Towards a standard model for research in agent-based modeling and simulation

Nuno Fachada, Vitor V Lopes, Rui C Martins, Agostinho C Rosa

Agent-based modeling (ABM) is a bottom-up modeling approach, where each entity of the system being modeled is uniquely represented as an independent decision-making agent. ABMs are very sensitive to implementation details. Thus, it is very easy to inadvertently introduce changes which modify model dynamics. Such problems usually arise due to the lack of transparency in model descriptions, which constrains how models are assessed, implemented and replicated. In this paper, we present PPHPC, a model which aims to serve as a standard in agent based modeling research, namely, but not limited to, conceptual model specification, statistical analysis of simulation output, model comparison and parallelization studies. This paper focuses on the first two aspects (conceptual model specification and statistical analysis of simulation output), also providing a canonical implementation of PPHPC. The paper serves as a complete reference to the presented model, and can be used as a tutorial for simulation practitioners who wish to improve the way they communicate their ABMs.

Towards a standard model for research in agent-based modeling and simulation

Nuno Fachada[1], **Vitor V. Lopes**[2], **Rui C. Martins**[3], **and Agostinho C. Rosa**[4]

[1,4]**Institute for Systems and Robotics, LARSyS, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal**
[2]**Universidad de las Fuerzas Armadas-ESPE, Sangolquí, Ecuador**
[3]**Life and Health Sciences Research Institute, School of Health Sciences, University of Minho, Braga, Portugal**

ABSTRACT

Agent-based modeling (ABM) is a bottom-up modeling approach, where each entity of the system being modeled is uniquely represented as an independent decision-making agent. ABMs are very sensitive to implementation details. Thus, it is very easy to inadvertently introduce changes which modify model dynamics. Such problems usually arise due to the lack of transparency in model descriptions, which constrains how models are assessed, implemented and replicated. In this paper, we present PPHPC, a model which aims to serve as a standard in agent based modeling research, namely, but not limited to, conceptual model specification, statistical analysis of simulation output, model comparison and parallelization studies. This paper focuses on the first two aspects (conceptual model specification and statistical analysis of simulation output), also providing a canonical implementation of PPHPC. The paper serves as a complete reference to the presented model, and can be used as a tutorial for simulation practitioners who wish to improve the way they communicate their ABMs.

Keywords: agent-based modeling, standard model, statistical analysis of simulation output, ODD

INTRODUCTION

Agent-based modeling (ABM) is a bottom-up modeling approach, where each entity of the system being modeled is uniquely represented as an independent decision-making agent. When prompted to act, each agent analyzes its current situation (e.g. what resources are available, what other agents are in the neighborhood), and acts appropriately, based on a set of rules. These rules express knowledge or theories about the respective low-level components. The global behavior of the system is the result from the simple, self-organized local relationships between the agents (Fachada, 2008). As such, ABM is a useful tool in simulating and exploring systems that can be modeled in terms of interactions between individual entities, e.g., biological cell cultures, ants foraging for food or military units in a battlefield (Macal and North, 2008). In practice, ABM can be considered a variation of discrete-event simulation, since state changes occur at specific points in time (Law, 2015).

Spatial agent-based models (SABMs) are a subset of ABMs in which a spatial topology defines how agents interact (Shook et al., 2013). For example, an agent may be limited to interact with agents located within a specific radius, or may only move to a near physical or geographical location (Macal and North, 2010). SABMs have been extensively used to study a range of phenomena in the biological and social sciences (Isaac, 2011; Shook et al., 2013).

ABMs are very sensitive to implementation details: the impact that seemingly unimportant aspects such as data structures, algorithms, discrete time representation, floating point arithmetic or order of events can have on results is tremendous (Wilensky and Rand, 2007; Merlone et al., 2008). As such, it is very easy to inadvertently introduce changes which will alter model dynamics. These type of issues usually derive from a lack of transparency in model descriptions, which constrains how models are assessed and implemented (Müller et al., 2014). Conceptual models should be well specified and adequately described in order to be properly implemented and replicated (Edmonds and Hales, 2003; Wilensky and Rand, 2007).

The ODD protocol (Overview, Design concepts, Details) is currently one of the most widely used templates for making model descriptions more understandable and complete, providing a comprehensive checklist that covers virtually all the key features that can define a model (Grimm et al., 2010). It allows modelers to communicate their models using a natural language description within a prescriptive and hierarchical structure, aiding in model design and fostering in-depth model comprehension (Müller et al., 2014). It is the recommended approach for documenting models in the CoMSES Net Computational Model Library (Rollins et al., 2014). However, Müller et al. (2014) argue that no single model description standard can completely and throughly characterize a model by itself, suggesting that besides a structured natural language description such as ODD, the availability of a model's source code should be part of a minimum standard for model communication. Furthermore, the ODD protocol does not deal with models from a results or simulation output perspective, which means that an additional section for statistical analysis of results is often required. In practice, however, the situation is very different. While many ABMs have been published and simulation output analysis is a widely discussed subject matter (Sargent, 1976; Kelton, 1997; Law, 2007; Nakayama, 2008; Law, 2015), comprehensive inquiries concerning the output of ABM simulations are hard to find in the scientific literature.

