Grimm, V. (2019):
Individual-based models (updated: 2018)
In: Fath, B.D. (ed.)
Encyclopedia of Ecology, 2nd Edition *Vol. 2*
Elsevier, Oxford, p. 65 - 73

Ecological Models: Individual-based Models[1]

Volker Grimm

Helmholtz Centre for Environmental Research - UFZ

Synopsis

Individual-based models (IBMs) are based on the explicit representation of individual organisms. They are developed for questions where individual variability, local interactions, and adaptive behavior are essential to get the right answers. IBMs are more complex than other mathematical models. Both their complexity and their uncertainty in model structure and parameters are a challenge, but new approaches exist that help meeting these challenges. These approaches are based on multiple patterns, observed in the real systems at different scales and hierarchical levels. Patterns indirectly provide information about the systems' internal organization. The task of the modeler is to decode this information. Specific software libraries and modeling platforms exist to implement IBMs as computer programs. Analyzing IBMs requires considering patterns, to define currencies for comparisons of alternative models and parameterizations, and to perform controlled simulation experiments. Robustness analyses then help identifying insights that can be generalized beyond the specific system that has been modeled. Three IBMs from the literature are briefly presented as examples of pragmatic and paradigmatic models of animal and plant populations and communities. IBMs are an integral and essential part of ecological modeling and further important general insights of Individual-based Ecology are to be expected, in particular since standards for model design, implementation, analysis, and communication are
increasingly used.

Keywords

Adaptive behavior, Agent-based modeling, Emergence, First principles, Individual-based modeling, Local interactions, ODD protocol, Pattern-oriented modeling, Predictions, Robustness analysis, Trait variation

Introduction

Individual-based models (IBMs) describe individual organisms as autonomous, unique entities. Some IBMs deal with quite simple individuals, which are characterized by just their position. Other IBMs include the individual's adaptive decisions of what to do next, which are based on features of the individuals , e.g. age, sex, size, social rank, energy reserves, memory, etc., and of their biotic and abiotic environment. But why should ecological models be based on the representation of individuals in the first place? Models always have to simplify, so why don't we ignore individuals, their variability, and their behavior, and rather consider average individuals as in classical theoretical population ecology?

---

[1] Grimm V (2008) Individual-based models. In: Jørgensen SE, Fath BD (eds), Ecological Models, Vol. [3] of Encyclopedia of Ecology, 5 vols. Elsevier, Oxford, pp. 1959-1968. (updated: 2018)

There are three main reasons why it can be necessary to represent individuals in ecological models: (i) Individual variability. Individuals usually are different, even if they are of the same age. If resources like food or space are scarce, individuals that larger, stronger, or have more experience than others may have a competitive advantage. Moreover, during their life cycle individuals not only change in size but often also in food requirements, behavior, trophic interaction etc. (ii) Local interactions. Most mathematical population models assume global interactions, i.e. all individuals interact with all other individuals, but real interactions are local, which can be important. For example, the global density of individuals may be low enough to provide, on average, enough food or space for each individual, but local density may in some places be much higher than global density. (iii) Adaptive behavior. Individuals seek to maximize fitness, i.e. survive and produce as many offspring as possible that reproduce themselves. To achieve this aim, they adapt their behavior to the current state of themselves and their biotic and abiotic environment (Fig. 1). Behavior not only includes movement, foraging, mating, etc., but also physiology, growth, development, and life history. Adaptive behavior is very likely to affect or even determine system-level properties of populations, communities, and ecosystems. For example, herbivores like elks change their foraging behavior in the presence of predators, e.g. wolves, by selecting other types of habitat. This in turn affects the structure and dynamics of the vegetation and of entire herbivore community, etc.

These three aspects of individual-based ecology are hard to deal with mathematically, e.g. using calculus. Therefore IBMs have to be formulated as simulation models that are run on computers. Computers with enough power for running IBMs are generally available since the end of the 1980ies, which is the time when individual-based modeling became a declared branch of ecological modeling. Nowadays the IBM approach is widely used in ecology but the term individual-based is used in a very broad sense. Some IBMs include only one individual-based aspect, for example the discreteness of individuals, or local interactions, but otherwise are very similar to classical mathematical models, whereas other IBMs include detailed descriptions of the individuals' fitness-seeking behavior.

In the following we first describe the elements of IBMs, which have to be specified by the modeler. Then we explain a general modeling strategy that can be used to optimize the complexity of IBMs and to deal with uncertainty in model structure and parameters (pattern-oriented modeling). Then we describe how IBMs are analyzed, and finally we briefly present three example models and give an outlook on the future of individual-based modeling. The emphasis of the following is thus more on what IBMs are and how they are developed and analyzed then on what so far has been achieved with the individual-based approach.
Note that IBMs are referred to as "agent-based models" (ABMs) in disciplines that deal with human agents, e.g. economics, social sciences, and demography. In these disciplines emphasis of ABMs always was on the agent's adaptive decisions, whereas most IBMs in ecology so far focus on individual variability and local interactions. But behavioral decisions are increasingly considered also in ecology, so no distinction between "individual-based" and "agent-based" should be made in the future. Likewise, "multi-agent systems" are agent-based models which have their roots in computer science and artificial intelligence, but if they are used to tackle ecological problems, they are not fundamentally different from IBMs or ABMs.

