease at time t is

$$D(y) = M_{bgdis} + (1 - M_{bgdis}) \cdot \left(1 - \left(\frac{1}{y_{max}}\right)\ y\right)^8 \tag{2.30}$$

for an animal with health condition y at time t. Hereby, the background mortality M_{bgdis} due to disease is a user-defined parameter which should correspond to the maximum lifespan of an animal that is in top shape throughout its life. If $y = y_{max}$, i.e. the animal is in top condition, then disease risk equals the user-defined background mortality M_{bgdis}. If $y = 0$, disease risk $D(y)$ equals one and the animal dies. Given the health variable is not enabled, so that M_{bgdis} will not come into effect, an appropriate background mortality needs to be included in the response function for predation (cf. Section 2.3.3.

Survival $S(x,y,o,u)$ then depends on local predation and disease risk as follows:

$$S(x,y,o,u) = [1 - M_{loc}(x,o,u)][1 - D(y)]. \tag{2.31}$$

Starvation occurs when an individual's energy reserves fall to the minimum value x_{max}. In case a parent individual dies for any reason, its dependent offspring will immediately die, as well.

2.3.4 Migration

Migration occurs between two locations which can differ in their local food availability, predation risk and the probability of active flight due to prevailing wind conditions. The duration of migration is an arbitrary number of complete decision epochs. However, it needs to be completed once started. During migration, animals experience the following predation risk M_{migr}:

$$M_{migr}(x) = M_{bgmigr} \cdot \left(1 + f_{M_{bgmigr},x}\left(\frac{x}{x_{max}}\right)\right). \tag{2.32}$$

Here, M_{bgmigr} is a user-defined constant for the background mortality during migration. The function $f_{M_{bgmigr},x}(x)$ describes how predation risk during migration depends on reserves. It can be of any of the supported functional types (see Table 3.1). The idea behind its introduction is that animals with higher fat reserves might spend more time being exposed to potential predators during feeding, and additionally might be less agile and fast when escaping predators ([Lim86, GGP95]).

Migration further reduces the level of energy reserves by the user-defined constants Δx_{act} when active flight mode is emploid and Δx_{pas} in passive flight mode. To compute the respective shares of active and passive flight, the probability that each of these modes is emploid needs to be given to the model. The user can either define a constant probability of active (p_{act}) and passive (p_{pas}) flight or read in time-dependent values as a csv file. Additionally, flight costs can be mass-dependent based on a user-defined function of the supported functional types. The change in energy reserves during migration Δx_{migr} then is a weighted average of the costs for active and passive flight:

$$\Delta x_{migr} = p_{act} \cdot \Delta x_{act} \cdot (1 + f_{x,act}(x)) + (1 - p_{act}) \cdot \Delta x_{pas} \cdot (1 + f_{x,pas}(x)), \tag{2.33}$$

where $f_{x,act}(x)$ and $f_{x,pas}(x)$ are functions capturing the reserve dependency of active and passive flight costs. Please note that there is no optimization of foraging intensity for the migration option. Instead, the given migratory costs must account for potential energetic gains through foraging during migration. Thus, a constant energy intake during migration needs to be prorated with the migration costs by the user, e.g. by calculating energy intake from the mean daily foraging time during migration and substracting this value from the original migration costs.

If applicable, health condition will decrease by the user-defined constant Δy_{act} when migrating in active flight mode and by Δy_{pas} in passive flight mode. For computing the total cost of migration on health condition, costs of active and passive flight are again weighted according to the their user-defined probability:

$$\Delta y_{migr} = p_{act} \cdot \Delta y_{act} + (1 - p_{act}) \cdot \Delta y_{pas}. \tag{2.34}$$

The probability that active or passive flight is used by the focal animal will depend on biological constraints of animal type and/or wind conditions. It can be constant or time dependent.

2.3.5 Additional stochasticity

Optimization models are prone to grid effects and insufficient stochasticity in the model may lead to unrealistic results. Additionally, during computation of the optimal annual routine, new values of energy reserves and health condition usually are no valid grid points. Thus, Houston and McNamara (1999) introduced a procedure to interpolate between grid points and to add additional stochasticity. This procedure as implemented in the model is outlined here (cf. [HM99] for details). It ensures that the new level of energy reserves and health condition (if applicable) is a valid node in the grid of the discretized state variable but, at the same time, is not uniquely determined by the selected behavioural

decision.

For energy reserves x, the routine computes the four closest feasible grid points surrounding the computed new deterministic value x_{t+1}: the second lower node x_{ln2}, the first lower node x_{ln1}, the first upper node x_{un1} and the second upper node x_{un2} by default. Whereas for health condition, by default, only the two surrounding nodes y_{ln1} and y_{un1} are used. Further, the routine assigns a probability to each of these outcomes. Please note, that the user may select to include additional stochasticity independently for energy reserves and/or health condition by setting the appropriate model parameters in the configuration file.

In order to illustrate the implemented default procedure, we denote the deterministic new value of an arbitrary state variable $\vec{z}$ (that likely will not be a valid grid point) by z. Then, the value of the first lower node next to z is determind by:

$$z_{ln1} = \min\left\{\max\left\{z_{min} + \vec{z}\left(\left\lfloor (z - z_{min}) \cdot \frac{1}{dz} \right\rfloor\right), z_{min}\right\}, z_{max}\right\}, \tag{2.35}$$

where dz denotes the interval size between two grid points, $\lfloor\ \rfloor$ is the floor of a given value and $\vec{z}\left(\left\lfloor (z - z_{min}) \cdot \frac{1}{dz} \right\rfloor\right)$ represents the value of the state variable $\vec{z}$ at the index $\left\lfloor (z - z_{min}) \cdot \frac{1}{dz} \right\rfloor$. The minimum and maximum are taken to ensure that the resulting value is within the allowed range. Then the second lower node is

$$z_{ln2} = \max(z_{ln1} - dz, z_{min}) \tag{2.36}$$

and the first and second upper nodes are

$$z_{un1} = \min(z_{ln1} + dz, z_{max}) \tag{2.37}$$

$$z_{un2} = \min(z_{un1} + dz, z_{max}). \tag{2.38}$$

Afterwards, the probabilities that the values of these nodes are taken is calculated using the distance of the node to the deterministic new value. The closer the deterministic new value is to a grid point, the higher the probability that it is taken at the next time step. Hereby, sOAR first determines the probability $p_{z,ln1}$ and $p_{z,un1}$ of a new state value for the case that only the two closest surrounding node values (z_{ln1} and z_{un1}) can be taken, as it is the case for health condition:

$$p_{z,un1} = \frac{(z - z_{ln1})}{dz}, \tag{2.39}$$

$$p_{z,ln1} = 1 - p_{z,un1}. \tag{2.40}$$

For energy reserves, sOAR then uses these values to determine the probabilities $\hat{p}_{z,ln2}$, $\hat{p}_{z,ln1}$, $\hat{p}_{z,un1}$, $\hat{p}_{z,un2}$ of the four closest surrounding node values that could be taken at the next time step. Additionally, the degree of stochasticity can be controlled by the user through a factor $\beta \in]0, \frac{1}{3}[$ where higher

β results in more evenly distributed probability values. Accordingly:

$$\hat{p}_{z,ln2} = \beta p_{z,ln1} \tag{2.41}$$
$$\hat{p}_{z,ln1} = (1-2\beta)p_{z,ln1} + \beta p_{z,un1} \tag{2.42}$$
$$\hat{p}_{z,un1} = \beta p_{z,ln1} + (1-2\beta)p_{z,un1} \tag{2.43}$$
$$\hat{p}_{z,un2} = \beta p_{z,un1}. \tag{2.44}$$

Finally, in the backward iteration, the reproductive value of the new state at time $t+1$ is averaged over the computed potential new state values of reserves and health according to their probabilities as follows:

$$\begin{aligned} V_(\vec{x}, t+1) = {} & \hat{p}_{x,ln2} \cdot p_{y,ln1} \cdot V(x_{ln2}, y_{ln1}, ..., t+1) + \hat{p}_{x,ln2} \cdot p_{y,un1} \cdot V(x_{ln2}, y_{un1}, ..., t+1) \\ & + \hat{p}_{x,ln1} \cdot p_{y,ln1} \cdot V(x_{ln1}, y_{ln1}, ..., t+1) + \hat{p}_{x,ln1} \cdot p_{y,un1} \cdot V(x_{ln1}, y_{un1}, ..., t+1) \\ & + \hat{p}_{x,un1} \cdot p_{y,ln1} \cdot V(x_{un1}, y_{ln1}, ..., t+1) + \hat{p}_{x,un1} \cdot p_{y,un1} \cdot V(x_{un1}, y_{un1}, ..., t+1) \\ & + \hat{p}_{x,un2} \cdot p_{y,ln1} \cdot V(x_{un2}, y_{ln1}, ..., t+1) + \hat{p}_{x,un2} \cdot p_{y,un1} \cdot V(x_{un2}, y_{un1}, ..., t+1) \end{aligned} \tag{2.45}$$

In the forward iteration, the calculated probabilities are directly returned to the main computational routine for determining the probability distribution of state variables at the next time step.

The level of energy reserves and health condition as well as the reproductive value of offspring that just got independent is assigned in the same manner.

In general, if additional stochasticity is switched on for a variable, it can take the value of one of four surrounding grid nodes, otherwise it will take one of two values. The reproductive value of the new state at the next time step will be cmoputed accordingly.

3 Using sOAR

In this chapter, we describe how to use sOAR. The program is provided in two different distributions along with a user manual, an illustrative example application and an R and Matlab sample script to process the output (see supporting information).

After the following short introduction on the flow of use of sOAR (see Figure 3.1), we will note important implementation details (3.1). Subsequently, the required inputs (3.2) will be described, followed by instructions on how to use sOAR (3.3). The chapter closes with a description of the outputs of the model (3.4).

The general flow of use of sOAR including required inputs and produced output files is illustrated in Figure 3.1.

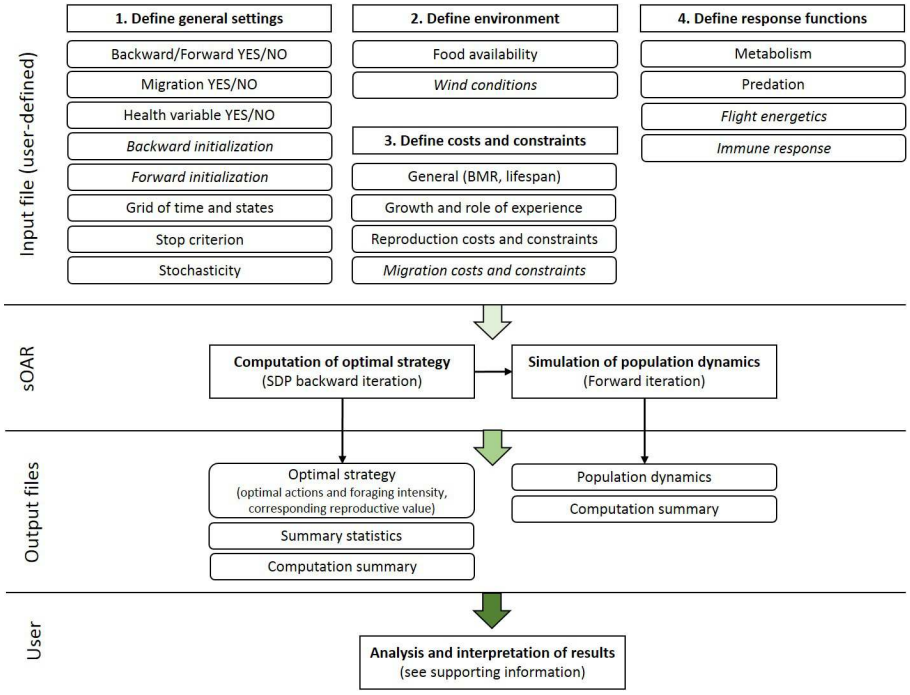

Figure 3.1: Flow of use of sOAR with required input information and resulting output files. Optional settings are italicised. It is possible to run backward and forward iteration independently from each other, given an optimal strategy is provided for the forward iteration. A configuration file for the illustrative example and for processing the resulting output are provided in the supporting information.

Though backward and forward iteration are typically run consecutively, they can also be run independently from each other in sOAR, given an optimal strategy is provided for the forward iteration. The backward iteration of sOAR computes the optimal behavioural strategy for the user-defined setting resulting in multiple multi-dimensional arrays giving the optimal behavioural action, optimal foraging intensity and reproductive value for each possible state combination and each decision epoch. These arrays are saved in binary format (see provided Matlab and R scripts for how to read and process the output). In the forward iteration, the model animals behave according to the computed (or provided) optimal strategy and are followed forward through time. Hereby, a steady state distribution of probabilities over states is computed. This distribution gives the probability that an animal following the provided strategy is in a particular state at a certain time for all potential state-time combinations. Alternatively, the results can be interpreted as the proportion of a population following the optimal strategy that is in any of the potential state-time combinations. Then, they allow making statements about the timing of behaviour, the proportion of a population performing a certain behaviour, the age structure of the population, the development of mean reserves and condition and about patterns of mortality (see sections 3.2 and 3.4 for details). Again, the results are saved as a multidimensional array in binary format.

3.1 Implementation Details

The sOAR model has been implemented as a standalone application for *Microsoft Windows* in C++ and incorporates a Brent optimizer based on Brent [Bre73], the *libconfig* library for processing structured configuration files Version 1.5 by Mark Lindner, Daniel Marjamäki, Andrew Tytula, Glenn Herteg and Matt Renaud (http://www.hyperrealm.com/libconfig) and code by Karsten Isakovic for higher performance und more user-friendliness.

It is provided in two different distributions:

1. Binary distribution
2. Source distribution

The ready-to-use binary distribution contains an executable program, a sample configuration file and the required library for processing the configuration file. Please note that this version only works on Microsoft Windows systems. The source distribution contains the source code, as well as a sample configuration file. The source distribution is intended for users who wish to work with the code itself, e.g. to check implementation details, to adjust it to specific needs or to compile the program for UNIX. We also provide ready solutions for the integrated development environments *Microsoft Visual C++ 2010 Express* and *Microsoft Visual C++ 2010 Express*, which are both freely available from the internet. These solutions can also be imported into later propriatory versions of *Microsoft Visual C++*. Additionally, a copy of this manual is included in both distributions for guidance.

Both sOAR distributions are freely available at *https://sourceforge.net/projects/soar-animal-behaviour/* and can be used under the terms of the *GNU General Public License*, version 3 (*GPLv3*) or any later

version published by the Free Software Foundation (see *https://www.gnu.org/licenses/licenses.en.html*).

Details on how to work with sOAR can be found in section 3.3.

3.2 Inputs

3.2.1 General

In order to run the sOAR software, a valid configuration file needs to be created in a text editor and saved with the filename extension .cfg (e.g. *configFilename.cfg*). This configuration file specifies all user-defined settings such as general settings, environmental conditions, costs and constraints of behaviour and certain response functions capturing biological processes like metabolism. Figure 3.1 includes an overview of the parameters and functions to be set by the user, whereas Table 3.2 lists all user-defined parameters in detail, including a short description of the parameter and the expected data type. In general, parameters are defined as follows:

```
Parametername = <someValue>;
```

For example, the maximum level of reserves (x_{max}) could be set to a value of ten with

```
ReservesMax = 10.0;
```

An example of a valid configuration file can be found in Chapter 4.1.2. Note that depending on the general settings some settings defining e.g. migration costs or the immune response are optional (marked by italic script in Figure 3.1 and Table 3.2).

3.2.2 Response functions (FuncTypes)

For defining the response functions, the data type *FuncType* was created in sOAR. This data type allows to define constant, linear, quadratic, hyperbolic, sigmoid or exponential response functions. Depending on the selected function type, a different number of function parameters must be passed to the function by the user. The general scheme of definition in the configuration file is:

```
Parametername = ( "FuncType, ( a, b, ...  )  );
```

However, the empty space between entities is not obligatory. For example, assuming a linear reserve dependency of basic metabolism could be written in the configuration file as:

```
ReserveDependencyOfBMR = ("Linear", (0,0.05));
```

Table 3.1 provides an overview of the allowed functional types.

Table 3.1: Functional types implemented in sOAR.