In this paper, we present PPHPC (Predator-Prey for High-Performance Computing), a conceptual model which captures important characteristics of SABMs, such as agent movement and local agent interactions. It aims to serve as a standard in agent based modeling research, and was designed with several goals in mind:

1. Provide a basis for a tutorial on complete model specification and thorough simulation output analysis.

2. Investigate statistical comparison strategies for model replication (Fachada et al., 2015a).

3. Compare different implementations from a performance point of view, using different frameworks, programming languages, hardware and/or parallelization strategies, while maintaining statistical equivalence among implementations (Fachada et al., 2015b).

4. Test the influence of different pseudo-random number generators (PRNGs) on the statistical accuracy of simulation output.

This paper aims to fulfill the first of these goals, and is organized as follows. First, in 'Background', we review several paradigmatic ABMs, as well as model description and analysis. Next, the 'Methodology' section is divided into five subsections, in which we: a) formalize the conceptual model using the ODD protocol; b) describe the canonical PPHPC realization implemented with the NetLogo ABM toolkit (Wilensky, 1999); c) discuss how to select output focal measures; d) explain how to collect and prepare data for statistical analysis; and, e) propose how to analyze focal measures from a statistical point-of-view. In 'Results', statistical analysis of output of the NetLogo implementation is performed. A discussion on how these results can be utilized in additional investigations is undertaken in 'Discussion'. 'Conclusions' provides a global outline of what was accomplished in this paper.

BACKGROUND

Several ABMs have been used for the purpose of modeling tutorials and/or model analysis and replication. Probably, the most well known standard ABM is the "StupidModel", which consists of a series of 16 pseudo-models of increasing complexity, ranging from simple moving agents to a full predator-prey-like model. It was developed by Railsback et al. (2005) as a teaching tool and template for real applications, as it includes a set of features commonly used in ABMs of real systems. It has been used to address a number of questions, including the comparison of ABM platforms (Railsback et al., 2006; Lytinen and Railsback, 2012), model parallelization (Lysenko and D'Souza, 2008; Tang and Wang, 2009), analysis of toolkit feasibility (Standish, 2008) and/or creating models as compositions of micro-behaviors (Kahn, 2007). The "StupidModel" series has been criticized for having some atypical elements and ambiguities (Lytinen and Railsback, 2012), reasons which lead Isaac (2011) to propose a reformulation to address these and other issues. However, its multiple versions and user-interface/visualization goals limit the series appeal as a pure computational model for the goals described in the introduction.

Other paradigmatic models which have been recurrently used, studied and replicated include Sugarscape (Epstein and Axtell, 1996; Axtell et al., 1996; Bigbee et al., 2007; D'Souza et al., 2007; Lysenko

87 and D'Souza, 2008), Heatbugs (Wilensky, 2004; Sallach and Mellarkod, 2005; Goldsby and Pancerella,
88 2013), Boids (Reynolds, 1987, 2006; Goldsby and Pancerella, 2013) and several interpretations of
89 prototypical predator-prey models (Smith, 1991; Hiebeler, 1994; Wilensky, 1997; Tatara et al., 2006;
90 Ottino-Loffler et al., 2007; Ginovart, 2014). Nonetheless, there is a lack of formalization and in-depth sta-
91 tistical analysis of simulation output in most of these implementations, often leading to model assessment
92 and replication difficulties (Edmonds and Hales, 2003; Wilensky and Rand, 2007). This might not come
93 as a surprise, as most models are not implemented with replication in mind.

94 Many models are not adequately analyzed with respect to their output data, often due to improper
95 design of simulation experiments. Consequently, authors of such models can be at risk of making
96 incorrect inferences about the system being studied (Law, 2007). A number of papers and books have
97 been published concerning the challenges, pitfalls and opportunities of using simulation models and
98 adequately analyzing simulation output data. In one of the earliest articles on the subject, Sargent (1976)
99 demonstrates how to obtain point estimates and confidence intervals for steady state means of simulation
100 output data using a number of different methodologies. Later, Law (1983) presented a state-of-the-art
101 survey on statistical analyses for simulation output data, addressing issues such as start-up bias and
102 determination of estimator accuracy. This survey was updated several times over the years, e.g. (Law,
103 2007), where Law discusses the duration of transient periods before steady state settles, as well as the
104 number of replications required for achieving a specific level of estimator confidence. In (Kelton, 1997),
105 the author describes methods to help design the runs for simulation models and interpreting their output
106 using statistical methods, also dealing with related problems such as model comparison, variance reduction
107 or sensitivity estimation. A comprehensive exposition of these and other important topics of simulation
108 research is presented in the several editions of "Simulation Modeling and Analysis" by Law and Kelton,
109 and its latest edition (Law, 2015) is used as a starting point for the analysis described in 'Methodology'
110 and conducted in 'Results'.

METHODOLOGY

Overview, design concepts and details of PPHPC

112 Here we describe the PPHPC model using the ODD protocol (Grimm et al., 2010). Time-dependent state
113 variables are represented with uppercase letters, while constant state variables and parameters are denoted
114 by lowercase letters. The $U(a,b)$ expression equates to a random integer within the closed interval $[a,b]$
115 taken from the uniform distribution.