Elements of Individual-based Models

A mathematical model consists of a set of equations, which are formulated in the general language of mathematics. For IBMs, equations and the language of mathematics are only applicable to submodels that describe certain processes, but the entire model consists of further elements, which are listed in the following. This list describes the decisions a modeler has to make while developing and formulating an IBM, and the details of each of these decisions can significantly influence the behavior of the IBM. In 2006, a general format for

describing IBMs was proposed, which is now widely used in ecology and increasingly also in other disciplines, the ODD (=Overview, Design concepts, Details) protocol. ODD cannot only be used to describe an existing model, but also as a hierarchical checklist for compiling and designing IBMs. It consists of seven numbered elements, with the first three providing an overview, the fourth being a list of relevant Design concepts, and remaining ones providing Details of model initilization, data input, and submodels.

1. Purpose

What is the specific purpose of the model, and what patterns and criteria are used to decide that a model is realistic enough for this purpose?

2. Entities, state variables, and scales

IBMs contain at least two types of entities: individuals and their environment. In the simplest case, there is only one type of individuals, for example one plant species, and a homogeneous environment in which the individuals are located. More complex IBMs consider more than one type of individual, for example a plant species, a herbivore, and a predator, and a more complex environment, for example a heterogeneous landscape consisting of different types of habitat.

The entities of the IBM are characterized by a set of state variables or behavioural attributes. Individuals might have one or more state variables, e.g. numbers representing position, age, sex, size, stage, rank, condition, energy reserves, memory of suitable habitat, etc., or being risk prone, highly mobile, or aggressive. It dependson the question addressed with the model which state variables are included, but the general guideline is to keep the representation of the individual as simple as possible. The environment often is represented as a grid of square grid cells, which represent spatial units of the landscape and may be characterized by the state variables habitat/non-habitat, cover, biomass, moisture, temperature, soil type, exposure to wind or predators, etc.

A further important design decision of an IBM is its spatial and temporal resolution and extent. The spatial resolution is determined by the size of the grid cells, which in ecology can range from cm^2 to km^2, and spatial extent is set by the number of grid cells, e.g. 100×100 cells. The temporal resolution is given by the length of a time step, which often is a year, a week, a day, or even less than an hour. Different parts of a year may also be represented with different resolution. The temporal extent is set by the time horizon considered, e.g. 100 years.

3. Process overview and scheduling

Processes cause changes of the state variables and, in case of individuals, also of the number of entities. Examples of processes in IBMs include energy budgets, growth, dispersal, foraging, habitat selection, mating, habitat dynamics, disturbance events, phenology, mortality, reproduction, etc. The modeler has to decide which process to represent explicitly, and in which detail. This decision is linked to the decisions on state variables and scales. If, for example, temporal resolution is one year, the process of foraging may not need to be represented explicitly. If, on the other hand, habitat selection in response to short-term environmental changes is considered important for the problem addressed with the model, a temporal resolution of one day or even less might be required, and in turn an explicit representation of the individual's decision where to move in the next time step.

If processes are not considered important enough to be included explicitly, they usually are represented by constant parameters. Mortality, for example, can explicitly depend on foraging and interaction with predators; or, mortality is represented by a constant parameter, which in the model is implemented as the probability of dying in a certain time step.

The submodels implementing the model processes are often formulated as a combination of mathematical expressions and IF-THEN conditions. Growth, for example, can be represented by a growth equation, whereas habitat choice will require probabilistic IF-THEN rules: „IF the best habitat within my perception range has higher quality then my current habitat AND IF predation risk is low THEN with a certain probability I will move to the new habitat."

All design decisions of the modeler regarding entities, processes, scheduling etc. are experimental, i.e. they have to be tested and analyzed and, as a result, usually be modified. A further important design decision of the modeller is the scheduling of the processes: who does what at what time in what order? IBMs are implemented as computer simulations, so processes cannot, as in reality, run in parallel, but only one after the other. The sequence in which processes are executed can be decisive for the resulting dynamics. An IBMs schedule describes the actions of the IBMs and how they are executed. An action is a list of model entities, the processes performed by these entities, and the order in which the entities are processed. For example, the action "feeding" may be defined as the list of all individuals which feed, one after the other in a fixed order, in their habitat cell. Or, feeding might be an action where all individuals first move to the neighbor habitat which has most food and then feed and where the individuals are processed in a random order.

Fixed schedules, which are used in most IBMs, define a single order in which events always occur, i.e. a cycle which is repeated every time step. Dynamic schedules as in discrete-event simulation allow the order to be changed while the model executes. For example, an individual that has just had an interaction with a predator and survived may put itself on a higher rank in the model's schedule, which would mimic its fleeing behavior. Flow charts, which are often used to illustrate how a model works, usually refer to model processes itself, for example dispersal, but not necessarily to the models schedule and actions. Table 1 shows a typical schedule of an IBM.