Functional type	Number of parameters	General formula	Call in configuration file
Constant	1	$F(x) = a$	("Constant",(a))
Linear	2	$F(x) = a + bx$	("Linear",(a,b))
Quadratic	3	$F(x) = a + bx + cx^2$	("Quadratic",(a,b,c))
Hyperbolic	2	$F(x) = ax/(x + b)$	("Hyperbolic",(a,b))
Sigmoid	2	$F(x) = ax^2/(x^2 + b)$	("Sigmoid",(a,b))
Exponential	2	$F(x) = ax(e^{bx}/e^b)$	("Exponential",(a,b))

3.2.3 Environmental food availability

There are two options each for defining environmental food availability and the probability of active flight as determined by wind conditions.

Environmental food availability can either be read in from file or be specified directly in the configuration file. To read in food availability from file, the command

```
FoodSupplyCSV = "filename.csv";
```

must be used in the configuration file. The food file must have CSV format and specify food supply for each decision epoch and location. Hereby, the first coloumn counts through the number of decision epochs and then each additional coloumn provides the corresponding food supply at a particular location. The first row represents the table head naming each coloumn. Entries in each row are separated per semicolon. Then, the file content looks like:

```
Week;Northern Location;Southern Location
1;0;-0.40
2;0;-0.39
3;0;-0.38
...;...;...
```

Alternatively, food availability can be defined using sOAR's internal food sine function that is characterized by the parameters

- `AverageFoodSupply`
- `FoodSeasonality`.

These parameters determine the average food supply (i.e. sinusoidal axis) and its degree of seasonality (i.e. amplitude of sine curve) over one complete time cycle. See Section 2.3.1 for details or Figure 4.1 for an example and compare with sample script above.

3.2.4 Probability of active flight

The probability of active flight p_{act} can also be read in from file or be set constant directly in the configuration file. Again, the file must be saved in CSV format and use a semicolon as delimiter character to separate the values in each line, or the fields in each data record, respectively. Also, the first row is reserved for the table head and the first coloumn has to count through the number of decision epochs. If the duration of migration is exactly one decision epoch, p_{act} only needs to be defined for each location. Hereby, one coloumn represents the probability of active flight (or wind conditions) at one location. If migration spans several decision epochs, additional coloumns (i.e. $s_{migr} - 1$ coloumns, or duration of migration $- 1$ respectively) need to be inserted that specify p_{act} en route. These can be imagined as stopover sites or major regions along the migratory route with a distance of one decision epoch to the next location or stopover site. Please note that coloumns specifying p_{act} must be arranged in spatially consecutive order. For example, the file content of a CSV file defining the probability of active flight for a migratory bird that travels two decision epochs, always experiences favourable wind conditions in the wintering location (*Loc1*) allowing passive flight and that can increasingly employ passive flight in its remaining habitat (*Loc2*, *Stop1*) with approaching spring could look like this:

```
Week;Loc1;Stop1;Loc2
1;0.0;1.0;1.0
2;0.0;1.0;1.0
3;0.0;1.0;1.0
4;0.0;1.0;1.0
5;0.0;0.9;1.0
6;0.0;0.9;1.0
7;0.0;0.8;0.9
8;0.0;0.8;0.9
...  and so on ...
```

Alternatively, if the probability of active flight is not considered to be time and location dependent, p_{act} can also be set constant using the command

```
ProbabilityOfActiveFlight = <someValue>;
```

in the configuration file.

Table 3.2: User-defined parameters

General settings	
Presettings	
RunBackward	
type:	boolean
symbol:	n.a.
description:	Option to run backward iteration for computing the optimal behavioural strategy in the user-defined scenario.
RunForward	
type:	boolean
symbol:	n.a.
description:	Option to run forward iteration simulating population dynamics for a given or computed strategy.
EnableMigrationOption	
type:	boolean
symbol:	n.a.
description:	Option enabling migration. If set to 'true', animals may migrate between two locations for which environment needs to be defined.
EnableHealthDimension	
type:	boolean
symbol:	n.a.
description:	Option for activating the additional state variable of health condition y (seee Section 2.1.1). If set to 'false' health condition will be set constant ($y = x_{max}$) by sOAR and not affect behavioural decisions.
Backward initialization	
BackwardMaximumNumberOfIterations	
type:	integer
symbol:	n
description:	Maximum number of runs of backward iteration after which computations will be stopped even if results have not converged yet.
BackwardFilePrefix	
type:	string
symbol:	n.a.
description:	Prefix of filenames for results output of backward iteration after convergence. Parameter needs to be set in double quotation marks.
BackwardCalibrateTheta	
type:	boolean
symbol:	n.a.

description: Option to calibrate factor $\theta \in [\theta_{\text{min}}, 1]$ that captures differences in foraging efficiency between juveniles and adults during computations such that the computed strategy is evolutionary stable (see Sections 2.1.1.1, 2.3.1.1 and [HM99]). If set to 'false' a fixed value of θ must be defined by the user.

BackwardCalibrateThetaMin

type: double

symbol: n.a.

description: Possibility to set a biologically realistically minimum value of $\theta \in]0, 1]$. If $\theta_{\text{min}} = 1$, inexperienced individuals have the same foraging success as fully experienced ones, whereas $\theta_{\text{min}} < 1$ results in comparatively lower energy intake of less experienced individuals for the same foraging activity (see Sections 2.1.1.1, 2.3.1.1 and [HM99]).

Forward initialization

ForwardMaximumNumberOfIterations

type: integer

symbol: n

description: Maximum number of runs of forward iteration.

ForwardFilePrefix

type: string

symbol: n.a.

description: Prefix of filenames for results output of forward iteration after convergence. Parameter needs to be set in double quotation marks.

ForwardStartEpoch

type: integer

symbol: n.a.

description: Decision epoch of the periodic cycle in which forward iteration shall start. Recommended is an epoch around the expected reproduction phase.

ForwardStartLocation

type: integer $\in [1; 2]$

symbol: n.a.

description: Location in which forward iteration shall start. Potential values are 0 or also 1 if migration is enabled. Recommended is the expected location of reproduction.

Grid of time and states

DecisionEpochsPerPeriod

type: integer

symbol: t

description: Number of decision epochs per environmental cycle at which a behavioural action has to be taken.

ReservesMin

type: double

symbol: x_{min}
description: Minimum level of energy reserves at which death will occur.

ReservesMax
type: double
symbol: x_{max}
description: Maximum level of energy reserves or energy capacity respectively beyond which no further energy uptake is possible.

ReservesSubdivisions
type: integer
symbol: n.a.
description: Grid resolution for energy reserves, i.e. number of intervals between x_{min} and x_{max}.

HealthLevelMin
type: double
symbol: y_{min}
description: Minimum level of health condition at which death due to diesease will occur.

HealthLevelMax
type: double
symbol: y_{max}
description: Maximum level of health condition beyond which it cannot increase.

HealthSubdivisions
type: integer
symbol: n.a.
description: Grid resolution for health condition, i.e. number of intervals between y_{min} and y_{max}.

ExperienceMax
type: integer
symbol: e_{max}
description: Maximum level of experience individuals can gain. Experience will increase in integer steps from 0 to e_{max} with probability p_{exp} per decision epoch (see Section 2.1.1) and may be connected to foraging efficiency (see Section 2.3.1).

AgeOfIndependence
type: integer
symbol: a_{max}
description: Age at which offspring becomes independent from the parent, in whole number of decision epochs.

NumberOfLocations
type: integer $\in [1, 2]$
symbol: o_{max}

description: Number of locations; value depends on whether the migration option is enabled or not.

Numerical parameters

ConvergenceCriterion
- *type:* double
- *symbol:* n.a.
- *description:* Maximum difference of convergence parameter λ between cycles to stop iteration (see Section 2.1.2.8).

AdditionalStochasticityReserves
- *type:* boolean
- *symbol:* n.a.
- *description:* Option to include additional stochasticity affecting reserves (see Section 2.3.5). Default value is `true`, requiring parameter StochasticityFactor to be set.

AdditionalStochasticityHealth
- *type:* boolean
- *symbol:* n.a.
- *description:* Option to include additional stochasticity affecting health (see Section 2.3.5). Default value is `false`.

StochasticityFactor
- *type:* double $\in]0, 1/3[$
- *symbol:* β
- *description:* Factor for additional stochasticity affecting reserves (see Section 2.3.5).

Environment

Food

AverageFoodSupply
- *type:* vector of doubles
- *symbol:* A_{food}
- *description:* Local average environmental food supply over the time cycle, one vector entry for each location. Alternatively to setting *AverageFoodSupply* and *FoodSeasonality*, read in a CSV file defining food supply for each location and decision epoch (see Sections 2.3.1 and 4.1.2).

FoodSeasonality
- *type:* vector of doubles
- *symbol:* ϵ
- *description:* Local seasonality of food supply; one vector entry for each location. Alternatively to setting *AverageFoodSupply* and *FoodSeasonality*, read in a CSV file defining food supply for each location and decision epoch (see Sections 2.3.1 and 4.1.2).

FoodSupplyCSV
- *type:* CSV file

symbol: $g(o,t)$
description: Environmental food supply for each decision epoch and each location, saved in a CSV file. Data in the CSV file must be formatted as described in Section 3.2.3. The parameters *AverageFoodSupply* and *FoodSeasonality* will be overwritten if *FoodSupplyCSV* is set.

Wind

ProbabilityOfActiveFlight
type: double
symbol: p_{act}
description: Constant probability of active flight per decision epoch (see Sections 2.3.4, 3.2.4 and 4.1.2). This parameter will be overwritten by time and location dependent values if *ProbabilityOfActiveFlightCSV* is set.

ProbabilityOfActiveFlightCSV
type: double
symbol: p_{act}
description: Time and location dependent probability of active flight, reflecting wind conditions, as defined in the provided CSV file (see Sections 2.3.4, 3.2.4 and 4.1.2).

Costs and constraints

Experience

ProbabilityOfExperienceGrowth
type: double
symbol: p_{exp}
description: Probability that experience grows by one unit within a decision epoch until $e = e_{max}$.

Theta
type: double
symbol: θ
description: Factor capturing implicite density-dependent effects on young by adjusting foraging efficiency depending on experience (see Section 2.3.1 and [HM99]); will be adjusted automatically for an evolutionary stable strategy if *BackwardCalibrateTheta* is set to 'true'.

Reproduction

ReserveCostsOfStartReproduction
type: double
symbol: Δx_{start}
description: Extra energetic cost of initiating reproduction, per decision epoch.

ReserveCostsOfIncubation
type: double
symbol: $\Delta x_{incubate}$
description: Extra energetic cost of incubation/pregnancy, per decision epoch (optional).

ReserveCostsOfCareForYoung
type: double
symbol: Δx_{care}
description: Extra energetic cost of provisioning offspring, per decision epoch.

HealthCostsOfStartReproduction
type: double
symbol: Δy_{start}
description: Extra immune cost of initiating reproduction, per decision epoch.

DurationOfIncubation
type: integer
symbol: d_{incub}
description: Duration of incubation phase, in whole numbers of decision epochs.

NumberOfOffspring
type: integer
symbol: $n_{offspring}$
description: Number of offspring (allocated per asexual or female individual).

FitnessAtIndependence
type: double
symbol: f_{indep}
description: Fitness of offspring in terms of energy reserves and health condition when gaining independence compared to adult in top condition ($x_{indep} = y_{indep} = f_{indep} \cdot x_{max}$).

Migration

DurationOfMigration
type: integer
symbol: s_{migr}
description: Duration of migration in whole numbers of decision epochs.

PredationRiskDuringMigration
type: double $\in [0, 1]$
symbol: M_{bgmigr}
description: Constant predation risk during migration, per decision epoch.

ReserveDependencyOfPredationDuringMigration
type: FuncType
symbol: $f_{M_{bgmigr},x}(x)$
description: Dependence of predation during migration on reserves.

ReserveCostsOfActiveFlight
type: double
symbol: Δx_{act}
description: Extra energetic cost of active migratory flight per decision epoch

ReserveCostsOfPassiveFlight
type: integer

symbol: Δx_{pas}
description: Extra energetic cost of passive migratory flight, per decision epoch.

HealthCostsOfActiveFlight
type: integer
symbol: Δy_{act}
description: Extra immune cost of active migratory flight, per decision epoch.

HealthCostsOfPassiveFlight
type: double
symbol: Δy_{pas}
description: Extra immune cost of passive migratory flight, per decision epoch.

Response functions

Metabolism

BasalMetabolicRate
type: double
symbol: c_{bmr}
description: Basal metabolic rate in units of energy reserves per decision epoch.

ReserveDependencyOfBMR
type: FuncType
symbol: $f_{c_{bmr},x}(x)$
description: Dependence of basal metabolic rate the level of energy reserves.

ActivityDependencyOfMetabolism
type: FuncType
symbol: $f_{c_f,u}(u)$
description: Dependence of metabolism on foraging activity u.

ReserveDependencyOfMetabolism
type: FuncType
symbol: $f_{c_f,x}(x)$
description: Dependence of activity related metabolism on level of energy reserves.

Predation

ReserveDependencyOfPredationLocation1
type: FuncType
symbol: $f_{M_{loc,x}}(x,1)$
description: Dependence of predation at location one on energy reserves.

ReserveDependencyOfPredationLocation2
type: FuncType
symbol: $f_{M_{loc,x}}(x,2)$
description: Dependence of predation at location two on energy reserves.

ActivityDependencyOfPredationLocation1
type: FuncType
symbol: $f_{M_{loc,u}}(x,1)$
description: Dependence of predation at location one on activity level.

ActivityDependencyOfPredationLocation2	
type:	FuncType
symbol:	$f_{M_{loc,u}}(x, 2)$
description:	Dependence of predation at location two on activity level.
Flight energetics	
ReserveDependencyOfActiveFlight	
type:	FuncType
symbol:	$f_{x,act}(x)$
description:	Dependence of costs for active flight on energy reserves.
ReserveDependencyOfPassiveFlight	
type:	FuncType
symbol:	$f_{x,pas}(x)$
description:	Dependence of costs for passive flight on energy reserves.
Immune response	
BackgroundMortalityByDisease	
type:	double
symbol:	M_{bgdis}
description:	Background mortality due to disease. Parameter only has an effect if the health variable is enabled and then affects mortality from disease.
MetabolismDependencyOfHealth	
type:	FuncType
symbol:	$\alpha(c(u, x))$
description:	Immediate health cost of energy expenditure through metabolism and foraging activity.

3.3 Running sOAR

sOAR is primarily designed for *Microsoft Windows* for which it is available as a ready-to-use executable file as well as complete solution for the free integrated development environments *Microsoft Visual C++ 2008 Express* and *Microsoft Visual C++ 2010 Express* or later propriatory versions of *Microsoft Visual C++*. On UNIX systems, sOAR's source code needs to be linked and compiled by the user in order to obtain a running program.

In any case, as a first step for running sOAR, the appropriate distribution for the user's system and demands must be downloaded from *https://sourceforge.net/projects/soar-animal-behaviour/*. Please note that the configuration file (*Schaefer_et_al_2017_A5_sOAR_ConfigFile.cfg*) contains all user-defined settings, whereas processing of the settings, the dynamic programming computational routine and some checkups are realized in the remaining files. How to proceed after the download is described in the following.

3.3.1 Binary distribution

The simplest option for working with sOAR is using the binary distribution under MS Windows and run sOAR in batch mode from the command prompt. The necessary steps for this option are:

1. Download the appropriate package from *https://sourceforge.net/projects/soar-animal-behaviour/*. It is distributed under the terms of the *GNU General Public License*, version 3 (*GPLv3*) or any later version (see *https://www.gnu.org/licenses/licenses.en.html*).

2. Keep the downloaded executable *soar_main.exe*, the dynamic library *libconfig++.dll* for the integrated libconfig package (more information on *http://www.hyperrealm.com/libconfig/*) and the configuration file *Schaefer_et_al_2017_A5_sOAR_ConfigFile.cfg* in the same folder for computations.

3. Adjust the parameters in the configuration file to the focal animal and environmental scenario (see Chapter 2 and Section 3.2 for further information). The name of the configuration file may be changed.

4. If the environmental food supply and/or the probability of active flight shall be read in from a CSV file, add these files to the same folder as the executable.