Purpose

118 The purpose of PPHPC is to serve as a standard model for studying and evaluating SABM implementation
119 strategies. It is a realization of a predator-prey dynamic system, and captures important characteristics
120 of SABMs, such as agent movement and local agent interactions. The model can be implemented using
121 substantially different approaches that ensure statistically equivalent qualitative results. Implementations
122 may differ in aspects such as the selected system architecture, choice of programming language and/or
123 agent-based modeling framework, parallelization strategy, random number generator, and so forth. By
124 comparing distinct PPHPC implementations, valuable insights can be obtained on the computational and
125 algorithmical design of SABMs in general.

Entities, state variables, scales

127 The PPHPC model is composed of three entity classes: *agents*, *grid cells* and *environment*. Each of these
128 entity classes is defined by a set of state variables, as shown in Table 1. All state variables explicitly assume
129 integer values to avoid issues with the handling of floating-point arithmetic on different programming
130 languages and/or processor architectures.

131 The t state variable defines the *agent* type, either s (*sheep*, i.e. prey) or w (*wolf*, i.e. predator). The
132 only behavioral difference between the two types is in the feeding pattern: while prey consume passive
133 cell-bound food, predators consume prey. Other than that, prey and predators may have different values
134 for other state variables, as denoted by the superscripts s and w. Agents have an energy state variable,
135 E, which increases by g^s or g^w when feeding, decreases by l^s or l^w when moving, and decreases by half
136 when reproducing. When energy reaches zero, the agent is removed from the simulation. Agents with
137 energy higher than r_T^s or r_T^w may reproduce with probability given by r_P^s or r_P^w. The grid position state
138 variables, X and Y, indicate which cell the agent is located in. There is no conceptual limit on the number
139 of agents that can exist during the course of a simulation run.

3/20

Entity	State variable	Symbol	Range
Agents	Type	t	w, s
	Energy	E	$1, 2, \ldots$
	Horizontal position in grid	X	$0, 1, \ldots, x_{\text{env}} - 1$
	Vertical position in grid	Y	$0, 1, \ldots, y_{\text{env}} - 1$
	Energy gain from food	g^s, g^w	$0, 1, \ldots$
	Energy loss per turn	l^s, l^w	$0, 1, \ldots$
	Reproduction threshold	r_T^s, r_T^w	$1, 2, \ldots$
	Reproduction probability	r_P^s, r_P^w	$0, 1, \ldots, 100$
Grid cells	Horizontal position in grid	x	$0, 1, \ldots, x_{\text{env}} - 1$
	Vertical position in grid	y	$0, 1, \ldots, y_{\text{env}} - 1$
	Countdown	C	$0, 1, \ldots, c_r$
Environment	Horizontal size	x_{env}	$1, 2, \ldots$
	Vertical size	y_{env}	$1, 2, \ldots$
	Restart	c_r	$1, 2, \ldots$

Table 1. Model state variables by entity. Where applicable, the *s* and *w* designations correspond to prey (*sheep*) and predator (*wolf*) agent types, respectively.

Instances of the *grid cell* entity class can be thought of the place or neighborhood where agents act, namely where they try to feed and reproduce. Agents can only interact with other agents and resources located in the same grid cell. Grid cells have a fixed grid position, (x, y), and contain only one resource, cell-bound food (*grass*), which can be consumed by prey, and is represented by the countdown state variable C. The C state variable specifies the number of iterations left for the cell-bound food to become available. Food becomes available when $C = 0$, and when a prey consumes it, C is set to c_r.

The set of all grid cells forms the *environment* entity, a toroidal square grid where the simulation takes place. The environment is defined by its size, $(x_{\text{env}}, y_{\text{env}})$, and by the restart parameter, c_r.

Spatial extent is represented by the aforementioned square grid, of size $(x_{\text{env}}, y_{\text{env}})$, where x_{env} and y_{env} are positive integers. Temporal extent is represented by a positive integer m, which represents the number of discrete simulation steps or iterations. Spatial and temporal scales are merely virtual, i.e. they do not represent any real measure.

Process overview and scheduling

Algorithm 1 describes the simulation schedule and its associated processes. Execution starts with an initialization process, `Init()`, where a predetermined number of agents are randomly placed in the simulation environment. Cell-bound food is also initialized at this stage.

After initialization, and to get the simulation state at iteration zero, outputs are gathered by the `GetStats()` process. The scheduler then enters the main simulation loop, where each iteration is sub-divided into four steps: 1) agent movement ; 2) food growth in grid cells ; 3) agent actions ; and, 4) gathering of simulation outputs.

State variables are asynchronously updated, i.e. they are assigned a new value as soon as this value is calculated by a process (e.g. when an agent gains energy by feeding).

Design concepts

Basic principles   The general concepts of this model are based on well studied predator-prey dynamics, initially through analytical approaches (Lotka, 1925; Volterra, 1926), and later using agent-based models (Smith, 1991). However, PPHPC is designed so that it can be correctly implemented using diverse computational approaches. Realizations of this model can provide valuable information on how to better implement SABMs on different computing architectures, namely parallel ones. In particular, they may shown the impact of different parallelization strategies on simulation performance.

Emergence   The model is characterized by oscillations in the population of both prey and predator, as well as in the available quantity of cell-bound food. Typically, a peak of predator population occurs slightly after a peak in prey population size, while quantity of cell-bound food is approximately in "phase

---

Algorithm 1 Main simulation algorithm. **for** loops can be processed in *any order* or in *random order*. In terms of expected dynamic behavior, the former means the order is not relevant, while the latter specifies loop iterations should be explicitly shuffled.