4. Design concepts

Many IBMs are designed ad hoc, without reference to any general theoretical or conceptual framework. Therefore, general concepts for the design for IBMs have been formulated, which are mainly taken from the research on Complex Adaptive Systems. These design concepts do not require that models necessarily have a certain structure, but their purpose is to make design decisions consciously. The most important design concepts are related to adaptive behavior: emergence vs. imposed properties, adaptation (does the model explicitly include behavior decisions), fitness (if adaptation is included, what fitness measures are used by the individuals), what do individuals know, and how do they predict the consequences of their decision alternatives? Further design concepts include: basic concepts, interaction, stochasticity, and collectives.

Particularly important for model testing and analysis is the design concept "observation". To test whether an IBM's implementation is correct and to understand how system-level dynamics emerge from the individual's interactions, specific observation tools are needed. These tools include graphical displays of patterns over space and time and the option to execute the model step by step and watch how state variables change. Some software platforms for IBMs like Swarm, Repast or NetLogo provide so-called monitors: on a graphical interface single individuals can be selected and their state variables displayed and manipulated (Fig. 2). Further observation tools are graphical and file output of summary statistics, for example average abundance, point-pattern characteristics of the individual's spatial distribution, or time series characteristics. Observation tools are not part of the model itself, but their appropriate choice determines what we can learn from an IBM.

5. Initialization

The outcome of an IBM simulation can critically depend on the initial number and state of the model entities. It is therefore important to test different initializations and to carefully document those which were used for the results presented in a publication. Sometimes, dependence on initial conditions is part of the question, for example if we want to understand whether a small, re-introduced population will establish. More often, however, we are interested in results which do not depend on initial conditions. To achieve this, we can pre-run the model until the distribution of the state variables becomes quasi-stationary and evaluate the model only from this time on.

6. Input data

IBMs may also be driven by environmental variables, which are not generated by the model itself, but read as an input to the model from external files. Annual rainfall, for example, may be calculated by a rainfall submodel, or, if long-term rainfall data are available, they are taken from an input file. All inputs to an IBM must be carefully documented and, if possible, made available.

7. Submodels

The processes listed in the ODD element 3, Process overview and scheduling, are implemented in submodels which are described here in all detail, using the same submodel names as in the overview and the program implementing the model. “In all detail” means that all information should be provided to allow for a complete re-implementation of the model, so that replication, the cornerstone of the scientific method, is also guaranteed for IBMs. This implies that here also all parameter values used should be listed in a table.

For complex models, the Submodels section often gets too long to be included in a journal article or book chapter, so that the full ODD has to be provided in an electronic supplement, while the main text only includes a summary description of the Overview part and the most important submodels.

Optimizing Model Complexity and Dealing with Uncertainty

IBMs are more complex than most mathematical population models, because they consider local interactions of individuals that are different and autonomous. This fact has led to the stereotype that IBMs are so complex that they are as hard to understand as nature itself, but this certainly is not true because even the most complex IBMs still are simplified representations of reality. Nevertheless, complexity is a challenge. Many IBMs seem to be more complex than really needed; specific techniques to test and analyze complex IBMs are not routinely used, which severely limits the potential for general insights. However, the general strategy of pattern-oriented modeling (POM) has been formulated which allows to optimize model complexity. POM is also useful for dealing with another important challenge of individual-based modeling: uncertainty of model structure and parameters. How can we know which process to include in a model and how to represent this process, for example dispersal? And how can we determine uncertain parameters?

POM is based on the notion that patterns observed in real systems are indicators of the system’s internal organization. If we are able to understand how these patterns emerge then we learned about key processes that determine the systems structure and drive its dynamics. A pattern is defined as anything beyond random variation, or any signal beyond noise. Some patterns are striking, like cycles in the abundance of small mammals, outbreaks of forest insects, or patchy and wave-like spatial patterns. Other patterns are less obvious but nevertheless contain information about the system’s key processes: patterns in age and size

structure, typical recovery dynamics after disturbance events, the fact that certain system-level state variables like biomass or number of species remain within certain limits, etc.
Patterns are the key to decode the internal organization of any system: atomic spectra helped decoding the structure of atoms, the red shift in the light of galaxies helped formulating the big bang theory, etc. With complex systems, however, that consist of many interacting building blocks, single patterns are not sufficient to narrow down our theories about the system to one single model. For example, there are at least eight models that explain the cycles of small mammals in the boreal zone, and more than 20 theories that explain why ecosystems near the equator have much more species than in other zones.

Thus, for complex systems we need multiple patterns observed at different scales and hierarchical levels. For example, Chargaff's rule of DNA base pairing was not sufficient to decode the structure of DNA. Two additional patterns helped to narrow down the model of the structure to the double helix: patterns from x-ray diffraction of DNA and tautomeric properties of the purine and pyrimidine bases.