5. Open the Windows command prompt and navigate to the folder with the above files.

6. Start the program from the command prompt, passing the appropriate configuration file, with the command:
   `soar_main.exe -config config_filename.cfg`
   where config_filename.cfg is to be substituted by the name of the appropriate configuration file.

A batch script can easily be programmed for automizing this procedure.

3.3.2 Source distribution

Alternatively, the program can be run using the integrated development environment *Microsoft Visual C++*. This variant also allows to see and modify the code of the computational procedure. We provide a ready solution for *Microsoft Visual C++ 2008 Express* and *Microsoft Visual C++ 2010 Express* which are both freely available on the internet. The solution can also be imported into the propriatory version of *Microsoft Visual C++* 2008 or later. In the following, we shortly describe how to work with these solutions. However, the user is free to use his preferred development environment or to create an appropriate Makefile for building an executable program from the original source code provided in the subfolder "src" of this distribution.

1. Ensure that *Microsoft Visual C++* or *Microsoft Visual C++ Express* 2008 or later is installed on your computer.

2. Download the appropriate package from *https://sourceforge.net/projects/soar-animal-behaviour/*. It is distributed under the terms of the *GNU General Public License*, version 3 (*GPLv3*) or any later version (see *https://www.gnu.org/licenses/licenses.en.html*).

3. The distribution contains the following folders:

doc	Files for generating a documentation from the annotated C++ sources with doxygen.
extern	External library libconfig (*http://www.hyperrealm.com/libconfig/*) for processing structured configuration files including corresponding author list and license.
src	sOAR's source code. In the subfolder *soar_main* is the source code for the main file, in the subfolder *soar_lib* for the computational routine and in the subfolder *soar_support_lib* for the integrated parameter manager by Karsten Isakovic.
tests	Test files for unit tests.
vs2008	sOAR solution for the IDE *MS Visual C++ 2008 Express*.
vs2010	sOAR solution for the IDE *MS Visual C++ 2010 Express*.

   Further, it includes the following top-level files:

gnu_general_public_license_v3.txt	Copy of the GNU General Public License Version 3.0.
pAct.csv	CSV file defining the probability of active flight for the provided sample application.
Readme.txt	Basic information on sOAR.
sOAR_example.cfg	Sample configuration file for setting the user-defined parameters in sOAR.

4. To open the ready solution, navigate to the subfolder *vs_2008* or *vs_2010* (depending on your version of *MS Visual C++*. Then open *soar_main.sln*. If you possess a later propriatory version of *Microsoft Visual C++*, import the solution in the subfolder *vs_2010* into it.

5. Include the configuration file in the command arguments as shown in Figure 3.2 and described below:

   Right-click on the project *soar_main* in the solution explorer.

   Select Properties, upon which the project property pages open.

   Navigate to *Configuration properties* and there to *Debugging*.

   Select *All Configurations* under *Configuration*.

   Insert the line `-config config_filename.cfg` in the field *Command arguments*.

   Ensure that the *Working Directory* (below) links to the location of the configuration file, or put the file to the specified working directory respectively.

   Click *OK*.

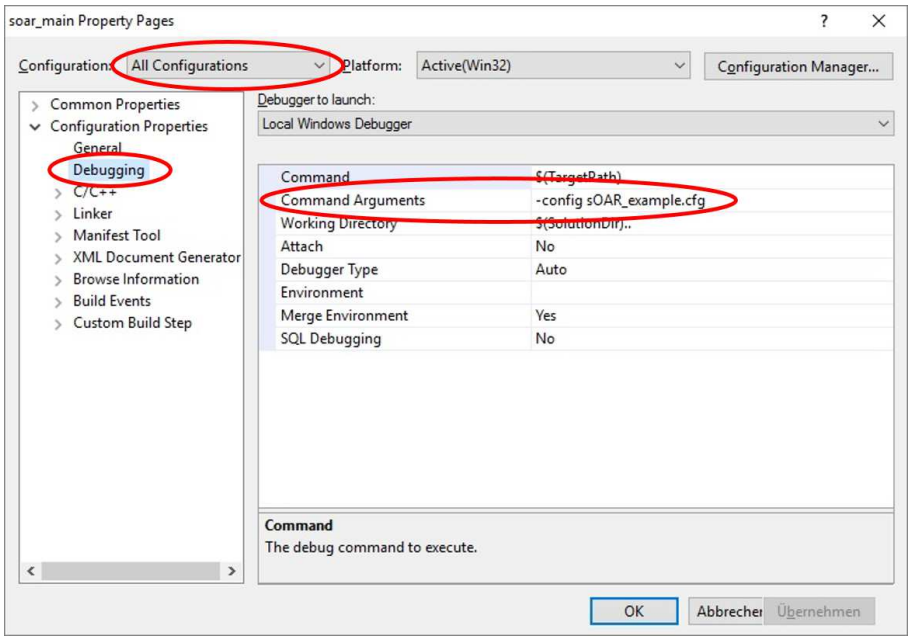

Figure 3.2: Instructions on how to include the configuration file in the command arguments.

6. Check and adjust the code to your needs.

7. To build a new executable file select the preferred configuration in *MS Visual C++ Express* (*Debug* or *Release*) and click on *Start Debugging*.

8. Reuse or distribute the created executable together with the required dynamic library for processing the configuration file.

Please note that the executable always needs to be rebuild any time the source code is changed in order to include these changes.

3.4 Outputs

While the sOAR executable is running, information on the computation is printed to the command prompt in real time. In particular, the user will receive a message on the loaded configuration file and potential errors in here, as well as a summary of the main response function. Subsequently, after each successful cycle n of backward or forward iteration, a respective notification on the computation status including the value of the convergence parameter λ_n is printed to the command prompt. The computations are concluding with a message on the final value of the convergence parameter, the required computation time and a note when all results have been saved. Additionally, after the backward iteration, a summary statistics of the strategy is printed to the command prompt. The console output

allows for real-time control of the computations, as well as a quick check of the given input and results. It could also be redirected to a file if wished.

During computations, data on the computed optimal strategy (backward iteration) and simulated population dynamics under the optimal strategy (forward iteration) is collected. This data is provided to the user in form of text and binary files after convergence for further analysis (see Table 3.3). The following subsections specifically describe the file output. Please note that the endianness of the source system on which the output binary files were created will need to be specified when reading them in on another system. Windows systems have *little* endianness, while Mac systems work with *big-endian*. Further note that the behavioural routine, i.e. the timing of behavioural actions over time, emerges from the model.

Table 3.3: Output files of sOAR and their content. The "*xx*" in file names is a substitute for the user-defined file prefix.

Iteration	**Output file**	**File content**
Backward	*xx_decision_BW.bin*	Optimal behavioural strategy including optimal behavioural actions and foraging intensities, as well as the corresponding reproductive value for all state-time combinations.
Backward	*xx_settings_BW.bin*	Parameter settings used for computing the optimal behavioural strategy (and as read from the configuration file)
Backward	*xx_summary_BW.txt*	Summary file providing for each decision epoch the proportion of a population (or probability of an individual) performing a certain behaviour, as well as the minimum and maximum state levels at which the behaviour is performed.
Forward	*xx_decision_FW.bin*	Behavioural strategy used to compute population dynamics.
Forward	*xx_populationdynamics_FW.bin*	Population dynamics of a model cohort following the provided behavioural strategy.

3.4.1 Model output of backward iteration

Main output of the backward iteration is saved in the file *xx_decision_BW.bin* where "*xx*" in the file name is a substitute for the user-defined file prefix. This file contains the state-dependent optimal behavioral strategy for the focal animal and the corresponding reproductive value in several multi-dimensional arrays. Hereby, the optimal strategy comprises two aspects. First, it provides the optimal

behavioural action for any combination of state and time the model animals can be in. Looking up the optimal behavioural action for an animal which is in a particular state at a certain time clarifies if the individual should start reproduction, care for its potential offspring, abandon its potential offspring, migrate or stay. Second, the computed strategy provides the optimal intensity with which an individual should forage during its optimal behavioural activity.

To import these results into other software for further analysis (e.g. Matlab or R) it is important to know in which order and format sOAR is writing them to the binary file. By default sOAR first writes the array of reproductive values, then the array of optimal foraging intensities and then the array of optimal behavioural decisions to the binary file. Hereby, each array is preceded by information on the number of its dimensions and the size of these dimensions before the array entries, e.g. of the reproductive value, for all state time combinations follow. Subsequently, information related to the convergence of the iteration is appended. Table 3.4 provides detailed information on the output contained in *xx_decision_BW.bin* by default (or as specified in the source file *Decision.cpp* respectively).

Table 3.4: Output contained in *xx_decision_BW.bin*, incl. its order of appearance as specified in the source file *decision.cpp*, data type and size. Note that the "*xx*" in the file name is a substitute for the user-defined file prefix.

Order	Output/Parameter	Data type	Data size in bytes (bits)
1	Number of dimensions (n_{dim}) of following array, i.e. number of state variables	`unsigned int`	$4\,(32)$
2	Vector giving length l_{dim} of each dimension of following array, i.e. number of discrete values of state variables	`unsigned int` array	$4\,(32) \times [l_x,l_y,l_e,l_a,l_o,l_s,l_t]$
3	Array with optimal reproductive value for each state-time combination	`double` array	$8\,(64) \times [l_x \times l_y \times l_e \times l_a \times l_o \times l_s \times l_t]$
4	Number of dimensions (n_{dim}) of following array, i.e. number of state variables	`unsigned int`	$4\,(32)$
5	Vector giving length l_{dim} of each dimension of following array, i.e. number of discrete values of state variables	`unsigned int` array	$4\,(32) \times n_{dim}$
6	Array with optimal foraging intensity for each state-time combination	`double` array	$8\,(64) \times [l_x \times l_y \times l_e \times l_a \times l_o \times l_s \times l_t]$
7	Number of dimensions (n_{dim}) of following array, i.e. number of state variables	`unsigned int`	$4\,(32)$
8	Vector giving length l_{dim} of each dimension of following array, i.e. number of discrete values of state variables	`unsigned int` array	$4\,(32) \times n_{dim}$
9	Array with optimal foraging intensity for each state-time combination	`double`	$8\,(64) \times [l_x \times l_y \times l_e \times l_a \times l_o \times l_s \times l_t]$

10	Average $\lambda_{avg,bwd}$ on convergence in backward iteration	`double`	8 (64)
11	θ	`double`	8 (64)
12	Indication if convergence was reached	`bool`	1 (8)
13	Number of last backward iteration	`int`	4 (32)
14	Convergence factor $\lambda_{state,bwd}$ for the particular state combination $(x_{max}, y_{max}, e_{max}, 0, 0, 0, T)$ on convergence in backward iteration	`double`	8 (64)
15	Convergence factor $\lambda_{worst,bwd}$ with the highest deviation from the intended value of one on convergence in backward iteration	`double`	8 (64)
16	Number of state combinations with $V_{n-1}(\vec{x}, 0) > 0$ that have not converged in backward iteration	`int`	4 (32)
17	Total number of state combinations with $V_{n-1}(\vec{x}, 0) > 0$ in last backward iteration	`int`	4 (32)

Evaluating the optimal strategy, patterns of behaviour can be inferred to answer questions like the following:

- Where should the animals spend their time in various stages of the cyclic time period?
- In which of the given locations should the animals reproduce?
- When should reproduction be initiated?
- In which time frame should reproduction preferably take place?
- When is it advisable to start migrating between the defined locations?
- In which time frame should migration happen?
- Is the spread in the timing of migration different when migrating to one location or the other?

The reproductive value of being in a certain state at a particular stage of the user-defined environmental cycle is measured as a function of state or as a function of time stage within the cycle. It helps answering questions such as:

- What is the reproductive value for given values of energy reserves and health condition under the optimal strategy and how does it change over the course of the user-defined environmental cycle?

- How does reproductive value depend on certain state variables for a given season, e.g. does reproductive value increase more strongly with reserves for a particular level of condition?
- How does reproductive value vary between divisions of the total cycle time for given levels of energy reserves and health condition?
- Can trade-offs and conflicting pressures be explained in terms of reproductive value?

Further, critical values of state variables can be extracted from the optimal strategy. These critical values give the range of state variable values within which a particular behavior is performed. If the value of the state variable is equal to or above the critical value, then the animal will perform the behavioral action. If the value of the state variable is below this critical value, then the animal will select another behavioral action. The critical values of state variables are useful for answering questions like:

- Is there a critical level of condition, reserves or experience for reproduction and migration?
- For which combinations of condition and reserves should an individual with a certain level of experience start reproduction or migrate?
- Do certain state variables have a stronger effect on decisions than others?
- How do critical levels of state variables depend on offspring age? When is offspring abandoned?

Besides this biologically interesting output, the parameter settings used for computing the optimal behavioural strategy (and as read in from the configuration file) are written to the binary file *xx_settings_BW.bin*, next to a summary text file. Here, the "xx" in the file name is again a substitute for the userdefined file prefix. Table 3.5 provides detailed information on how this file is structured by default (or as specified in the source file *Settings.cpp* respectively). The summary file provides the proportion of a population (or probability of an individual) performing a certain behaviour for each decision epoch, as well as the minimum and maximum state levels at which the behaviour is performed.

Table 3.5: Output contained in *xx_settings_BW.bin*, incl. its default order of appearance as specified in the source file *Settings.cpp*, its data type and size. Note that the "*xx*" in the file name is a substitute for the user-defined file prefix. Functional types are specified by 3 `doubles` and 1 `int` (also see 3.1). Output with italic order numbers in the table highlight optional parameters which will only contain meaningful values if they had been set by the user, otherwise they will take their default values.

Order	Output/Parameter	Data type	Data size in bytes (bits)

1	RunBackward	`bool`	1 (8)
2	RunForward	`bool`	1 (8)
3	EnableMigrationOption	`bool`	1 (8)
4	EnableHealthDimension	`bool`	1 (8)
5	BackwardMaximumNumberOfIterations	`unsigned int`	4 (32)
6	Length of backward file prefix; needed to read in BackwardFilePrefix	`unsigned int`	4 (32)
7	BackwardFilePrefix	`char[512]`	1 (8) × 512
8	BackwardCalibrateTheta	`bool`	1 (8)
9	BackwardCalibrateThetaMin	`double`	8 (64)
10	ForwardMaximumNumberOfIterations	`unsigned int`	4 (32)
11	Length of backward file prefix; needed to read in ForwardFilePrefix	`unsigned int`	4 (32)
12	ForwardFilePrefix	`char[512]`	1 (8) × 512
13	ForwardStartEpoch	`unsigned int`	4 (32)
14	ForwardStartLocation	`unsigned int`	4 (32)
15	DecisionEpochsPerPeriod	`unsigned int`	4 (32)
16	ReservesMin	`double`	8 (64)
17	ReservesMax	`double`	8 (64)
18	ReververesSubdivisions	`unsigned int`	4 (32)
19	HealthLevelMin	`double`	8 (64)
20	HealthLevelMax	`double`	8 (64)
21	HealthSubdivisions	`unsigned int`	4 (32)
22	ExperienceMax	`unsigned int`	4 (32)
23	AgeOfIndependence	`unsigned int`	4 (32)
24	NumberOfLocations	`unsigned int`	4 (32)
25	ConvergenceCriterion	`double`	8 (64)
26	AdditionalStochasticityReserves; default 'true'	`bool`	1 (8)
27	AdditionalStochasticityHealth; default 'false'	`bool`	1 (8)