1: INIT()
2: GETSTATS()
3: $i \leftarrow 1$
4: **for** $i <= m$ **do**
5:      **for each** agent **do**   ▷ Any order
6:          MOVE()
7:      **end for**
8:      **for each** grid cell **do**   ▷ Any order
9:          GROWFOOD()
10:     **end for**
11:     **for each** agent **do**   ▷ Random order
12:         ACT()
13:     **end for**
14:     GETSTATS()
15:     $i \leftarrow i + 1$
16: **end for**

---

172 opposition" with the prey's population size.

173 **Sensing**   Agents can sense the presence of food in the grid cell in which they are currently located. This
174 means different thing for prey and predators. Prey agents can read the local grid cell C state variable,
175 which if zero, means there is food available. Predator agents can determine the presence of prey agents.

176 **Interaction**   Agents interact with sources of food present in the grid cell they are located in.

177 **Stochasticity**   The following processes are random: a) initialization of specific state variables ; b) agent
178 movement ; c) the order in which agents act ; and, d) agent reproduction.

179 **Observation**   The following vector is collected in the GetStats() process, where i refers to the
180 current iteration:

$$\mathbf{O}_i = (P_i^s, P_i^w, P_i^c, \overline{E}_i^s, \overline{E}_i^w, \overline{C}_i)$$

181 P_i^s and P_i^w refer to the total prey and predator population counts, respectively, while P_i^c holds the
182 quantity of available cell-bound food. $\overline{E}_i^s$ and $\overline{E}_i^w$ contain the mean energy of prey and predator populations.
183 Finally, $\overline{C}_i$ refers to the mean value of the C state variable in all grid cells.

184 ***Initialization***
185 The initialization process begins by instantiating the *environment* entity, a toroidal square grid, and filling
186 it with $x_{\text{env}} \times y_{\text{env}}$ grid cells. The initial value of the countdown state variable in each grid cell, C_0, is set
187 according to Eq. 1,

$$C_0 = \begin{cases} U(1, c_r), & \text{if } c_0 = 0 \\ 0, & \text{if } c_0 = 1 \end{cases}, \quad \text{with } c_0 = U(0,1) \tag{1}$$

188 In other words, cell-bound food is initially available with 50% probability. If not available, the
189 countdown state variable is set to a random value between 1 and c_r.
190 The initial value of the state variables for each agent is determined according to Eqs. 2 and 3.

$$E_0 = U(1, 2g), \quad \text{with } g \in \{g^s, g^w\} \tag{2}$$

$$(X_0, Y_0) = \big(U(0, x_{\text{env}} - 1), U(0, y_{\text{env}} - 1)\big) \tag{3}$$

Submodels

As stated in *Process overview and scheduling* , each iteration of the main simulation loop is sub-divided
into four steps, described in the following paragraphs.

Move()   In step 1, agents Move(), in any order, within a Von Neumann neighborhood, i.e. up, down,
left, right or stay in the same cell, with equal probability. Agents lose l^s or l^w units of energy when they
move, even if they stay in the same cell; if energy reaches zero, the agent dies and is removed from the
simulation.

GrowFood()   In step 2, during the GrowFood() process, each grid cell checks if $C = 0$ (meaning there
is food available). If $C > 0$ it is decremented by one unit. Eq. 4 summarizes this process.

$$C_i = \max(C_{i-1} - 1, 0) \tag{4}$$

Act()   In step 3, agents Act() in explicitly random order, i.e. the agent list should be shuffled before
the agents have a chance to act. The Act() process is composed of two sub-actions: TryEat()
and TryReproduce(). The Act() process is atomic, i.e. once called, both TryEat() and
TryReproduce() must be performed; this implies that prey agents may be killed by predators before
or after they have a chance of calling Act(), but not during the call.

   TryEat()   Agents can only interact with sources of food present in the grid cell they are located in.
Predator agents can kill and consume prey agents, removing them from the simulation. Prey agents can
consume cell-bound food, resetting the local grid cell C state variable to c_r. A predator can consume one
prey per iteration, and a prey can only be consumed by one predator. Agents who act first claim the food
resources available in the local grid cell. Feeding is automatic: if the resource is there and no other agent
has yet claimed it, the agent will consume it. Moreover, only one prey can consume the local cell-bound
food if available (i.e. if $C = 0$). When an agent successfully feeds, its energy E is incremented by g^s or
g^w, depending on whether the agent is a prey or a predator, respectively.

   TryReproduce()   If the agent's energy, E, is above its species reproduction threshold, r_T^s or r_T^w, then
reproduction will occur with probability given by the species reproduction probability, r_P^s or r_P^w, as shown
in Algorithm 2. When an agent successfully reproduces, its energy is divided (using integer division)
with its offspring. The offspring is placed in the same grid cell as his parent, but can only take part in the
simulation in the next iteration. More specifically, newly born agents cannot Act(), nor be acted upon.
The latter implies that newly born prey cannot be consumed by predators in the current iteration. Agents
immediately update their energy if they successfully feed and/or reproduce.

Algorithm 2 Agent reproduction.