Patterns for model structure

The most important guide for designing a model is the question addressed. This question, or problem, allows us to decide, in an experimental way, which element of the real system to represent in the model and at which resolution. With POM, in addition we are asking: what patterns can we observe? If we agree that a certain pattern is typical, or even essential for describing the system's identity, we should chose a model structure that in principle allows the same pattern to emerge in the model. This means in particular that we have to include the state variables of the pattern. If, for example, the pattern is spatial, we need to include spatial variables in the model; if the pattern is in size distributions, we need to include size as an individual's state variable; if the pattern is a response to a disturbance event, e.g. a drought, we need to include soil moisture as an environmental state variable, etc. Patterns thus make the choice of the model structure and complexity less arbitrary and more directly linked to the internal organization of the real system. The task of the modeler then is to check whether the model is able to reproduce the observed patterns. If not, key processes might be still be missing in the model.

Patterns for testing alternative theories

The first milestone in developing an IBM is a first version of the model that, without too much fine-tuning of parameters, largely behaves like the real system. For example a population which largely has the right size and spatial distribution. The question then is, how can we find the best representation of a certain process, for example habitat selection, mate choice, or life history? The answer is to follow the scientific method of strong inference and formulate alternative models, or theories, of the process and then to check how good the entire IBM, including one of the alternative theories, is capable of reproducing a set of observed patterns. In this way model development becomes less ad hoc, more rigorous, and it becomes easier to communicate why a certain model formulation has been chosen.

Patterns for reducing parameter uncertainty

With IBMs, easily too many parameters are too uncertain, or even completely unknown, to use the model for any practical purpose. It has also been argued that the complexity of IBMs leads to error propagation: even moderate uncertainties of individual parameters would multiply to such a large uncertainty at the system level that the model becomes useless. If, however, the IBM was designed according to POM, it should be structurally realistic, i.e. reflect the key structures of the real systems and allow for independent predictions for model validation. This structural realism makes it possible to determine entire sets of unknown parameters by inverse modeling: the model is fitted to the entire set of observed patterns. Technically, this is straightforward: for each unknown parameter, a range and a number of

values within this range is specified. Then the set of all possible parameter combinations is constructed, which can be very large. Specific techniques, like Latin Hypercube Sampling, exist for reducing the number of parameter sets, but typically thousands of parameter sets need to be analyzed. The model is run for each parameter set and checked if it is able to reproduce a certain pattern, otherwise the parameter set is discarded. This is repeated for two or more patterns. Eventually, typically less then 50 parameter sets remain that are able to reproduce all patterns simultaneously (Table 2). It has been shown in several examples that uncertainty in model output of this remaining parameter set is largely reduced, so that the model can even be used to support management decisions.

Analyzing Individual-based Models

IBMs have to be constructed in an iterative way. The first model version should be very simple, for example by making the environment constant, or consider identical individuals, or by representing a certain process by a constant parameter only, etc. This first, or Null, version of the model is deliberately oversimplified. It serves the purpose to start the iterative process of model development as soon as possible. This is done by providing a first set of tools for analyzing the model, for example graphical output, summary statistics, etc. Once we have these observation tools implemented, we can start refining the model, testing its implementation, and comparing it to observed patterns. Analyzing and developing the model then is a time consuming and complex task, but it can be performed as rigorous as real experiments and allows us to understand the relative importance of different processes and how individual behavior is related to system-level properties, and vice versa. In the following, three main elements of analyzing IBMs are explained in more detail.

Currencies

IBMs contain much more information about the state of all the individuals and their environment than we can process. We thus need to aggregate this information into one or more aggregated system-level state variables. We need to define currencies (often also referred to as “indices”) that allow us to characterize a single run of the model over a certain time horizon by, ideally, one single number. This number is then used as a currency to compare different parameterizations or formulations of the model. For example, in models addressing extinction risk of small populations, the risk of extinction over a certain time horizon is such a currency; or, if we want to compare model output to an observed time series, we can use root mean square deviation as a currency. In general, any quantitative measure of how good a certain observed pattern is reproduced is a good currency. Often, it is not clear from the outset which currency allows for deepest insights, so currencies are experimental and have to be tested and we usually we will need to consider a set of currencies simultaneously.

Simulation experiments

Once we have a first version of the model that runs over the time horizon of interest and have both first observation tools and currencies implemented, we can start doing science with the model by performing controlled simulation experiments. As in real experiments, we try to create situations where all but one parameter is kept constant; we try to design experiments whose results are easy to predict just by reasoning. If our predictions fail, we either design even simpler experiments, or we modify our predictions, etc. It is thus useful to consider an IBM as a virtual laboratory. As in real laboratories, we build up understanding step by step. We might, however, not necessarily end up with a full comprehensive understanding of how the model system works. We should, however, be able to say under which conditions the model produces certain types of outputs.

Robustness analysis

With many IBMs, we are interested in understanding one fundamental property of real systems, for example persistence of a population, diversity of a community, or coexistence of different life forms, for example trees and grass in savannas. The first task of model development and analysis is to reproduce the desired property with the model. But a model might produce a certain phenomenon only for a very restricted range of parameter combinations, which would make it unlikely that the real system has that property for the same reasons as the model. What we want to achieve are robust explanations that do not depend too much on details of the IBMs formulation and parameters (the same holds for any type of simulation model). For example, IBMs reproducing schooling behavior of fish turned out to be quite robust, even in a quantitative way, to details of how the interactions of neighbor fish is described, as long as the principle mechanism was included that the influence of neighbor fish is averaged in some way. Similarly, complex IBMs of tropical and temperate forests turned out to be quite robust to changes in many model parameters. This robustness reflects internal feedbacks, for example between mortality, recruitment, and dynamics of gaps in the canopy. Such feedbacks are likely to play a similar role in real forests.
Robustness analyses thus means to test the robustness of key model results to changes in model structure and parameters, and also to identify thresholds, for example critical parameter values, or key mechanisms that have to be in the model in order to get the desired model output. Robustness analyses thus helps to check how much of the model's complexity really is needed for explaining the real system's internal organization.