28	StochasticityFactor; only applicable if additional stochasticity switched on	`double`	8 (64)
29	Number $n_{dim,AFS}$ of dimensions of AverageFoodSupply	`unsigned int`	4 (32)
30	Lengths l_{AFS} of dimensions of AverageFoodSupply	`unsigned int` array	4 (32) $\times$ $n_{dim,AFS}$
31	Values of AverageFoodSupply	`double` array	8 (64) $\times$ $[l_{AFS,1} \times ...]$
32	Number $n_{dim,FSS}$ of dimensions of FoodSeasonality	`unsigned int`	4 (32)
33	Lengths l_{FSS} of dimensions of FoodSeasonality	`unsigned int` array	4 (32) $\times$ $n_{dim,FSS}$
34	Values of FoodSeasonality	`double` array	8 (64) $\times$ $[l_{FSS,1} \times ...]$
35	Number $n_{dim,FSCSV}$ of dimensions in CSV file for food supply	`unsigned int`	4 (32)
36	Lengths l_{FSCSV} of dimensions in CSV file for food supply; only applicable if $n_{dim,FSCSV} > 0$	`unsigned int` array	4 (32) $\times$ $n_{dim,FSCSV}$
37	Values of food supply in CSV file; only applicable if $n_{dim,FSCSV} > 0$	`double` array	8 (64) $\times$ $[l_{FSCSV,1} \times ...]$
38	ProbabilityOfActiveFlightConst	`double`	8 (64)
39	Number $n_{dim,PACT}$ of dimensions in CSV file for probability of active flight	`unsigned int`	4 (32)
40	Lengths l_{PACT} of dimensions in CSV file for probability of active flight; only applicable if $n_{dim,PACT} > 0$	`unsigned int` array	4 (32) $\times$ $n_{dim,PACT}$
41	Values of probability of active flight in CSV file; only applicable if $n_{dim,PACT} > 0$	`double` array	8 (64) $\times$ $[l_{PACT,1} \times ...]$
42	ProbabilityOfExperienceGrowth	`double`	8 (64)
43	Theta	`double`	8 (64)
44	ReserveCostsOfStartReproduction	`double`	8 (64)
45	ReserveCostsOfDurationOfIncubation	`double`	8 (64)
46	ReserveCostsOfCareForYoung	`double`	8 (64)
47	HealthCostsOfStartReproduction	`double`	8 (64)
48	DurationOfIncubation	`int`	4 (32)
49	NumberOfOffspring	`int`	4 (32)
50	FitnessAtIndependence	`double`	8 (64)
51	DurationOfMigration	`int`	4 (32)
52	PredationRiskDuringMigration	`double`	8 (64)
53	First parameter in functional type for ReserveDependencyOfPredationDuringMigration	`double`	8 (64)

54	Second parameter in functional type for ReserveDependencyOfPredationDuringMigration	`double`	8 (64)
55	Third parameter in functional type for ReserveDependencyOfPredationDuringMigration	`double`	8 (64)
56	Functional type variant for ReserveDependencyOfPredationDuringMigration	`int`	4 (32)
57	ReserveCostsOfActiveFlight	`double`	8 (64)
58	ReserveCostsOfPassiveFlight	`double`	8 (64)
59	HealthCostsOfActiveFlight	`double`	8 (64)
60	HealthCostsOfPassiveFlight	`double`	8 (64)
61	BasalMetabolicRate	`double`	8 (64)
62	First parameter in functional type for ReserveDependencyOfBMR	`double`	8 (64)
63	Second parameter in functional type for ReserveDependencyOfBMR	`double`	8 (64)
64	Third parameter in functional type for ReserveDependencyOfBMR	`double`	8 (64)
65	Functional type variant for ReserveDependencyOfBMR	`int`	4 (32)
66	First parameter in functional type for ActivityDependencyOfMetabolism	`double`	8 (64)
67	Second parameter in functional type for ActivityDependencyOfMetabolism	`double`	8 (64)
68	Third parameter in functional type for ActivityDependencyOfMetabolism	`double`	8 (64)
69	Functional type variant for ActivityDependencyOfMetabolism	`int`	4 (32)
70	First parameter in functional type for ReserveDependencyOfMetabolism	`double`	8 (64)
71	Second parameter in functional type for ReserveDependencyOfMetabolism	`double`	8 (64)
72	Third parameter in functional type for ReserveDependencyOfMetabolism	`double`	8 (64)
73	Functional type variant for ReserveDependencyOfMetabolism	`int`	4 (32)
74	First parameter in functional type for ReserveDependencyOfPredationLocation1	`double`	8 (64)
75	Second parameter in functional type for ReserveDependencyOfPredationLocation1	`double`	8 (64)

76	Third parameter in functional type for ReserveDependencyOfPredationLocation1	`double`	8 (64)
77	Functional type variant for ReserveDependencyOfPredationLocation1	`int`	4 (32)
78	First parameter in functional type for ReserveDependencyOfPredationLocation2	`double`	8 (64)
79	Second parameter in functional type for ReserveDependencyOfPredationLocation2	`double`	8 (64)
80	Third parameter in functional type for ReserveDependencyOfPredationLocation2	`double`	8 (64)
81	Functional type variant for ReserveDependencyOfPredationLocation2	`int`	4 (32)
82	First parameter in functional type for ActivityDependencyOfPredationLocation1	`double`	8 (64)
83	Second parameter in functional type for ActivityDependencyOfPredationLocation1	`double`	8 (64)
84	Third parameter in functional type for ActivityDependencyOfPredationLocation1	`double`	8 (64)
85	Functional type variant for ActivityDependencyOfPredationLocation1	`int`	4 (32)
86	First parameter in functional type for ActivityDependencyOfPredationLocation2	`double`	8 (64)
87	Second parameter in functional type for ActivityDependencyOfPredationLocation2	`double`	8 (64)
88	Third parameter in functional type for ActivityDependencyOfPredationLocation2	`double`	8 (64)
89	Functional type variant for ActivityDependencyOfPredationLocation2	`int`	4 (32)
90	First parameter in functional type for ReserveDependencyOfActiveFlight	`double`	8 (64)
91	Second parameter in functional type for ReserveDependencyOfActiveFlight	`double`	8 (64)
92	Third parameter in functional type for ReserveDependencyOfActiveFlight	`double`	8 (64)
93	Functional type variant for ReserveDependencyOfActiveFlight	`int`	4 (32)
94	First parameter in functional type for ReserveDependencyOfPassiveFlight	`double`	8 (64)
95	Second parameter in functional type for ReserveDependencyOfPassiveFlight	`double`	8 (64)
96	Third parameter in functional type for ReserveDependencyOfPassiveFlight	`double`	8 (64)

97	Functional type variant for ReserveDependencyOfPassiveFlight	`int`	4 (32)
98	BackgroundMortalityByDisease	`double`	8 (64)
99	First parameter in functional type for MetabolismDependencyOfHealth	`double`	8 (64)
100	Second parameter in functional type for MetabolismDependencyOfHealth	`double`	8 (64)
101	Third parameter in functional type for MetabolismDependencyOfHealth	`double`	8 (64)
102	Functional type variant for MetabolismDependencyOfHealth	`int`	4 (32)

3.4.2 Model output of forward iteration

The forward iteration shows how the optimal behavioral strategy is put into effect in a population following the optimal strategy under the provided setting. Main output of the forward iteration is a multi-dimensional array giving the proportion of a population in a particular state at a certain time for all possible states and time steps of the cyclic period. Alternatively, the array entries can be interpreted as the probability with which an individual will be in the given state combinations at the defined time steps. The array is saved in the binary file *xx_populationdynamics_FW.bin* and completed by information on the iteration's convergence. Hereby, the "*xx*" in file names is a substitute for the user-defined file prefix. Ultimately, the output of the forward iteration allows predictions on population statistics, patterns of mortality and the timing of reproduction and migration.

Table 3.6 details the output contained in the binary file, including its order and format. This information is necessary for importing the results into other software for further analysis (e.g. Matlab or R). By default, sOAR first saves the number of the array's dimensions and their size before the array entries for all state time combinations follow. Subsequently, information related to the convergence of the iteration is appended.

Table 3.6: Output of sOAR's forward iteration in the binary file *xx_populationdynamics_FW.bin*, including its default order of appearance as specified in the source file *Settings.cpp*, its data type and size. The output represents the population dynamics of a model cohort following the provided behavioural strategy. Alternatively, it can be interpreted as the likelihood of an individual animal's behaviour over time. The "*xx*" in file names is a substitute for the user-defined file prefix.

Order	Output/Parameter	Data type	Data size in bytes (bits)
1	Number of dimensions (n_{dim}) of following array, i.e. number of state variables	`unsigned int`	4 (32)
2	Vector giving length l_{dim} of each dimension of following array, i.e. number of discrete values of state variables	`unsigned int` array	4 (32) $\times$ [l_x,l_y,l_e,l_a,l_o,l_s,l_t]
3	Array with proportion of population being in each state-time combination	`double` array	8 (64) $\times$ [$l_x \times l_y \times l_e \times l_a \times l_o \times l_s \times l_t$]
4	Average $\lambda_{avg,fwd}$ on convergence in forward iteration	`double`	8 (64)
5	Indication if convergence was reached	`bool`	1 (8)
6	Number of last forward iteration	`int`	4 (32)
7	Convergence factor $\lambda_{state,fwd}$ for the particular state combination $(x_{max}, y_{max}, e_{max}, 0, 0, 0, T)$ on convergence in forward iteration	`double`	8 (64)
8	Convergence factor $\lambda_{worst,fwd}$ with the highest deviation from the intended value of one on convergence in forward iteration	`double`	8 (64)
9	Number of state combinations with $V_{n-1}(\vec{x}, 0) > 0$ that have not converged in forward iteration	`int`	4 (32)
10	Total number of state combinations with $V_{n-1}(\vec{x}, 0) > 0$ in last forward iteration	`int`	4 (32)

Further analysis of the results of the forward iteration gives information on:

- Population statistics:
  - Development of population size over the cyclic period
  - Distribution of experience
  - Mean reserves and condition as a function of time
- Patterns of mortality:
  - Proportion of population still alive as a function of time
  - Mortality from disease over the cycle time
  - Mortality from predation over the cycle time
  - Mortality of dependent young

- Timing of reproduction and migration
    - Proportion of population starting reproduction as a function of time
    - Proportion of population caring for offspring as a function of time
    - Proportion of population migrating as a function of time

Please note that exemplary Matlab and R scripts for further analysis and visualization of the output are provided in the supporting information.

4 Illustrative Examples

4.1 Timing of reproduction and migration

4.1.1 Background

We illustrate sOAR with an example application by simulating the effect of seasonality in the wintering habitat on the timing of migration and reproduction in a migratory bird.

We used a hypothetical migratory bird to show sOAR's capability to determine a complete life-history strategy based on differing cyclic environmental conditions at two distinct locations representing the breeding and wintering habitat. Ecologically, the example reveals how the degree of seasonality in the wintering habitat can significantly change the optimal timing of migration and reproduction. Moreover, it shows that the optimal timing of behaviour depends on the state of the animal and that different degrees of synchrony in the optimal timing of migration between adults and juveniles might result from different environmental conditions.

Table 4.1: Model parameters and their values or formulas for their computation as used in the illustrative example. Where three values are given, these refer to the three simulated environmental scenarios. Extra costs of reproductive or migratory activities, add to the costs of subsistence when performed in the model.

Parameter	**Symbol**	**Value / Formula**
Basic state variables:		
Range of energy reserves	x	$\in [0, 10],\ \in \mathbb{N}$
Range of health condition	y	$\in [0, 10],\ \in \mathbb{N}$
Range of experience	e	$\in [0, 2],\ \in \mathbb{N}$
Age of any offspring	a	$\in [-1, a_{incub} + a_{indep} + 2],\ \in \mathbb{N}$
Location	o	$\in [1, n_{loc}],\ \in \mathbb{N}$
State of migration	s	$\in [0, s_{migr}],\ \in \mathbb{N}$
Week of year	t	$\in [0, 51],\ \in \mathbb{N}$
General biology:		
Foraging intensity	u	$\in [0, 1]$
Background mortality by disease	M_{bgdis}	0.0016
Basic metabolic cost	c_{bmr}	2.0
Energetic expenditure, subsisting	$c(u, x)$	$c_{bmr} * (1 + 0.01x^2) + 6u^2 * (1 + 0.01x^2)$
Health cost of subsistence	$\alpha(c(u, x))$	$1 - 0.2 \cdot c(u, x) - 0.01 \cdot (c(u, x))^2$

Local environment:		
Number of locations	o	2
Average food availability at site 1	$A_{food}(1)$	1.0
Average food availability at site 2	$A_{food}(2)$	1.0
Degree of seasonality of food at site 1	$\epsilon(1)$	0.1, 0.2, 0.3
Degree of seasonality of food at site 2	$\epsilon(2)$	0.7
Local predation risk	$M_{loc}(x, o, u)$	$0.01 \cdot u^2 \cdot (1 + 0.01 \cdot x^2)$
Reproduction:		
Duration of incubation in weeks	d_{incub}	4
Age of independence in weeks	a_{indep}	6
Energetic cost of starting a brood	Δx_{start}	0.5
Energetic cost of incubation	Δx_{incub}	1.5
Energetic cost of brood care	Δx_{care}	7.0
Health cost of starting a brood	Δy_{start}	1.0
Number of offspring	$n_{offspring}$	2
Fitness of offspring at independence	f_{indep}	0.5
Foraging efficiency of inexperienced bird ($e = 0$)	θ	$0.72, 0.76, 0.81$
Probability of increasing experience	p_{exp}	0.02
Migration:		
Duration of migration in weeks	s_{migr}	3
Energetic cost of active flight	Δx_{act}	2.0
Reserves' influence on flight cost	$f_{x,act}(x)$	$0.001x^2$
Health cost of active flight	Δy_{act}	1.0
Predation risk during migration	$M_{migr}(x)$	$0.01 \cdot (1 + 0.01x^2)$

In the example, both the health variable and the migration option were enabled. Biological parameter settings (see Table 4.1) were oriented at a medium-sized long-distant migrant employing active flapping flight.

For demonstration, the probability of active flight was read in as a file, but here set to one for all time steps and locations (see Section 3.2 for instructions on how to read probability of active vs. passive flight from file). For larger birds that depend on favourable winds allowing passive flight, this file is easily adaptable. The environment at the two locations of migration (e.g. Northern Europe and Eastern Africa) had the same average food supply but differed in their degree of seasonality. Here, sOARs integrated sine curves for describing food supply are adapted, though food availability can also be read in from file, analoguous to the probability of active flight. We computed the optimal behavioural strategy for three scenarios differing in seasonality of environmental food availability in the wintering habitat such that the maximum potential energy gain from foraging in the two locations differed by approximately 25 %, 30 % and 35 % during peak times (see Figure 4.1).

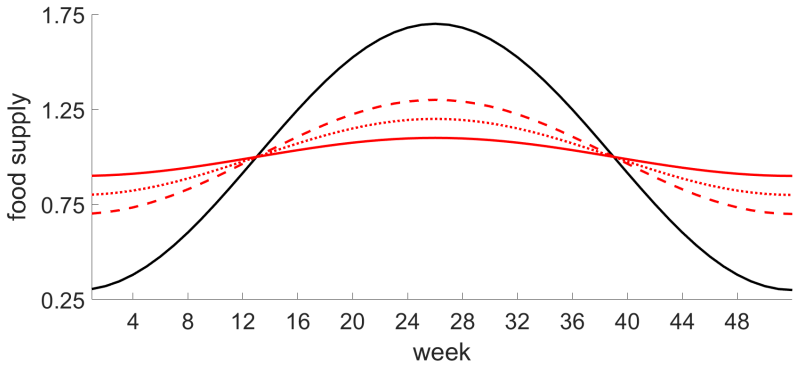

Figure 4.1: Environmental food availability in the breeding (black) and wintering (red) habitat varying sinusoidal over the year. Average yearly food supply in both locations is the same but the degree of seasonality in the wintering habitat varies from low (solid) to high (dashed).

4.1.2 Sample configuration file

In the following, we provide a sample configuration file for the illustrative example. The configuration file is a structured text file created in a text editor and saved with the filename extension .cfg. For demonstration, we here list all potential parameters, even if they are not set in this example. In general, lines starting with a double slash represent comments and are neglected by sOAR. The value of the parameter BackgroundMortalityByDisease corresponds to a maximum age of 12 years for a bird which is in top shape throughout its life.