---
function TRYREPRODUCE()
    if $E > r_T$ **then**
        if $U(0,99) < r_P$ **then**
            $E^{\text{child}} \leftarrow E/2$ ▷ Integer division
            $E \leftarrow E - E^{\text{child}}$
            NEWAGENT($t, E^{\text{child}}, X, Y$)
        end if
    end if
end function

Parameterization   Model parameters can be qualitatively separated into size-related and dynamics-
related parameters, as shown in Table 2. Although size-related parameters also influence model dynamics,
this separation is useful for parameterizing simulations.

   Concerning size-related parameters, more specifically, the grid size, we propose a base value of
100×100, associated with 400 prey and 200 predators. Different grid sizes should have proportionally
assigned agent population sizes, as shown in Table 3. In other words, there are no changes in the agent
density nor the ratio between prey and predators.

Type	Parameter	Symbol
Size	Environment size	x_{env}, y_{env}
	Initial agent count	P_0^s, P_0^w
	Number of iterations	m
Dynamics	Energy gain from food	g^s, g^w
	Energy loss per turn	l^s, l^w
	Reproduction threshold	r_T^s, r_T^w
	Reproduction probability	r_P^s, r_P^w
	Cell food restart	c_r

Table 2. Size-related and dynamics-related model parameters.

Size	$x_{env} \times y_{env}$	P_0^s	P_0^w
100	100×100	400	200
200	200×200	1600	800
400	400×400	6400	3200
800	800×800	25600	12800
1600	1600×1600	102400	51200
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 3. A selection of initial model sizes.

For the dynamics-related parameters, we propose two sets of parameters, Table 4, which generate two distinct dynamics. The second parameter set typically yields more than twice the number of agents than the first parameter set. Matching results with runs based on distinct parameters is necessary in order to have a high degree of confidence in the similarity of different implementations (Edmonds and Hales, 2003). While many more combinations of parameters can be experimented with this model, these two sets are the basis for testing and comparing PPHPC implementations. We will refer to a combination of model size and parameter set as "size@set", e.g. 400@1 for model size 400, parameter set 1.

Parameter	Symbol	Set 1	Set 2
Prey energy gain from food	g^s	4	30
Prey energy loss p/ turn	l^s	1	1
Prey reprod. threshold	r_T^s	2	2
Prey reprod. probability	r_P^s	4	10
Predator energy gain from food	g^w	20	10
Predator energy loss p/ turn	l^w	1	1
Predator reprod. threshold	r_T^w	2	2
Predator reprod. probability	r_P^w	5	5
Cell food restart	c_r	10	15

Table 4. Dynamics-related parameter sets.

While simulations of the PPHPC model are essentially non-terminating[1], the number of iterations, m, is set to 4000, as it allows to analyze steady-state behavior for all the parameter combinations discussed here.

A NetLogo implementation

NetLogo is a well-documented programming language and modeling environment for ABMs, focused on both research and education. It is written in Scala and Java and runs on the Java Virtual Machine (JVM). It

---

[1]A non-terminating simulation is one for which there is no natural event to specify the length of a run (Law, 2015).

uses a hybrid interpreter and compiler that partially compiles ABM code to JVM bytecode (Sondahl et al., 2006). It comes with powerful built-in procedures and is relatively easy to learn, making ABMs more accessible to researchers without programming experience (Martin et al., 2012). Advantages of having a NetLogo version include real-time visualization of simulation, pseudo-code like model descriptions, simplicity in changing and testing different model aspects and parameters, and command-line access for batch runs and cycling through different parameter sets, even allowing for multithreaded simultaneous execution of multiple runs. A NetLogo reference implementation is also particularly important as a point of comparison with other ABM platforms (Isaac, 2011).

The NetLogo implementation of PPHPC, Figure 1, is based on NetLogo's own *Wolf Sheep Predation* model (Wilensky, 1997), considerably modified to follow the ODD discussed in the previous section. Most NetLogo models will have at least a *setup* procedure, to set up the initial state of the simulation, and a *go* procedure to make the model run continuously (Wilensky, 2014). The `Init()` and `GetStats()` processes (lines 1 and 2 of algorithm 1) are defined in the *setup* procedure, while the main simulation loop is implemented in the *go* procedure. The latter has an almost one-to-one relation with its pseudo-code counterpart in Algorithm 1. By default, NetLogo shuffles agents before issuing them orders, which fits naturally into the model ODD. The implementation is available at `https://github.com/FakenMC/pphpc/tree/netlogo`.
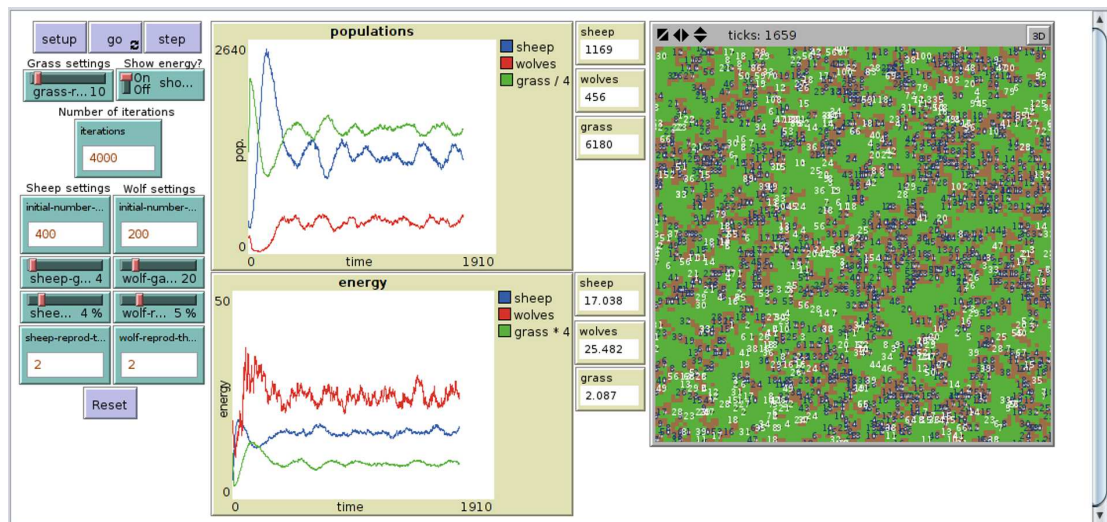
Figure 1. NetLogo implementation of the PPHPC model.