Implementing Individual-based Models

Implementing an IBM as a computer program, including tools for observation and analyses of a large number of parameterizations and formulations, can be quite challenging. Many tools exist in computer science that support the development and maintenance of complex software, but ecologists usually have no training in computer science. Most developers of IBMs thus are writing their software from scratch, without using concepts and tools of, for example, object-oriented programming. This makes the resulting software prone to errors and its development very inefficient.

There are mainly two alternatives to programming from scratch: (i) Software libraries. The modeler still has to use a certain all-purpose programming language but can utilize many predefined elements that support the implementation of IBMs. Examples of such libraries include Swarm (based on Objective C), Repast (Java), and Mason (Java). These libraries are supported by active user communities and their developers, but often are lacking a comprehensive documentation and tutorials. The learning curve for beginners is quite steep, but for the experienced they provide a very powerful framework for implementing IBMs. (ii) Modeling environments. They consist of menus or simplified programming languages that are easy to use and learn and allow to very quickly developing prototype IBMs. Examples include CORMAS, and NetLogo. NetLogo is freely available on the internet, is well-documented, comes with a good tutorial and many example models, and is actively maintained by its developers. Comparisons with Swarm and Repast showed that NetLogo is much less limited in performance and scope than one might expect due to NetLogo's history as a teaching environment. Being first designed for education and later being recommended for beginners in modelling, NetLogo is now increasingly used for entire modelling projects, sometimes of high complexity. In social simulation NetLogo has become the dominant software platform, while in ecology its use is increasing, a trend that is fostered by textbooks that were published since 2012.

Modern personal computers usually have enough memory and power for developing and running IBMs, but vast analyses of parameter space, for example for parameterization,

may need the combined power of PC clusters. For analyzing the output of IBMs, many modelers are using other software packages, for example R, Matlab, Mathematica, Excel, SPSS, etc. For NetLogo, also direkt links to R exist.

Examples

Three example IBMs are briefly described. The coyote model is an example of an IBM that does not explicitly include adaptive behavior. Rather, individuals behave according to empirically determined rates or probabilities. The shorebird model, in contrast, explicitly represents adaptive behavior: each bird is, on a time scale of several hours, making decisions where to move and feed in the habitat. Gap models, finally, have a very long and successful history in ecological modeling of forests.

Coyote model

The coyote model of W. C. Pitt and coworkers is a good example of an IBM that describes a species with a quite complex social behavior but nevertheless is quite simple. The purpose of the model is to support management decisions. Individuals have the state variables sex, age, social status (alpha, beta, pup), and the group (pack) they belong to. The model considers packs but not territories, and is thus not spatially explicit.

The most important rules of the coyote model are for individuals leaving the group and density-dependent mortality and reproduction. Coyotes between one-half and two years old have a probability of leaving their pack that is proportional to the square of pack size. Coyotes that leave their pack enter a pool of transients. Mortality of transients increases with the total number of transients and thus tends to be higher than that of pack members. The number of offspring produced is assumed to be density-dependent, i.e. to decrease with pack size.
The coyote model was implemented using the Swarm software library. This had the advantage that Swarm's framework for implementing an IBM's schedule could also be used for communicating this schedule (Table 1). This makes is easier to understand and re-implement the model. The schedule also shows that the coyote model consists of actions that are very simple. The complexity of IBMs is more in its implementation and analysis, not necessarily in its formulation.

The model was verified by using five currencies: mean pack size, proportion of transients, average offspring survival rate, average litter size, and proportion of females breeding. Model prediction matched observations surprisingly good, even without fine-tuning of parameters that were taken from the literature. An important insight gained from the coyote model was that the transients are buffering population dynamics. On the one hand they limit, due to their density-dependent mortality, population growth. On the other hand they buffer the loss of alpha individuals.

Shorebird models

The shorebird models of J. D. Goss-Custard, R. A. Stillman, and coworkers are good examples of IBMs that needed to include adaptive behavior because empirical model rules would not have been sufficient. The IBMs were developed to predict the impact of land reclamation, resource harvesting, and recreation on the winter mortality of species of shorebirds and waterfowl, for example the oystercatcher (*Haematopus ostralegus*) in the Exe estuary in England. The IBMs had to predict the effect of new environmental conditions, for which no empirical rules or data were available. The models had thus to operate on basic principles, i.e. physiology and fitness-seeking feeding behavior that is based on adaptive decisions.