Sample file (filename.cfg):

```
// ================================================
// GENERAL SETTINGS
// ================================================
// --------------------------------
// Presettings
// --------------------------------
RunBackward = true;
RunForward = true;
EnableMigrationOption = true;
EnableHealthDimension = true;
// --------------------------------
// Backward initialization
// --------------------------------
BackwardMaximumNumberOfIterations = 50;
BackwardFilePrefix = "sOAR_example";
BackwardCalibrateTheta = true; // false;
BackwardCalibrateThetaMin = 0.3;
```

```
// --------------------------------
// Forward initialization
// --------------------------------
ForwardMaximumNumberOfIterations = 50;
ForwardFilePrefix = "sOAR_example";
ForwardStartEpoch = 24;
ForwardStartLocation = 2;
// --------------------------------
// Grid of time and states
// --------------------------------
DecisionEpochsPerPeriod = 52;
ReservesMin = 0.0;
ReservesMax = 10.0;
RevervesSubdivisions = 10;
HealthLevelMin = 0.0;
HealthLevelMax = 10.0;
HealthSubdivisions = 10;
ExperienceMax = 2;
AgeOfIndependence = 6;
NumberOfLocations = 2;
// --------------------------------
// Numerical parameters
// --------------------------------
ConvergenceCriterion = 1e-006;
// AdditionalStochasticityReserves = true;
// AdditionalStochasticityHealth = false;
StochasticityFactor = 0.166666666666667;
// ================================================
// ENVIRONMENT
// ================================================
// --------------------------------
// Food
// --------------------------------
AverageFoodSupply = ( 1.0, 1.0 );
FoodSeasonality = ( 0.1, 0.7 ); // ( 0.2, 0.7 ); // ( 0.3, 0.7 );
// FoodSupplyCSV = "filename.csv";
// --------------------------------
// Wind
// --------------------------------
// ProbabilityOfActiveFlight = 1.0;
ProbabilityOfActiveFlightCSV = "pAct.csv";
// ================================================
```

```
// COSTS AND CONSTRAINTS
// ================================================
// --------------------------------
// General
// --------------------------------
BasalMetabolicRate = 2.0;
// --------------------------------
// Reproduction
// --------------------------------
ReserveCostsOfStartReproduction = 0.5;
ReserveCostsOfIncubation = 1.5;
ReserveCostsOfCareForYoung = 7.0;
HealthCostsOfStartReproduction = 1.0;
DurationOfIncubation = 4;
NumberOfOffspring = 2;
FitnessAtIndependence = 0.5;
// --------------------------------
// Experience
// --------------------------------
ProbabilityOfExperienceGrowth = 0.02;
//Theta = 0.716545; // 0.758081; // 0.812251;
// --------------------------------
// Migration
// --------------------------------
DurationOfMigration = 3;
PredationRiskDuringMigration = 0.01;
ReserveDependencyOfPredationDuringMigration = ("Quadratic", (0,0,0.01));
ReserveCostsOfActiveFlight = 2.0;
ReserveCostsOfPassiveFlight = 1.0;
HealthCostsOfActiveFlight = 1.0;
HealthCostsOfPassiveFlight = 0.5;
// ================================================
// RESPONSE FUNCTIONS
// ================================================
// --------------------------------
// Metabolism
// --------------------------------
ReserveDependencyOfBMR = ("Quadratic", (0,0,0.01));
ActivityDependencyOfMetabolism = ("Quadratic", (0,0,6));
ReserveDependencyOfMetabolism = ("Quadratic", (0,0,0.01));
// --------------------------------
// Predation
```

```
// --------------------------------
ReserveDependencyOfPredationLocation1 = ("Quadratic", (0,0,0.01));
ReserveDependencyOfPredationLocation2 = ("Quadratic", (0,0,0.01));
ActivityDependencyOfPredationLocation1 = ("Quadratic", (0,0,0.01));
ActivityDependencyOfPredationLocation2 = ("Quadratic", (0,0,0.01));
// --------------------------------
// Flight energetics
// --------------------------------
ReserveDependencyOfActiveFlight = ("Quadratic", (0,0,0.001));
ReserveDependencyOfPassiveFlight = ("Quadratic", (0,0,0.001));
// --------------------------------
// Immune response
// --------------------------------
BackgroundMortalityByDisease = 0.0016;
MetabolismDependencyOfHealth = ("Quadratic", (1,-0.2,-0.01));
```

Remarks

Food supply is defined using the internal food sine function with parameters

- `AverageFoodSupply` and
- `FoodSeasonality`.

These determine the average food supply (i.e. sinusoidal axis) and its degree of seasonality (i.e. amplitude of sine curve) over one complete time cycle (see Section 2.3.1.2 for details). For the illustrative example, we ran three scenarios in which food seasonality in the wintering location (first entry of the parameter definition in the sample script above) took the values 0.1, 0.2, 0.3.

Alternatively, environmental food availability can be read in from file using the command

```
FoodSupplyCSV = "filename.csv";
```

The file must have CSV format and specify food supply for each decision epoch and location (see Chapter 3.2).

The probability of active flight p_{act} was read in from CSV file. Following the guidelines in Section 3.2, the file content for our illustrative example of a migratory bird employing always active flight and having a duration of migration of three weeks was:

```
Week;Loc1;Stop1;Stop2;Loc2
1;1.0;1.0;1.0;1.0
2;1.0;1.0;1.0;1.0
```

```
3;1.0;1.0;1.0;1.0
4;1.0;1.0;1.0;1.0
...  and so on ...
```

Here, we only used the file option for demonstrative purposes. If probability of active flight is not considered to be time or location dependent, p_{act} could also be set constant using the command

```
ProbabilityOfActiveFlight = <someValue>;
```

For details on how to configure environmental food supply and probability of active flight see Section 3.2.

4.1.3 Results

In the following, first the major results of the sOAR computations are presented as in the corresponding article. Then, for the sake of completeness, we provide a short overview of the correct output as produced by the accompanying examplary Matlab (or R) file for analysing sOAR results.

Timing of reproduction and migration

Though our illustrative example is simple, results showed that food availability can be critical in determining the optimal timing of migration. In particular, the preferential departure date varied depending on seasonality of food supply in the wintering habitat. An intermediate degree of synchronic seasonality in the southern location further resulted in differential timing of autumn migration between experience classes (Figure 4.2a-c), whereas for lower or higher synchronic seasonality in the wintering habitat the optimal autumn departure mainly depended on the level of reserves (Figure 4.2d-f) and health condition (not shown). Hence, similar temporal food availability between breeding and wintering habitat may cause various migratory patterns in birds irrespective of other potential influencing factors. However, even if mean migration dates vary with age, the overlap in timing of autumn migration at the population level can be considerable (Figure 4.2c). Timing of spring migration was overall less variable because the birds have matured by that time and because the optimal timing of spring migration is more influenced by conditions in the breeding habitat. However, spring departure timing became more variable with smaller differences in seasonal food supply between locations (higher synchronic seasonality) because individuals have more difficulties in building up reserves after strenuous autumn migration. In the example, no major differences in the optimal timing of the breeding period occur but highest food abundance coincides with hatching of the young for all settings as would be expected.
We note that time slots of favourable wind conditions might be similarly important in driving the timing of migration in soaring and gliding birds since passive flight can decrease their energy consumption during migration considerably (Norberg, 1990; Pennycuick, 1972), which can be further explored using sOAR.

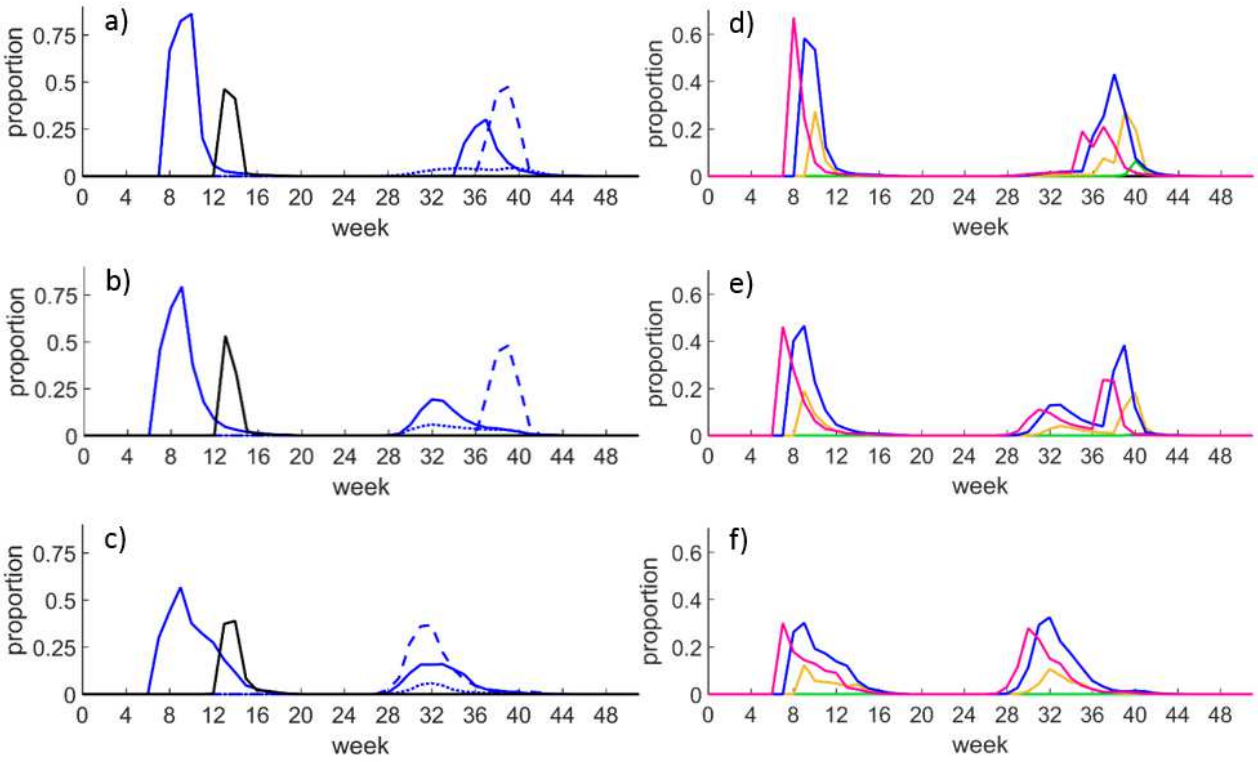

Figure 4.2: The timing of behaviour with respect to different experience (a–c, left) and reserve classes (d–f, right) for low (top), medium (middle) and high (bottom) synchronic seasonality of environmental food supply between breeding and wintering habitat. Graphs represent the proportion of a population following the optimal strategy that performs a certain behaviour or has a certain condition over the year. Left: The timing of migration (blue), especially autumn migration, varies with seasonality in the wintering habitat, in contrast to the breeding period (black). Varying seasonality can induce synchronic or differential migration between individuals with low (dashed), medium (dotted) or high (solid) experience. Right: The level of reserves on the onset of migration is depicted from high to low in pink, blue, ochre and green. Regardless of other factors, an early departure is optimal when reserves are high.

Below now follows the figure output of the accompanying Matlab script for analysing sOAR results for the case that food seasonality in the wintering location is 0.1.

Results: Matlab/R figure output

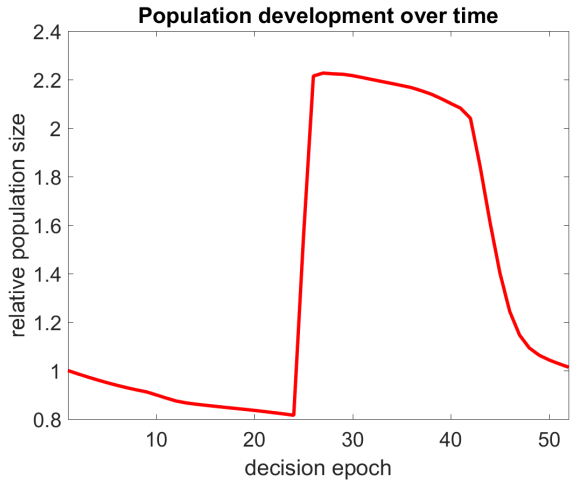

Figure 4.3: Development of population size over time relative to population size at the first stage of the cycle. Default file name *figurePopulationDevelopment.png*.

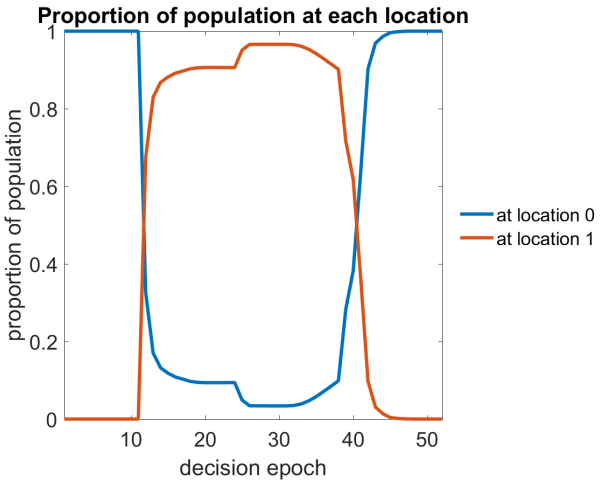

Figure 4.4: Development of population size over time relative to population size at the first stage of the cycle, for each location. Default file name *figurePopulationDevelopmentLocation.png*.

Please note that Figure 4.4 will only be produced if the migration option had been switched on (`EnableMigrationOption == true`) during sOAR computations.

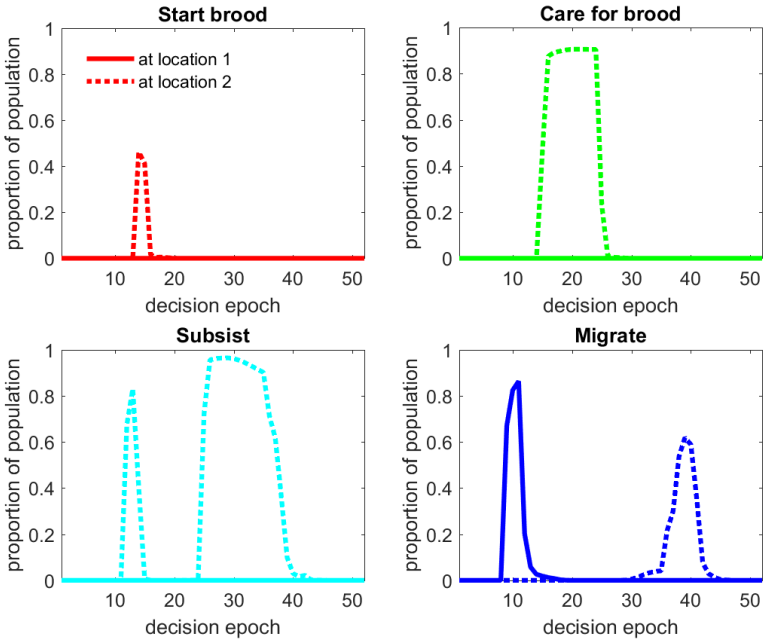

Figure 4.5: Proportion of the population performing a certain behaviour over time (one subplot per behavioural option). Solid lines indicate behaviour performed at location 1 and, if applicable, dotted lines are used for behaviour at location 2. Default file name *figureTimingBehavLocSubplots.png*.

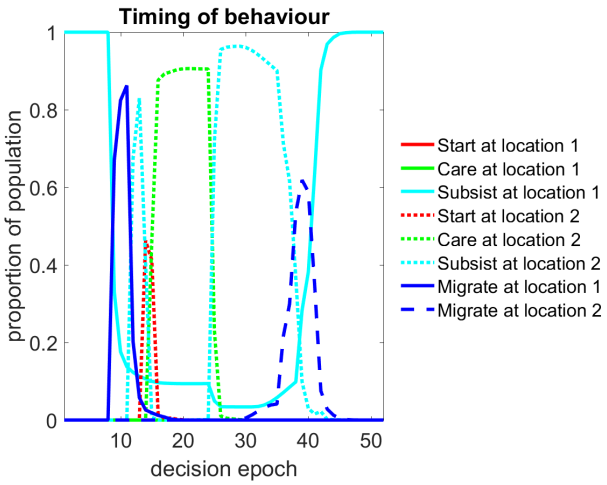

Figure 4.6: Proportion of the population performing a certain behaviour over time (all behavioural options in one plot). Solid lines indicate behaviour performed at location 1 and, if applicable, dotted lines are used for behaviour at location 2. Default file name *figureTimingBehavLoc.png*.