Selection of focal measures

In order to analyze the output of a simulation model from a statistical point-of-view, we should first select a set of focal measures (f.m.'s) which summarize each output. Wilensky and Rand (2007) use this approach in the context of statistical comparison of replicated models. Typically, f.m.'s consist of long-term or steady-state means. However, being limited to analyze average system behavior can lead to incorrect conclusions (Law, 2015). Consequently, other measures such as proportions or extreme values can be used to assess model behavior. In any case, the selection of f.m.'s is an empirical exercise and is always dependent of the model under study. A few initial runs are usually required in order to perform this selection.

For the PPHPC model, the typical output of a simulation run is shown in Figure 2 for size 400 and both parameter sets. In both cases, all outputs undergo a transient stage and tend to stabilize after a certain number of iterations, entering steady-state. For other sizes, the situation is similar apart from a vertical scaling factor. Outputs display pronounced extreme values in the transient stage, while circling around a long-term mean and approximately constant standard deviation in the steady-state phase. This standard deviation is an important feature of the outputs, as it marks the overall variability of the predator-prey cycles. Having this under consideration, six statistics, described in Table 5, where selected for each output. Considering there are six outputs, a total of 36 f.m.'s are analyzed for the PPHPC model.
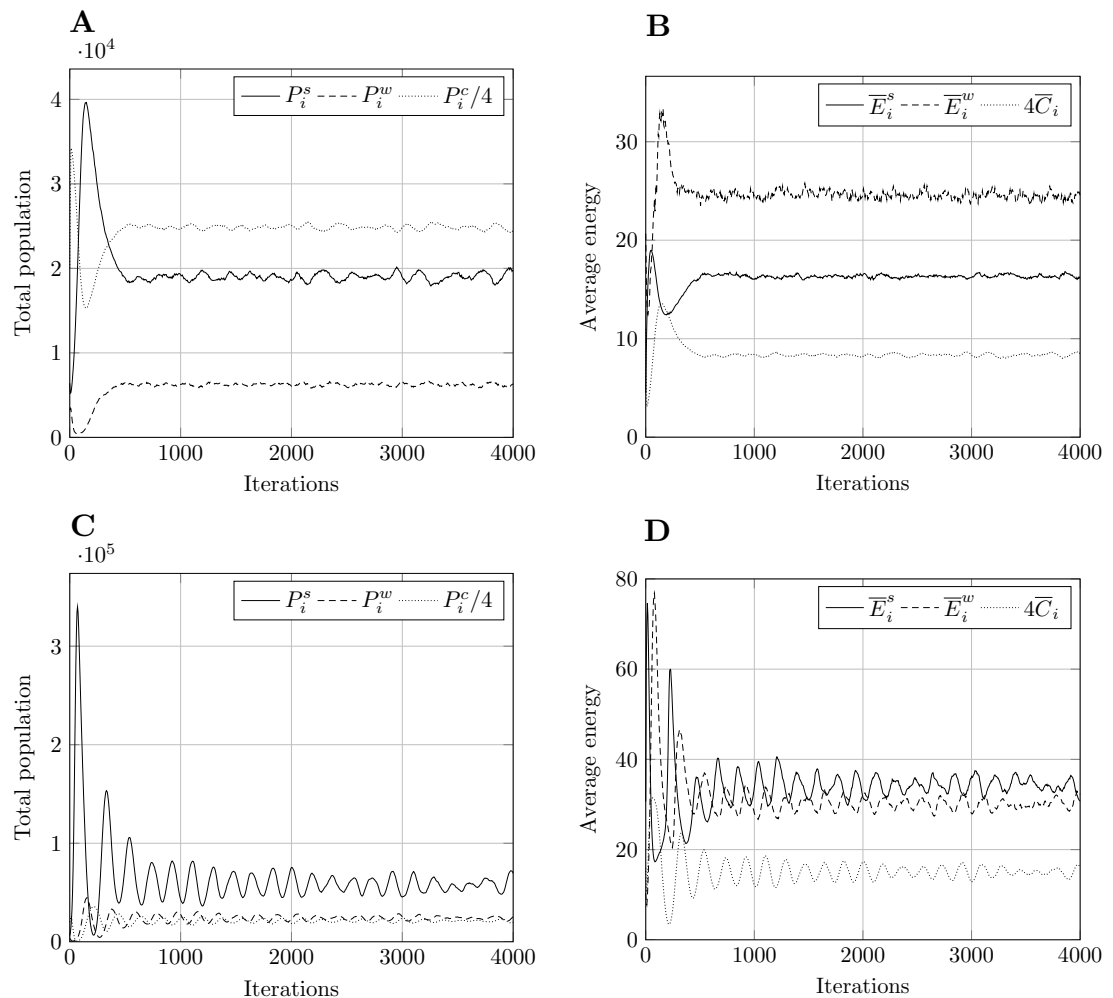
Figure 2. Typical model output for model size 400. Other model sizes have outputs which are similar, apart from a vertical scaling factor. P_i refers to total population, $\overline{E}_i$ to mean energy and $\overline{C}_i$ to mean value of the countdown state variable, C. Superscript s relates to prey, w to predators, and c to cell-bound food. P_i^c and $\overline{C}_i$ are scaled for presentation purposes. (A) Population, param. set 1. (B) Energy, param. set 1. (C) Population, param. set 2. (D) Energy, param. set 2.