The tidal-flat habitat is divided into discrete patches, which vary in their exposure and their quantity and type of food. During each time step birds choose where and on what to feed, or whether to roost. Times steps typically represent 1-6 hours. The bird's state variables

include foraging efficiency, dominance, location, diet, assimilation rate, metabolic rate, and amount of body reserves. Key environmental inputs to the models are the timings of ebb and flow and temperature, which both affect feeding and the amount of food needed to survive.

A main behavioral process of the model is interference competition (e.g., food stealing), which is related to the individual's dominance status and to local bird density on the patches. The submodels describing the bird's decision where to move, what to eat, and how much time to spend feeding, are based on first principles from optimal foraging theory and game theory. The individuals are assumed to always try and maximize their own chance of survival.

Model predictions were compared with many observed patterns, and after several iterations of the modeling cycle, patch selection, prey choice, and the proportion of time spent feeding were accurately predicted for many species and sites. In one case, the increase in winter mortality due to land reclamation was known from observations. The model was parameterized for the pre-impact situation, then run for the situation with reduced feeding area and the increase in winter mortality determined. The match of observed and predicted increase in winter mortality was almost perfect. It could also be shown that the model if it had existed at the time the land reclamation took place, could have been used to recommend a certain mitigation measure that was under discussion but not realized because it was unclear whether it could really compensate the loss of original feeding areas.

The first shore-bird model of the group of Goss-Custard and Stillman needed several years for implementation, parameterization, and testing, but subsequently simpler and more flexible models were developed, that could be used for management support within one to two years. Since then, the models has been used for a suite of species of shorebirds, waterfowl, and other bird species at more than 20 different sites all over Europe.

Gap models of forests

Gap models describe the change of species composition on a gap in the forest that was created by death of a canopy tree. Typical gap size in early gap models is 0.01 ha. The entire forest is assumed to be an ensemble of gaps with temporally and spatially independent dynamics. The purpose of gap models is to understand long-term species composition and succession and how they depend on environmental variables. Gap models were thus developed by ecologists, not forest managers.

The representation of the individuals, the trees, is extremely simple: they have only one state variable, trunk diameter at breast height (dbh), which is a standard measure of size in forestry. Tree height is calculated from dbh using an empirical relationship. The structure of gap models is extremely simple: each tree grows according to a sigmoidal potential growth curve. Potential annual growth is then reduced by multipliers which reflect the influence of competition and environmental factors. Competition is only considered vertically and calculated from the vertical distribution of light that is determined by the trees existing in the gap.

Thus, each time step of the model, which usually represents a year, first the vertical light profile is calculated, and then the growth increment. Different species are characterized by different parameters of the growth equation and the relationship between dbh and height. Mortality depends on the growth rate: trees that do not grow for a certain time span, i.e. are suppressed by larger trees, will die sooner or later.

Starting from the pioneering model JABOWA, more than 30 gap models have been developed for a wide range of forest types and questions. More recent gap models try to be more realistic in some way, for example by including spatial interactions between neighboring gaps. The great success of gap models has three main reasons: they are conceptually very simple, their growth equations are relatively easy to parameterize, and they make important testable predictions regarding species composition and dynamics of real

forests. Modern gap models, which are more complex but also more flexibly, include representations of photosynthesis and thereby link forest models to the global carbon cycle. Using information from satellite images it is nowadays possible to model, e.g., the entire Amazon rain forest, taking into account each individual tree.
.

The Future of Individual-based Modeling

IBMs are a flexible and powerful tool and very likely to lead to important insights into how system-level properties of ecological systems, for example stability properties (e.g., resilience), emerge from the interactions of the individuals among each other and with their environment. The approach poses also new challenges, in particular optimization of model complexity, coping with uncertainty in model structure and parameters, formulating the models according to a unifying framework, and implementing, testing, and communicating IBMs according to general standards.

IBMs will continue to be used both in a more pragmatic and a more paradigmatic way. Pragmatic IBMs usually do not refer to adaptive behavior and often are designed to be as compatible with more simple mathematical models as possible. There are certainly many questions where this type of IBM is sufficient, because, as we have seen in the coyote model and gap models described above, for many questions we do not need to refer to adaptive behavior explicitly.

Paradigmatic IBMs, however, are based on the assumption that adaptive behavior, i.e. the simple fact that individuals adapt their behavior to their current situation and that they are seeking to maximize fitness, is key to understanding most, if not all phenomena at the system level. Paradigmatic IBMs are seen as the nucleus of an Individual-based Ecology which links individual behavior, including life history and phenotypic plasticity, to system-level structures and dynamics. Next-generation IBMs are based on first principles, use information on trait distributions and remotely-sensed data, and are tested and parameterized with multiple patterns, observed at different scales and levels of observation. They make testable predictions about functional relationships, and they are designed to link theory and application via high levels of structural realism.

Further readings

Botkin, D. B., Janak, J. F. and Wallis, J. R. (1972). Some ecological consequences of a computer model of forest growth. *Journal of Ecology* **60**, 849-873.

Grimm, V., Berger, U., DeAngelis, D.L., Polhill, G., Giske, J., Railsback, S.F. (2010) The ODD protocol: a review and first update. *Ecological Modelling* **221**, 2760-2768.