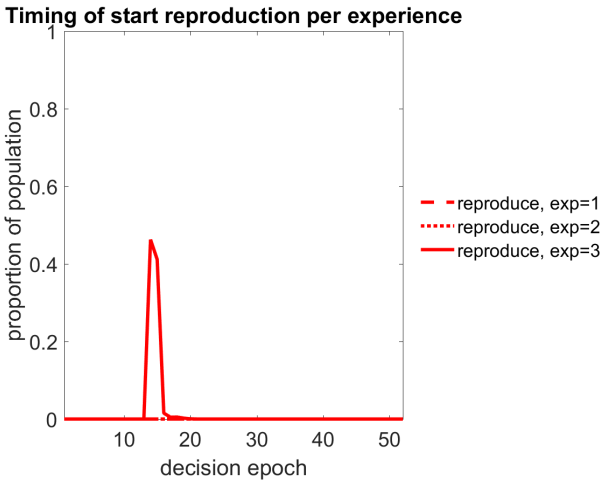

Figure 4.7: Proportion of the population starting reproduction activities over time, differentiated by experience classes. Default file name *figureTimingStartExp.png*.

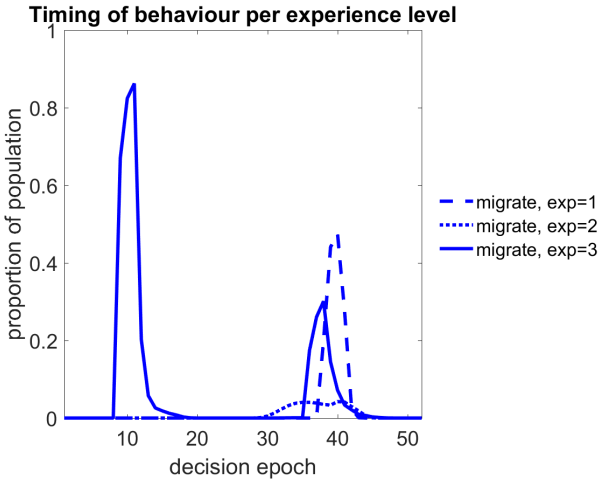

Figure 4.8: Proportion of the population starting migration over time, differentiated by experience classes. Default file name *figureTimingMigrExp.png*.

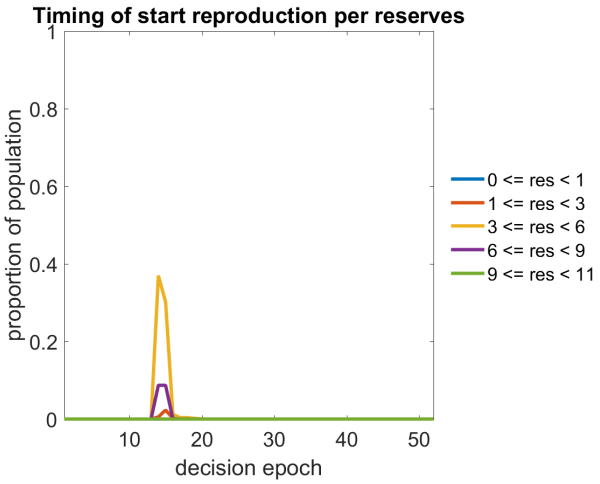

Figure 4.9: Proportion of the population starting reproduction activities over time, differentiated by reserve classes. Default file name *figureTimingStartRes.png*.

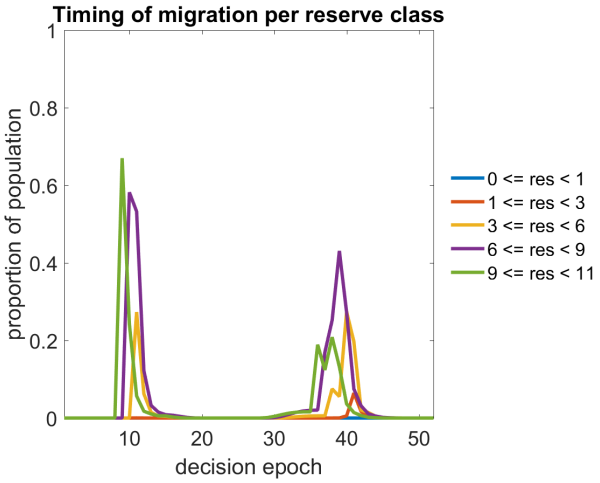

Figure 4.10: Proportion of the population starting migration over time, differentiated by reserve classes. Default file name *figureTimingMigrRes.png*.

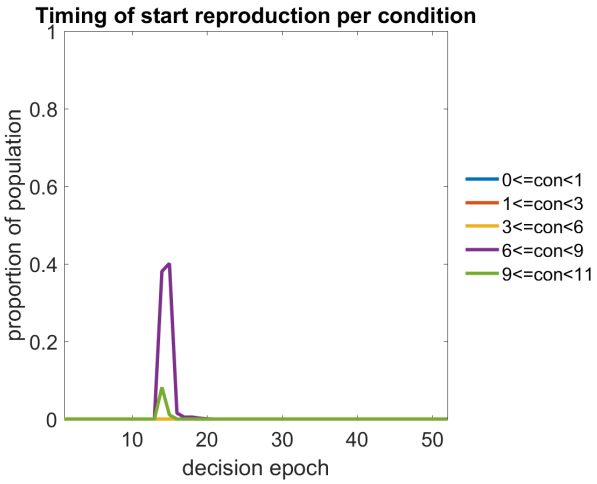

Figure 4.11: Proportion of the population starting reproduction activities over time, differentiated by health condition classes. Default file name *figureTimingStartCon.png*.

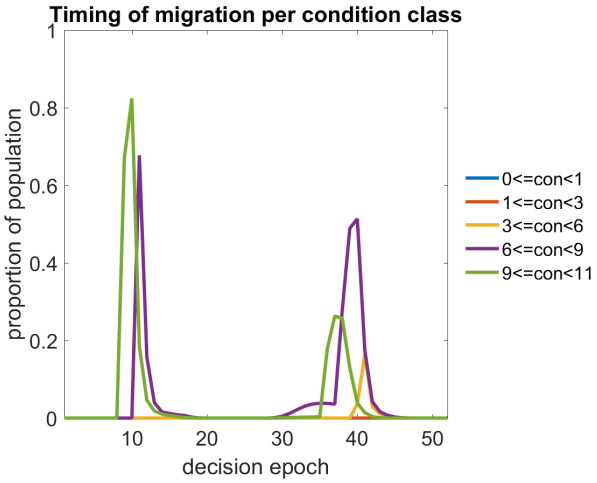

Figure 4.12: Proportion of the population starting migration over time, differentiated by health condition classes. Default file name *figureTimingMigrCon.png*.

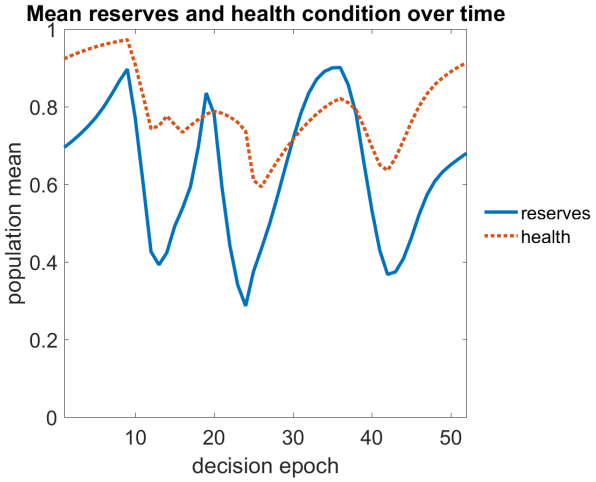

Figure 4.13: Normalized mean level of energy reserves and health condition of the population over time. Default file name *figureMeanReservesHealth.png*.

4.2 Number of brood cycles

4.2.1 Background

In the following, we provide another simple example which can be used to ensure that the software has been set up properly and is working as expected. Here, the age at which the offspring becomes independent from the parent is varied, taking a value of 3, 4 or 5 weeks. Consequently, the optimal number of brood cycles per year changes. In this example, neither the migration option nor the optional health variable were switched on. Accordingly, the parameters associated with migration or health development do not need to be set in the configuration file (else they will be neglected upon a warning message).

4.2.2 Sample configuration file

Sample file (Schaefer_et_al_2017_A5_sOAR_ConfigFile_NrBroodCycles.cfg):

```
// ================================================
// GENERAL SETTINGS
// ================================================
// --------------------------------
// Presettings
// --------------------------------
RunBackward = true;
RunForward = true;
EnableMigrationOption = false;
EnableHealthDimension = false;
```

```
// --------------------------------
// Backward initialization
// --------------------------------
BackwardMaximumNumberOfIterations = 50;
BackwardFilePrefix = "sOAR_example_NrBroodCycles";
BackwardCalibrateTheta = true; // false;
BackwardCalibrateThetaMin = 0.0;
// --------------------------------
// Forward initialization
// --------------------------------
ForwardMaximumNumberOfIterations = 50;
ForwardFilePrefix = "sOAR_example_NrBroodCycles";
ForwardStartEpoch = 20;
ForwardStartLocation = 1;
// --------------------------------
// Grid of time and states
// --------------------------------
DecisionEpochsPerPeriod = 52;
ReservesMin = 0.0;
ReservesMax = 10.0;
RevervesSubdivisions = 10;
ExperienceMax = 2;
AgeOfIndependence = 3; // 4; // 5;
NumberOfLocations = 1;
// --------------------------------
// Numerical parameters
// --------------------------------
ConvergenceCriterion = 1e-006;
StochasticityFactor = 0.1666666666666667;
// ================================================
// ENVIRONMENT
// ================================================
// --------------------------------
// Food
// --------------------------------
AverageFoodSupply = ( 1.1 );
FoodSeasonality = ( 0.2 );
// ================================================
// COSTS AND CONSTRAINTS
// ================================================
// --------------------------------
// Experience
```

```
// --------------------------------
ProbabilityOfExperienceGrowth = 0.04;
//Theta = 0.499795; // 0.513293; // 0.525951;
// --------------------------------
// Reproduction
// --------------------------------
ReserveCostsOfStartReproduction = 0.5;
ReserveCostsOfIncubation = 2.3;
ReserveCostsOfCareForYoung = 4.6;
DurationOfIncubation = 2;
NumberOfOffspring = 1;
FitnessAtIndependence = 0.8;
// ================================================
// RESPONSE FUNCTIONS
// ================================================
// --------------------------------
// Metabolism
// --------------------------------
BasalMetabolicRate = 1.87;
ReserveDependencyOfBMR = ("Constant", (0.0));
ActivityDependencyOfMetabolism = ("Quadratic", (0,2,3));
ReserveDependencyOfMetabolism = ("Constant", (0.0));
// --------------------------------
// Predation
// --------------------------------
ReserveDependencyOfPredationLocation1 = ("Constant", (0.0));
ActivityDependencyOfPredationLocation1 = ("Quadratic", (0.0019,0,0.004));
```

4.2.3 Results

Again, we first present the major results of sOAR computations for this example, as given in the corresponding article. Then, a short overview of the correct output figures follows which would be produced by the accompanying examplary R (or Matlab) file for analysing sOAR results.

As expected, the optimal number of breeding cycles per year decreases with the offspring becoming independent from the parent at an older age (see 4.14). Further differentiating the breeding population by reserve classes (cf. Figure 4.15) again reveals that birds with high levels of energy reserves may start breeding earlier in the year than those with lower reserve levels.

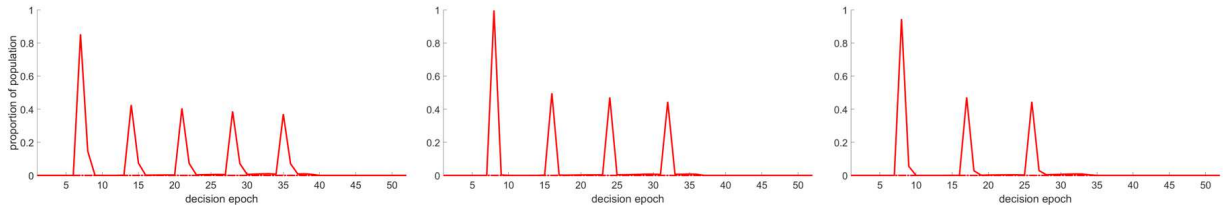

Figure 4.14: Proportion of population initiating breeding over the year for birds with high (solid), medium (dotted) and low (dashed) experience. In the three depicted scenarios, the age at which young become independent was varied between 3 (left), 4 (middle) and 5 (right) weeks.

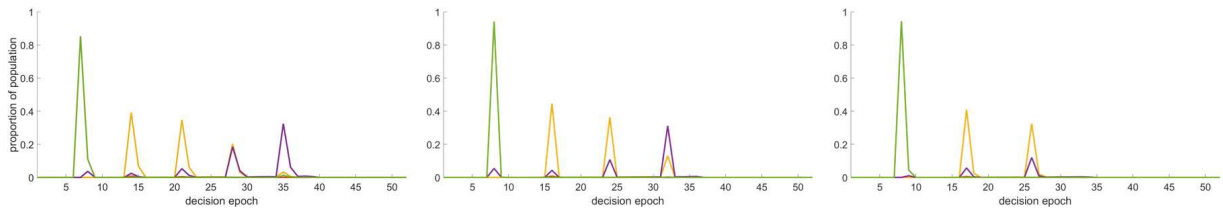

Figure 4.15: Proportion of population initiating breeding over the year for birds with very high (green), higher (violet), lower (yellow) and very low (orange) levels of reserves. In the three depicted scenarios, the age at which young become independent was varied between 3 (left), 4 (middle) and 5 (right) weeks.

The figure output of the accompanying R script for analysing sOAR results follows below and corresponds to the case that offspring gets independent from the parent bird at an age of 4 weeks.

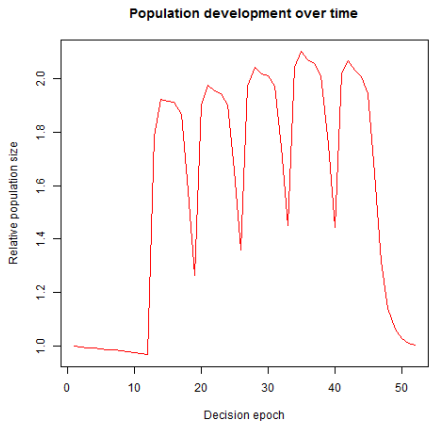

Figure 4.16: Development of population size over time relative to population size at the first stage of the cycle. Default file name *figurePopulationDevelopment.png*.

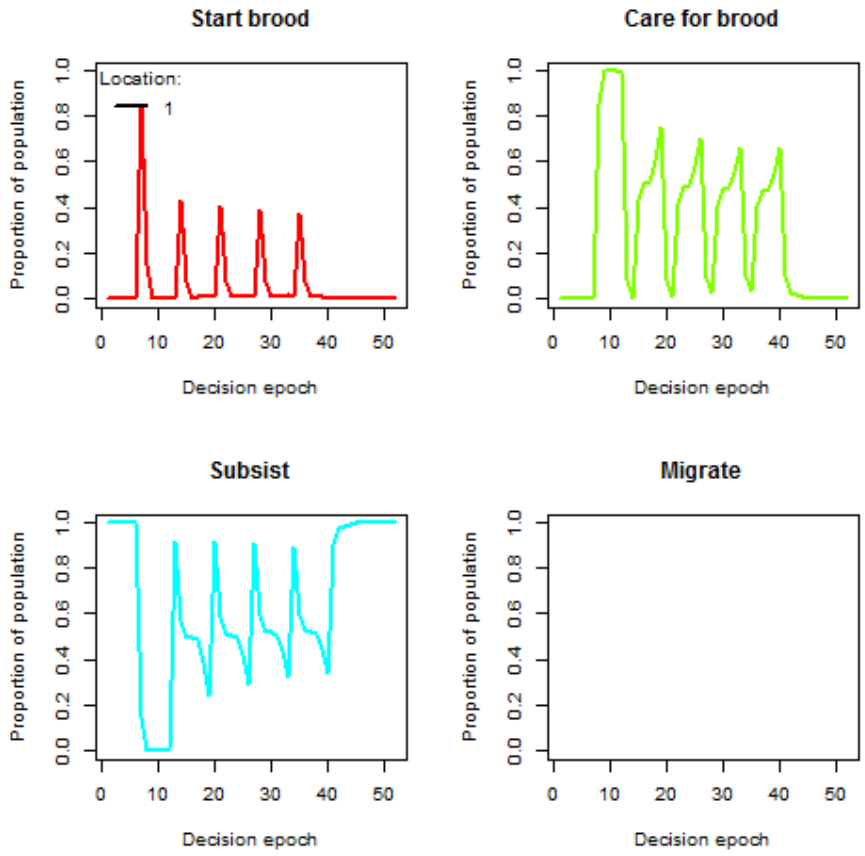

Figure 4.17: Proportion of the population performing a certain behaviour over time (one subplot per behavioural option). Since the migration option was not enabled, the corresponding plot is empty and there is only one location represented in each plot. Default file name *figureTimingBehavLocSubplots.png*.