Collecting and preparing data for statistical analysis

Let $X_{j0}, X_{j1}, X_{j2}, ..., X_{jm}$ be an output from the jth simulation run (rows under 'Iterations' in Table 6). The X_{ji}'s are random variables that will, in general, be neither independent nor identically distributed (Law, 2015), and as such, are not adequate to be used directly in many formulas from classical statistics (which are discussed in the next section). On the other hand, let $X_{1i}, X_{2i}, ..., X_{ni}$ be the observations of an output at iteration i for n runs (columns under 'Iterations' in Table 6), where each run begins with the same initial conditions but uses a different stream of random numbers as a source of stochasticity. The X_{ji}'s will now be independent and identically distributed (i.i.d.) random variables, to which classical statistical analysis can be applied. However, individual values of the output X at some iteration i are not representative of X as a whole. Thus, we use the selected f.m.'s as representative summaries of an output, as shown in Table 6, under 'Focal measures'. Taken column-wise, the observations of the f.m.'s are i.i.d. (because they are obtained from i.i.d. replications), constituting a *sample* prone to statistical analysis.

Regarding steady-state measures, $\overline{X}^{\text{ss}}$ and S^{ss}, care must be taken with initialization bias, which may cause substantial overestimation or underestimation of the long-term performance (Sanchez, 1999). Such problems can be avoided by discarding data obtained during the initial transient period, before the system reaches steady-state conditions. The simplest way of achieving this is to use a fixed truncation point, l, for

9/20

Statistic	Description
$\max\limits_{0 \le i \le m} X_i$	Maximum value.
$\arg\max\limits_{0 \le i \le m} X_i$	Iteration where maximum value occurs.
$\min\limits_{0 \le i \le m} X_i$	Minimum value.
$\arg\min\limits_{0 \le i \le m} X_i$	Iteration where minimum value occurs.
$\overline{X}^{\mathrm{ss}} = \sum_{i=l+1}^{m} X_i/(m-l)$	Steady-state mean.
$S^{\mathrm{ss}} = \sqrt{\dfrac{\sum_{i=l+1}^{m}(X_i - \overline{X}_{\mathrm{ss}})^2}{m-l-1}}$	Steady-state sample standard deviation.

Table 5. Statistical summaries for each output X, where X_i is the value of X at iteration i, m denotes the last iteration, and l corresponds to the iteration separating the transient and steady-state stages.

Rep.	Iterations					Focal measures					
1	X_{10}	X_{11}	$\ldots$	$X_{1,m-1}$	$X_{1,m}$	$\max X_1$	$\arg\max X_1$	$\min X_1$	$\arg\min X_1$	$\overline{X}_1^{\mathrm{ss}}$	S_1^{ss}
2	X_{20}	X_{21}	$\ldots$	$X_{2,m-1}$	$X_{2,m}$	$\max X_2$	$\arg\max X_2$	$\min X_2$	$\arg\min X_2$	$\overline{X}_2^{\mathrm{ss}}$	S_2^{ss}
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	X_{n0}	X_{n1}	$\ldots$	$X_{n,m-1}$	$X_{n,m}$	$\max X_n$	$\arg\max X_n$	$\min X_n$	$\arg\min X_n$	$\overline{X}_n^{\mathrm{ss}}$	S_n^{ss}

Table 6. Values of a generic simulation output (under 'Iterations') for n replications of m iterations each (plus iteration 0, i.e. the initial state), and the respective f.m.'s (under 'Focal measures'). Values along columns are i.i.d..