Grimm, V., Railsback, S. F. (2012) Pattern-oriented modelling: a 'multi-scope'for predictive systems ecology. *Phil. Trans. R. Soc. B*, **367**, 298-310.

Grimm, V., Berger, U. (2016) Structural realism, emergence, and predictions in next-generation ecological modelling: synthesis from a special issue. *Ecological Modelling* **326**, 177-187.

Liu, J. and Ashton, P. S. (1995). Individual-based simulation models for forest succession and management. *Forest Ecology and Management* **73**, 157-175.

Pitt, W. C., Box, P. W. and Knowlton, F. F. (2003). An individual-based model of canid populations: modelling territoriality and social structure. *Ecological Modelling* **166**, 109-121.

Railsback, S.F., Grimm. V. (2012) *Agent-based and Individual-based Modeling: A Practical Introduction*. Princeton University Press, Princeton, N.J.

Rödig, E., Cuntz, M., Heinke, J., Rammig, A., & Huth, A. (2017). Spatial heterogeneity of biomass and forest structure of the Amazon rain forest: Linking remote sensing, forest modelling and field inventory. *Global Ecology and Biogeography* **26**, 1292-1302.

Stillman, R. A., & Goss-Custard, J. D. (2010). *Individual-based ecology of coastal birds. Biological Reviews* **85**: 413-434.

Strand, E. (2003). *Adaptive models of vertical migration in fish*. Doctoral thesis. Bergen, Norway, University of Bergen.

Wiegand, T., Revilla, E., and Knauer, F. (2004). Dealing with uncertainty in spatially explicit population models. *Biodiversity and Conservation* **13,** 53-78.

Wilensky, U. (1999) NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Figures

Figure 1.
Real individuals have to make adaptive decisions all the time, i.e. decisions that minimize their risk of starving or being eaten, and maximizing the chance to reproduce. For example, for many fish species vertical migration is a key behavior. Moving up usually means better conditions for feeding, but also higher risk of predation. The very decision made by an individual depends on many internal and external variables, i.e. we assume the fish knows, or at least has estimates, of these variables (modified after Strand 2003).

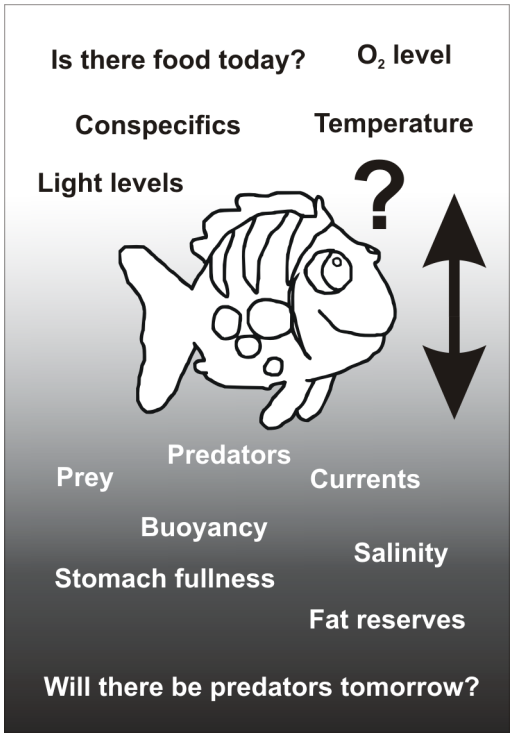
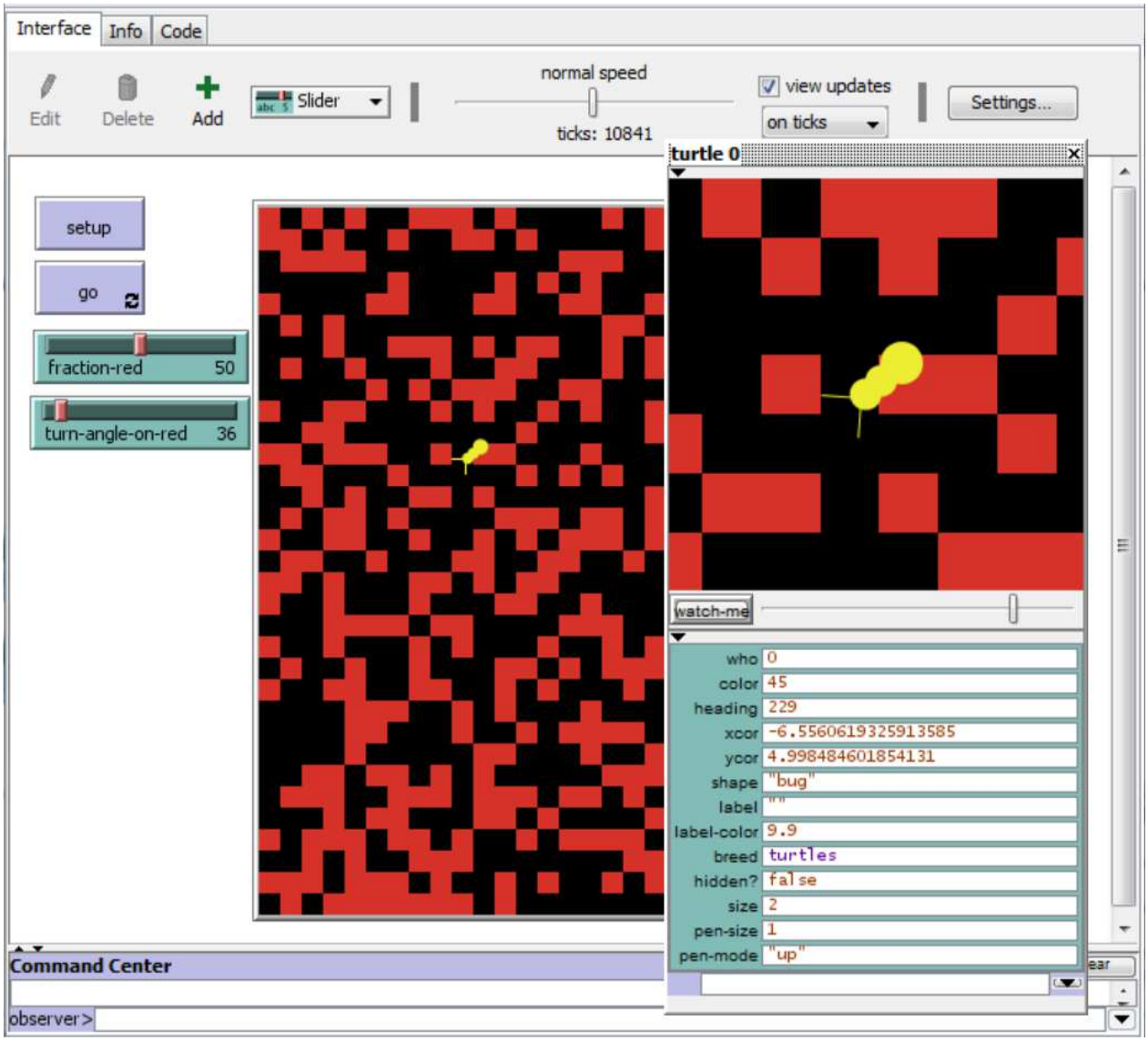