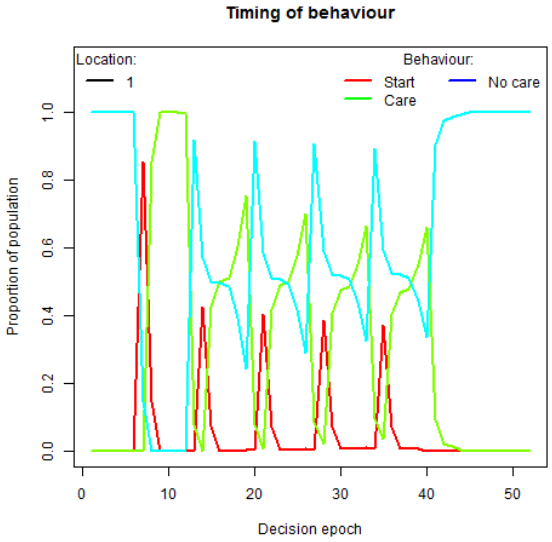

Figure 4.18: Proportion of the population performing a certain behaviour over time (all behavioural options in one plot). Different behavioural options are visualized by different colours. Since the migration option was not enabled, there is only one location represented by a solid line. Default file name *figureTimingBehavLoc.png*.

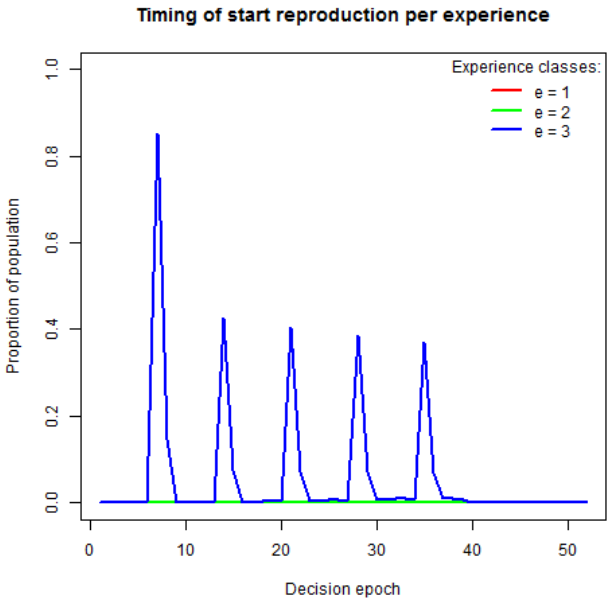

Figure 4.19: Proportion of the population starting reproduction activities over time, differentiated by experience classes. Default file name *figureTimingStartExp.png*.

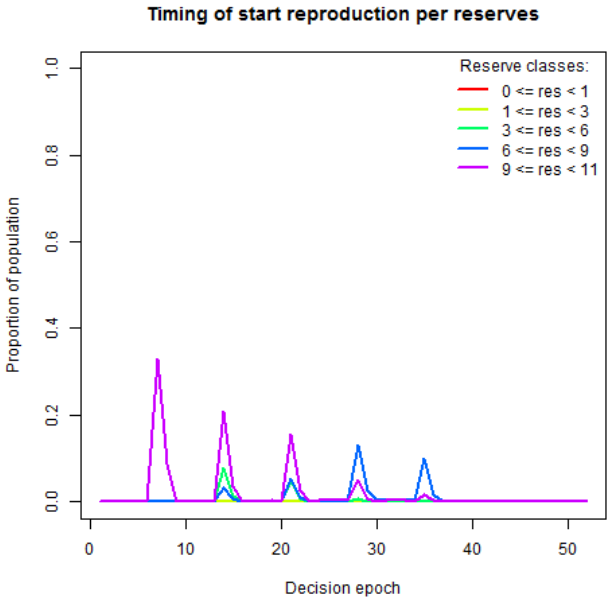

Figure 4.20: Proportion of the population starting reproduction activities over time, differentiated by reserve classes. Default file name *figureTimingStartRes.png*.

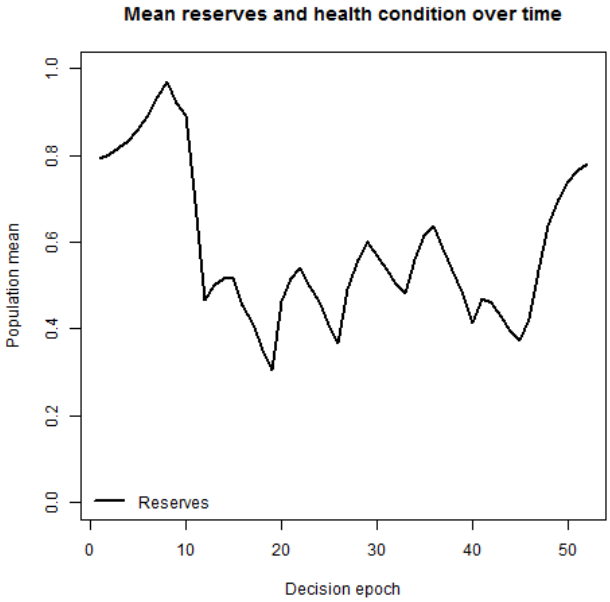

Figure 4.21: Normalized mean level of energy reserves of the population over time. Since the health variable was not enabled, it is not represented by any graph. Default file name *figureMeanReservesHealth.png*.

4.3 Published examples

4.3.1 Reproduction model by Houston and McNamara 1999

sOAR was tested on the example based on the timing of reproduction presented in Chapter 9.5 in Houston and McNamara (1999). In the following, we include some sample figures of corresponding results for proof of concept and comparison with the original sources. The parameter settings correspond to those in Houston and McNamara (1999). To interpolate between grid points and to add additional stochasticity, both energy reserves and health condition took one of four potential values in our computation (cf. Chapter 3.1 in [HM99]). Hereby, the corresponding stochasticity factor $0 < \alpha < 1/3$ was set to $alpha = 0.166666666666667$, resulting in the optimized experience-related factor of $\theta = 0.819$. We attribute remaining differences between our results and those of Houston and McNamara [HM99] to likely differences in the actual implementation of the computational routine, in the selected stochasticity settings and convergence criterion as well as numerical effects.

4.3.1.1 Sample configuration file

Sample file (Schaefer_et_al_2017_A5_sOAR_ConfigFile_HoustonMcNamara1999Variant.cfg):

```
// ================================================
// GENERAL SETTINGS
// ================================================
// --------------------------------
// Presettings
// --------------------------------
RunBackward = true;
RunForward = true;
EnableMigrationOption = false;
EnableHealthDimension = true;
// --------------------------------
// Backward initialization
// --------------------------------
BackwardMaximumNumberOfIterations = 50;
BackwardFilePrefix = "sOAR_example_HoustonMcNamara1999Variant";
BackwardCalibrateTheta = true;
BackwardCalibrateThetaMin = 0.0;
// --------------------------------
// Forward initialization
// --------------------------------
ForwardMaximumNumberOfIterations = 50;
ForwardFilePrefix = "sOAR_example_HoustonMcNamara1999Variant";
ForwardStartEpoch = 24;
ForwardStartLocation = 1;
```

```
// --------------------------------
// Grid of time and states
// --------------------------------
DecisionEpochsPerPeriod = 52;
ReservesMin = 0.0;
ReservesMax = 1.0;
RevervesSubdivisions = 16;
HealthLevelMin = 0.0;
HealthLevelMax = 1.0;
HealthSubdivisions = 16;
ExperienceMax = 2;
AgeOfIndependence = 10;
NumberOfLocations = 1;
// --------------------------------
// Numerical parameters
// --------------------------------
ConvergenceCriterion = 1e-006;
AdditionalStochasticityReserves = true;
AdditionalStochasticityHealth = true;
StochasticityFactor = 0.1666666666666667;
// ================================================
// ENVIRONMENT
// ================================================
// --------------------------------
// Food
// --------------------------------
AverageFoodSupply = ( 1.0 );
FoodSeasonality = ( 0.3 );
// ================================================
// COSTS AND CONSTRAINTS
// ================================================
// --------------------------------
// Experience
// --------------------------------
ProbabilityOfExperienceGrowth = 0.1;
// --------------------------------
// Reproduction
// --------------------------------
ReserveCostsOfStartReproduction = 0.2;
ReserveCostsOfIncubation = 0.0;
ReserveCostsOfCareForYoung = 0.4;
HealthCostsOfStartReproduction = 0.1;
```

```
DurationOfIncubation = 0;
NumberOfOffspring = 1;
FitnessAtIndependence = 0.5;
// ================================================
// RESPONSE FUNCTIONS
// ================================================
// --------------------------------
// Metabolism
// --------------------------------
BasalMetabolicRate = 0.3;
ReserveDependencyOfBMR = ("Constant", (0.0));
ActivityDependencyOfMetabolism = ("Quadratic", (0,0,0.4));
ReserveDependencyOfMetabolism = ("Constant", (0.0));
// --------------------------------
// Predation
// --------------------------------
ReserveDependencyOfPredationLocation1 = ("Constant", (0.0));
ActivityDependencyOfPredationLocation1 = ("Quadratic", (0,0,0.004));
// --------------------------------
// Immune response
// --------------------------------
BackgroundMortalityByDisease = 0.004;
MetabolismDependencyOfHealth = ("Quadratic", (0.0625,0.0,-0.390625));
```

4.3.1.2 Results

The style of sOAR results presented here follows Houston and McNamara (1999) to ease comparison. Note that the provided Matlab and R sample code for results analysis feature a different style.

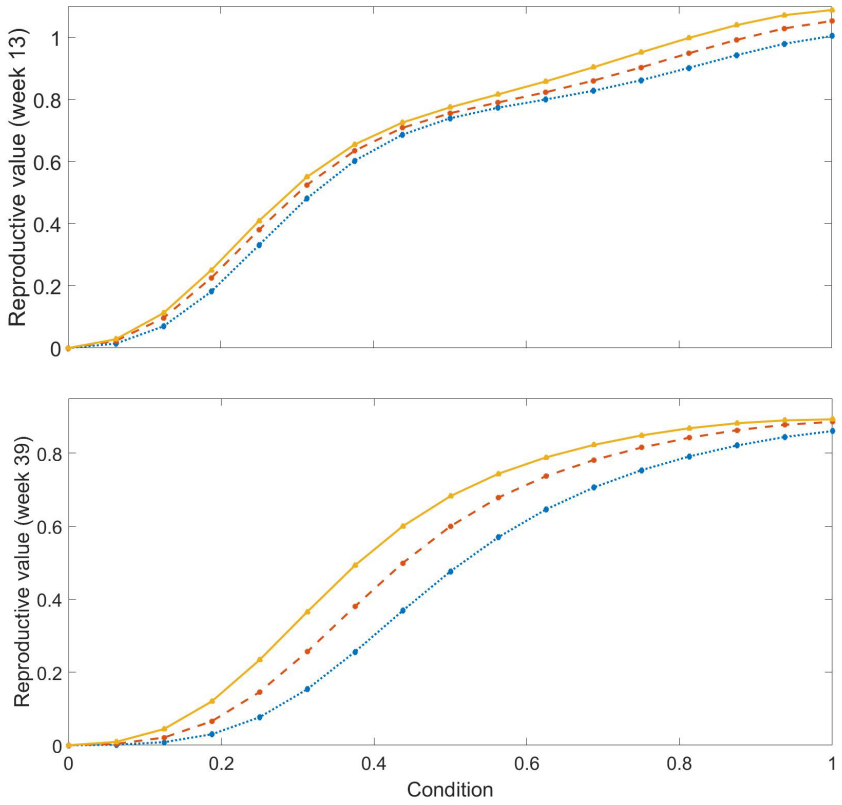

Figure 4.22: Reproductive value as a function of health condition for three levels of energy reserves (a) at the end of winter (week 13) and (b) at the beginning of autumn (week 39). In each case the animal has maximum experience ($e = 2$) and is not caring for offspring ($a = -1$).

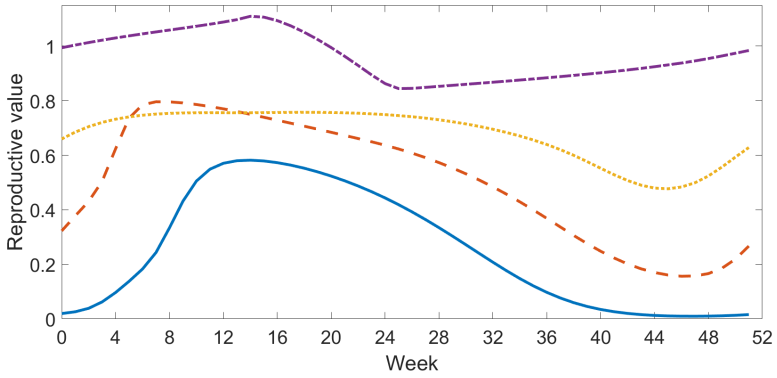

Figure 4.23: Reproductive value of newly independent young as a function of the time of year at which they become independent. Newly independent young (solid blue) have energy reserves $x = 0.5$, health condition $y = 0.5$ and experience $e = 0$. For comparison, reproductive value as a function of time of year is also shown for the following combinations of energy reserves x, health condition y and experience e: $x = 1$, $y = 1$, $e = 0$ (dashed orange), $x = 0.5$, $y = 0.5$, $e = 2$ (dotted yellow), $x = 1$, $y = 1$, $e = 2$ (dash-dot violet). In all cases $a = -1$.

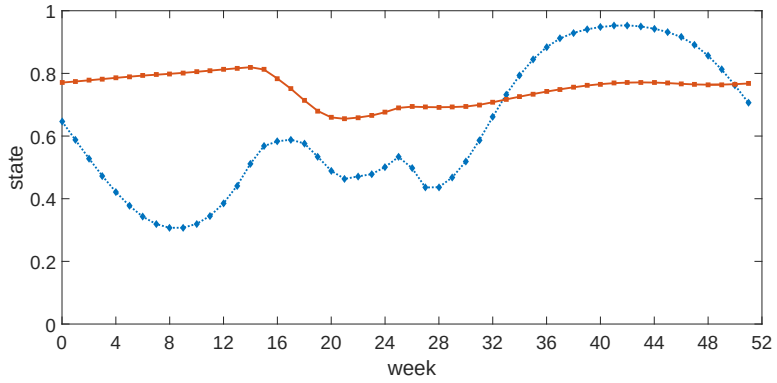

Figure 4.24: Mean reserves (dotted blue) and condition (solid red) as a function of time of year. We give the equilibrium pattern for animals with maximum experience ($e = 2$) which are not caring for offspring ($a = -1$).

4.3.2 Migration model by McNamara et al. 1998

The sOAR software has also been tested on a migration model oriented at McNamara et al. (1998). Though the exact parametrization and equations of that model are not replicable anymore, the resulting optimal timing of migration and reproduction (cf. Figure 4.25) is in good agreement with the original.