Figure 2
Screenshot of the interface of a typical NetLogo implementation of an IBM. The interface consists of the grid representing the space, which consists of grid cells (patches), buttons to setup and run the model, sliders to change parameters. Further elements that are added by drag and drop are plots, output windows for text, etc. Individual patches and agents (=individuals) can be inspected (see window labeled ‚turtle 0'), i.e. all their built-in and user-defined state variables displayed while the model is executed. State variables can also be modified interactively. Besides of the Interface tab, the Info tab contains a verbal description of the model, and the Code tab the program, which is using a programming language, NetLogo, that contains numerous commands for dealing with patches and agents (see NetLogo web pages). Note that typical NetLogo applications include more then two types of patches and many agents that interact with each other and the environment, i.e. the patches.



Tables

Table 1. Scheduling of the coyote model of Pitt and co-workers as an example of how the schedule of IBMs is organized into actions and IF-THEN rules.

Pack actions (executed by all packs):

- Check whether both an alpha male and an alpha female are present.
- If both alphas exist, and it is April, produce offspring: create pups, the number of which is stochastic but also depends on pack size, and add them to the pack.
- Check whether either alpha is replaced:
  - If it is December, and there is a contender (another adult of the same sex in the pack), both the male and female alpha coyotes are at risk of being replaced.
  - Replacement is a stochastic function with the probability of being replaced increasing with the alpha's age.
  - If replacement occurs, the alpha becomes a transient and the contender becomes the new alpha.
- Update the dispersal probability of each member according to its age and pack size.
- Force death of pups less then 2 months old if the pack has no adults.

Pack member actions (executed by all individual coyotes that belong to a pack):

- If the age of two months is attained, leave the den.
- If the age of six months is attained, change from pup to beta adult.
- Update the age-dependent mortality probability and determine whether death occurs.
- If individual is a beta less than two years old, determine whether it leaves the pack, according to its dispersal probability.

Transient coyote actions (executed by all individuals not belonging to packs):

- Update individual's mortality probability, depending on the total number of transients.
- Determine whether death occurs.

Pack alpha replacement actions (executed by packs that lack a alpha individual):

- If there are beta individuals of the appropriate sex in the pack, promote the oldest beta to alpha.
- Otherwise, select a transient of the appropriate sex and promote it to alpha.
- If there are no available transients, select a beta from another pack.

Table 2. Pattern-oriented parameterization of an IBM describing brown bears spreading from Slovenia into the Alps. Five different observed patterns are used as filters to reduce the number of possible paramter sets. Note that pattern 5 is as good a filter as patterns 2, 3 and 4 in combination (modified after Wiegand et al. 2004).

Pattern description	Patterns	Number of model parameterizations in agreement with observed pattern
No filter	0	557
Density of females in transition area	1	506
Bear observation in central Austria	2	138
Bear observation in the Carnic Alps	3	154
Bear observation in the Karawanken	4	180
Census time series of females with cubs	5	12
	2+3+4	13
	5+1	10
	2+3+4+1	11