4.3.2.1 Sample configuration file

Sample file (Schaefer_et_al_2017_A5_sOAR_ConfigFile_McNamaraEtAl1998Variant):

```
// ================================================
// GENERAL SETTINGS
// ================================================
// --------------------------------
// Presettings
// --------------------------------
RunBackward = true;
RunForward = true;
EnableMigrationOption = true;
EnableHealthDimension = true;
// --------------------------------
// Backward initialization
// --------------------------------
BackwardMaximumNumberOfIterations = 50;
BackwardFilePrefix = "sOAR_example_McNEtAl_1998";
BackwardCalibrateTheta = true;
BackwardCalibrateThetaMin = 0.0;
// --------------------------------
// Forward initialization
// --------------------------------
ForwardMaximumNumberOfIterations = 50;
ForwardFilePrefix = "sOAR_example_McNEtAl_1998";
ForwardStartEpoch = 24;
ForwardStartLocation = 2;
// --------------------------------
// Grid of time and states
// --------------------------------
DecisionEpochsPerPeriod = 52;
ReservesMin = 0.0;
ReservesMax = 10.0;
RevervesSubdivisions = 10;
HealthLevelMin = 0.0;
HealthLevelMax = 10.0;
```

```
HealthSubdivisions = 10;
ExperienceMax = 2;
AgeOfIndependence = 4;
NumberOfLocations = 2;
// --------------------------------
// Numerical parameters
// --------------------------------
ConvergenceCriterion = 1e-006;
AdditionalStochasticityReserves = true;
AdditionalStochasticityHealth = true;
StochasticityFactor = 0.1666666666666667;
// ================================================
// ENVIRONMENT
// ================================================
// --------------------------------
// Food
// --------------------------------
AverageFoodSupply = ( 1.0, 1.0 );
FoodSeasonality = ( 0.0, 0.4 );
// --------------------------------
// Wind
// --------------------------------
ProbabilityOfActiveFlight = 1.0;
// ================================================
// COSTS AND CONSTRAINTS
// ================================================
// --------------------------------
// Experience
// --------------------------------
ProbabilityOfExperienceGrowth = 0.1;
// --------------------------------
// Reproduction
// --------------------------------
ReserveCostsOfStartReproduction = 2.0;
ReserveCostsOfIncubation = 0.0;
ReserveCostsOfCareForYoung = 6.0;
HealthCostsOfStartReproduction = 1.0;
DurationOfIncubation = 0;
NumberOfOffspring = 1;
FitnessAtIndependence = 0.5;
// --------------------------------
// Migration
```

```
// --------------------------------
DurationOfMigration = 1;
PredationRiskDuringMigration = 0.01;
ReserveDependencyOfPredationDuringMigration = ("Quadratic", (0,0,0.1));
ReserveCostsOfActiveFlight = 5.0;
ReserveCostsOfPassiveFlight = 5.0;
HealthCostsOfActiveFlight = 1.6;
HealthCostsOfPassiveFlight = 1.6;
// ================================================
// RESPONSE FUNCTIONS
// ================================================
// --------------------------------
// Metabolism
// --------------------------------
BasalMetabolicRate = 2.0;
ReserveDependencyOfBMR = ("Quadratic", (0,0,0.1));
ActivityDependencyOfMetabolism = ("Quadratic", (0,2,4));
ReserveDependencyOfMetabolism = ("Quadratic", (0,0,0.1));
// --------------------------------
// Predation
// --------------------------------
ReserveDependencyOfPredationLocation1 = ("Quadratic", (0,0,0.1));
ReserveDependencyOfPredationLocation2 = ("Quadratic", (0,0,0.1));
ActivityDependencyOfPredationLocation1 = ("Quadratic", (0,0,0.01));
ActivityDependencyOfPredationLocation2 = ("Quadratic", (0,0,0.01));
// --------------------------------
// Flight energetics
// --------------------------------
ReserveDependencyOfActiveFlight = ("Quadratic", (0,0,0.001));
ReserveDependencyOfPassiveFlight = ("Quadratic", (0,0,0.001));
// --------------------------------
// Immune response
// --------------------------------
BackgroundMortalityByDisease = 0.004;
MetabolismDependencyOfHealth = ("Quadratic", (0.8, -0.1, -0.025));
```

4.3.2.2 Results

Timing of migration and reproduction

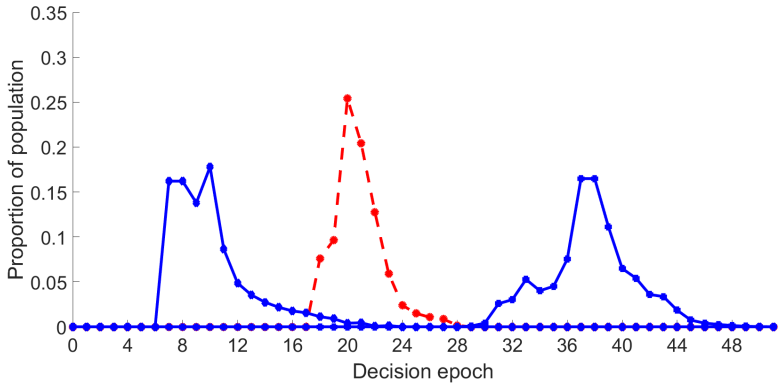

Figure 4.25: The timing of migration (solid blue) and the initiation of breeding (dashed red) by the modelled birds, oriented at the baseline scenario of [MWH98].

5 Conclusion

sOAR provides a powerful and user-friendly implementation of the optimal annual routine framework by Houston and McNamara (1999) for computing optimal life history strategies of animals under periodic environmental conditions and simulating their population dynamics given such a strategy. Complementing theoretical studies of animal behaviour, it will improve our understanding of how natural selection shapes trade-offs in animal behaviour within cyclic environments. We further extended the original framework to differentiate between costs of active and passive flight and to consider periodic wind conditions acting on birds during migration. For example, this extension allows studying the timing of migration between obligate soaring vs. flapping birds, or studying ontogenetic differences in birds where soaring vs. flapping flight is age dependent.

Our illustrative example showed how the optimal timing of spring and autumn migration may vary with environmental food supply at a site. Such insights are particularly interesting in the face of global change that may differently alter food availability in different regions. Changing environments may render current behavioural strategies of migrants and other animals suboptimal with potentially negative effects on population dynamics such that, in the long-term, a new optimal strategy should be adapted in order for a population to persist. Such facets, the consequences of suboptimal behaviour or the development of new behavioural strategies, can be easily explored using sOAR. Also, individual variation can be explored within OAR frameworks and be compared to empirically observed movement data, for example by means of telemetry [FSB+08]. Overall, such analyses will contribute to better understanding the different constraints on movement and behaviour of individuals and the consequences for population and community dynamics [NGR+08, JBP+13].

The following list provides a summary of suggested usages of the current program (cf. [FSB+08]) whereby additional sites and processes such as thermoregulation, molt or explicit density-dependent effects [BMH+08] could be integrated into future versions of sOAR:

- Analysis of life-history constraints under global change, e.g. studying potential effects of land use or climate change on the population dynamics of animals that are not flexible enough to readily adapt their behavioural strategy.
- Prediction of potential new adaptive behavioural strategies, e.g. for migratory birds adjusting to climate change.
- Theoretical studies of carry-over effects at the population level, e.g. analysing how the level of reserves and health condition at the end of the breeding or overwintering season determine reproductive success in subsequent seasons.

- Analysis of phenotypic variation in a population within a life-history context, e.g. determining conditions that lead to optimal behavioural strategies resulting in high or low variability of states and of timing of behaviour between individuals.
- Studies of functional groups of organisms, e.g. to analyse how the optimal timing of behaviour varies for classes of birds employing different degrees of active versus passive flight.
- Combining life-history models with large-scale datasets that are becoming increasingly available nowadays, e.g. incorporating available environmental data into the model, validating it with tracking data or suggesting important factors for which refined data should be collected.

Bibliography

[Ale90] Thomas Alerstam. *Bird migration*. Cambridge University Press, Cambridge and others, 1 edition, 1990.

[Ale91] Thomas Alerstam. Bird flight and optimal migration. *Trends in Ecology & Evolution*, 6(7):210–215, 1991.

[AP70] J. Aschoff and H. Pohl. Der Ruheumsatz von Vögeln als Funktion der Tageszeit und der Körpergröße. *Journal für Ornithologie*, 111(1):38–47, 1970.

[Ber05] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, Massachusetts, USA, 3 edition, 2005.

[Ber12] Peter Berthold. *Vogelzug: Eine aktuelle Gesamtübersicht*. Wissenschaftliche Buchgesellschaft, Darmstadt, 7 edition, 2012.

[BHM+06] Zoltán Barta, Alasdair I Houston, John M McNamara, Robert K Welham, Anders Hedenström, Thomas P Weber, and Orsolya Feró. Annual routines of non-migratory birds: optimal moult strategies. *Oikos*, 112(3):580–593, mar 2006.

[BK13] Silke Bauer and Marcel Klaassen. Mechanistic models of animal migration behaviour–their diversity, structure and use. *Journal of Animal Ecology*, 82(3):498–508, 2013.

[BMH+08] Zoltán Barta, John M McNamara, Alasdair I Houston, Thomas P Weber, Anders Hedenström, and Orsolya Feró. Optimal moult strategies in migratory birds. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 363(1490):211–229, jan 2008.

[Bre73] Richard P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, New Jersey, 1973.

[CM00] Colin W. Clark and Marc Mangel. *Dynamic State Variable Models in Ecology: Methods and Applications*. Oxford University Press, New York, 1 edition, 2000.

[DADB97] Charlotte Deerenberg, Victor Arpanius, Serge Daan, and Nicolaas Bos. Reproductive effort decreases antibody responsiveness. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 264(1384):1021 LP – 1029, jul 1997.

[DMSV89] Serge Daan, Dirkjan Masman, Arjen Strijkstra, and Simon Verhulst. Intraspecific Allometry of Basal Metabolic Rate: Relations with Body Size, Temperature, Composition, and Circadian Phase in the Kestrel, Falco tinnunculus. *Journal of Biological Rhythms*, 4(2):155–171, jun 1989.

[FSB+08] Orsolya Feró, Philip A Stephens, Zoltán Barta, John M McNamara, and Alasdair I Houston. Optimal annual routines: New tools for conservation biology. *Ecological Applications*, 18(6):1563–1577, sep 2008.

[GBB+06] Volker Grimm, Uta Berger, Finn Bastiansen, Sigrunn Eliassen, Vincent Ginot, Jarl Giske, John Goss-Custard, Tamara Grand, Simone K. Heinz, Geir Huse, Andreas Huth, Jane U. Jepsen, Christian Jørgensen, Wolf M. Mooij, Birgit Müller, Guy Pe'er, Cyril Piou, Steven F. Railsback, Andrew M. Robbins, Martha M. Robbins, Eva Rossmanith, Nadja Rüger, Espen Strand, Sami Souissi, Richard A. Stillman, Rune Vabø, Ute Visser, and Donald L. DeAngelis. A standard protocol for describing individual-based and agent-based models. *Ecological Modelling*, 198(1):115–126, 2006.

[GBD+10] Volker Grimm, Uta Berger, Donald L. DeAngelis, J. Gary Polhill, Jarl Giske, and Steven F. Railsback. The ODD protocol: A review and first update. *Ecological Modelling*, 221(23):2760–2768, 2010.

[GGP95] Andrew G Gosler, Jeremy J D Greenwood, and Christopher Perrins. Predation risk and the cost of being fat. *Nature*, 377(6550):621–623, oct 1995.

[Gla05] Douglas S Glazier. Beyond the '3/4-power law': variation in the intra-and interspecific scaling of metabolic rate in animals. *Biological Reviews*, 80(4):611–662, nov 2005.

[HBSS+13] Barbara Helm, Rachel Ben-Shlomo, Michael J Sheriff, Roelof A Hut, Russell Foster, Brian M Barnes, and Davide Dominoni. Annual rhythms that underlie phenology: biological time-keeping meets environmental change. *Proceedings of the Royal Society B: Biological Sciences*, 280(1765):1–10, jul 2013.

[HG12] A. Hastings and Louis J. Gross. *Encyclopedia of theoretical ecology*. University of California Press, Berkeley, 2012.

[HH95] Noél Holmgren and Anders Hedenström. The scheduling of molt in migratory birds. *Evolutionary Ecology*, 9(4):354–368, 1995.

[HM99] Alasdair I. Houston and John M. McNamara. *Models of Adaptive Behaviour: An Approach Based on State*. Cambridge University Press, Cambridge, 1 edition, 1999.

[JBP+13] Florian Jeltsch, Dries Bonte, Guy Pe'er, Björn Reineking, Peter Leimgruber, Niko Balkenhol, Boris Schröder, Carsten M Buchmann, Thomas Mueller, Niels Blaum, Damaris Zurell, Katrin Böhning-Gaese, Thorsten Wiegand, Jana A Eccard, Heribert Hofer, Jette Reeg, Ute Eggers, and Silke Bauer. Integrating movement ecology with

biodiversity research - exploring new avenues to address spatiotemporal biodiversity dynamics. *Movement Ecology*, 1(1):6, 2013.

[Kou06] *Chronobiometry: Analyzing for Rhythms*, pages 577–602. Springer Netherlands, Dordrecht, 2006.

[LD00] Robert L Lochmiller and Charlotte Deerenberg. Trade-offs in evolutionary immunology: just what is the cost of immunity? *Oikos*, 88(1):87–98, jan 2000.

[Lim86] Steven L Lima. Predation Risk and Unpredictable Feeding Conditions: Determinants of Body Mass in Birds. *Ecology*, 67(2):377–385, apr 1986.

[MBD+13] Birgit Müller, Friedrich Bohn, Gunnar Dreßler, Jürgen Groeneveld, Christian Klassert, Romina Martin, Maja Schlüter, Jule Schulze, Hanna Weise, and Nina Schwarz. Describing human decisions in agent-based models - ODD + D, an extension of the ODD protocol. *Environmental Modelling & Software*, 48:37–48, 2013.

[MHC01] J McNamara, A Houston, and E Collins. Optimality Models in Behavioral Biology. *SIAM Review*, 43(3):413–466, jan 2001.

[MWH98] John M McNamara, Robert K Welham, and Alasdair I Houston. The Timing of Migration within the Context of an Annual Routine. *Journal of Avian Biology*, 29(4):416–423, 1998.

[NE00] Ken Norris and Matthew R Evans. Ecological immunology: life history trade-offs and immune defense in birds. *Behavioral Ecology*, 11(1):19–26, jan 2000.

[NGR+08] Ran Nathan, Wayne M Getz, Eloy Revilla, Marcel Holyoak, Ronen Kadmon, David Saltz, and Peter E Smouse. A movement ecology paradigm for unifying organismal movement research. *Proceedings of the National Academy of Sciences*, 105(49):19052–19059, dec 2008.

[NH14] Hideharu Numata and Barbara Helm, editors. *Annual, Lunar, and Tidal Clocks: Patterns and Mechanisms of Nature's Enigmatic Rhythms*. Springer Japan, Tokyo, 2014.

[SV96] Ben C Sheldon and Simon Verhulst. Ecological immunology: costly parasite defences and trade-offs in evolutionary ecology. *Trends in Ecology & Evolution*, 11(8):317–321, 1996.

[TMHB12] Jácint Tökölyi, John M McNamara, Alasdair I Houston, and Zoltán Barta. Timing of avian reproduction in unpredictable environments. *Evolutionary Ecology*, 26(1):25–42, 2012.

[VCvO+10] Marcel E Visser, Samuel P Caro, Kees van Oers, Sonja V Schaper, and Barbara Helm. Phenology, seasonal timing and circannual rhythms: towards a unified framework. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 365(1555):3113–3127, sep 2010.

[VJTF07] Øystein Varpe, Christian Jørgensen, Geraint A Tarling, and Øyvind Fiksen. Early Is Better: Seasonal Egg Fitness and Timing of Reproduction in a Zooplankton Life-History Model. *Oikos*, 116(8):1331–1342, 2007.

[WW08] David S Wilcove and Martin Wikelski. Going, Going, Gone: Is Animal Migration Disappearing. *PLoS Biol*, 6(7):e188, jul 2008.

GNU Free Documentation License

Version 1.3, 3 November 2008

<http://fsf.org/>

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.
This License is a kind of copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.
We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as **you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.
A **Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.
A **Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's

overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not Transparent" is called **Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The **publisher**" means any person or entity that distributes copies of the Document to the public.

A section **Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **Acknowledgements**", **Dedications**", **Endorsements**", or **History**".) To **Preserve the Title**" of such a section when you modify the Document means that it remains a section Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty

Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.
You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.
If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.
If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.
It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled Acknowledgements" or Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.
You may add a section Entitled Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.
You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.
The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.
The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique

number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.
In the combination, you must combine any sections Entitled History" in the various original documents, forming one section Entitled History"; likewise combine any sections Entitled Acknowledgements", and any sections Entitled Dedications". You must delete all sections Entitled Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.
You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.
If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.
If a section in the Document is Entitled Acknowledgements", Dedications", or History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.
However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.
Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.
Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.
Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

Massive Multiauthor Collaboration Site" (or MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A Massive Multiauthor Collaboration" (or MMC") contained in the site means any set of copyrightable works thus published on the MMC site.
CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.
An MMC is eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.
The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the with ... Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.
If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.