all runs with the same initial conditions, selected such that: a) it systematically occurs after the transient state; and, b) it is associated with a round and clear value, which is easier to communicate (Sanchez, 1999). Law (2015) suggests the use of Welch's procedure (Welch, 1981) in order to empirically determine l. Let $\overline{X}_0, \overline{X}_1, \overline{X}_2, \ldots, \overline{X}_m$ be the averaged process taken column-wise from Table 6 (columns under 'Iterations'), such that $\overline{X}_i = \sum_{j=1}^{n} X_{ji}/n$ for $i = 0, 1, \ldots, m$. The averaged process has the same transient mean curve as the original process, but its variance is reduced by a factor of n. A low-pass filter can be used to remove short-term fluctuations, leaving the long-term trend of interest, allowing us to visually determine a value of l for which the averaged process seems to have converged. A moving average approach can be used for filtering:

$$\overline{X}_i(w) = \begin{cases} \dfrac{\sum_{s=-w}^{w} \overline{X}_{i+s}}{2w+1} & \text{if } i = w+1, \ldots, m-w \\[2ex] \dfrac{\sum_{s=-(i-1)}^{i-1} \overline{X}_{i+s}}{2i-1} & \text{if } i = 1, \ldots, w \end{cases} \tag{5}$$

where w, the *window*, is a positive integer such that $w \le \lfloor m/4 \rfloor$. This value should be large enough such that the plot of $\overline{X}_i(w)$ is moderately smooth, but not any larger. A more in-depth discussion of this procedure is available in (Welch, 1981; Law, 2015).

Statistical analysis of focal measures

Let $Y_1, Y_2, \ldots, Y_n$ be i.i.d. observations of some f.m. with finite population mean μ and finite population variance σ^2 (i.e. any column under 'Focal measures' in Table 6). Then, as described by Law (2007, 2015), unbiased point estimators for μ and σ^2 are given by

$$\overline{Y}(n) = \frac{\sum\limits_{j=1}^{n} Y_j}{n} \tag{6}$$

10/20

306 and

$$S^2(n) = \frac{\sum\limits_{j=1}^{n} [Y_j - \overline{Y}(n)]^2}{n-1} \tag{7}$$

307 respectively.

308 Another common statistic usually determined for a given f.m. is the confidence interval (c.i.) for $\overline{Y}(n)$,
309 which can be defined in several different ways. The t-distribution c.i. is commonly used for this purpose
310 (Law, 2007, 2015), although it has best coverage for normally distributed samples, which is often not the
311 case for simulation models in general (Sargent, 1976; Law, 2015) and agent-based models in particular
312 (Helbing and Balietti, 2012). If samples are drawn from populations with multimodal, discrete or strongly
313 skewed distributions, the usefulness of t-distribution c.i.'s is further reduced. While there is not much to
314 do in the case of multimodal distributions, Law (2015) proposes the use of the c.i. developed by Willink
315 (2005), which takes distribution skewness into account. Furthermore, c.i.'s for discrete distributions
316 are less studied and usually assume data follows a binomial distribution, presenting some issues of its
317 own (Brown et al., 2001). As suggested by Radax and Rengs (2010), we focus on providing a detailed
318 assessment of the distributional properties of the different f.m.'s, namely whether they are sufficiently
319 "normal" such that normality-assuming (parametric) statistical techniques can be applied, not only for c.i.
320 estimation, but also for model comparison purposes.

321 The normality of a data set can be assessed graphically or numerically (Park, 2008). The former
322 approach is intuitive, lending itself to empirical interpretation by providing a way to visualize how random
323 variables are distributed. The latter approach is a more objective and quantitative form of assessing
324 normality, providing summary statistics and/or statistics tests of normality. In both approaches, specific
325 methods can be either descriptive or theory-driven, as shown in Table 7.

	Graphical methods	Numerical methods
Descriptive	**Histogram**, Box plot, Dot plot	**Skewness**, Kurtosis
Theory-driven	**Q-Q plot**, P-P plot	**Shapiro-Wilk**, Anderson-Darling, Cramer-von Mises, Kolmogorov-Smirnov, Jarque-Bera and other tests

Table 7. Methods for assessing the normality of a data set, adapted from Park (2008). Boldface methods are used in this study.

326 For this study we chose one method of each type, as shown in boldface in Table 7. This approach not
327 only provides a broad overview of the distribution under study, but is also important because no single
328 method can provide a complete picture of the distribution.

329 Under the graphical methods umbrella, a **histogram** shows the approximate distribution of a data set,
330 and is built by dividing the range of values into a sequence of intervals (*bins*), and counting how many
331 values fall in each interval. A **Q-Q plot** compares the distribution of a data set with a specific theoretical
332 distribution (e.g., the normal distribution) by plotting their quantiles against each other (thus "Q-Q").
333 If the two distributions match, the points on the plot will approximately lie on the $y = x$ line. While a
334 histogram gives an approximate idea of the overall distribution, the Q-Q plot is more adequate to see how
335 well a theoretical distribution fits the data set.

336 Concerning numerical methods, **Skewness** measures the degree of symmetry of a probability distribu-
337 tion about its mean, and is a commonly used metric in the analysis of simulation output data (Sargent,
338 1976; Nakayama, 2008; Law, 2015). If skewness is positive, the distribution is skewed to the right, and if
339 negative, the distribution is skewed to the left. Symmetric distributions have zero skewness, however, the
340 converse is not necessarily true, e.g. skewness will also be zero if both tails of an asymmetric distribution
341 account for half the total area underneath the probability density function. In the case of theory-driven
342 numerical approaches, we select the **Shapiro-Wilk** (SW) test (Shapiro and Wilk, 1965), as it has been
343 shown to be more effective when compared to several other normality tests (Razali and Wah, 2011). We
344 focus on the p-value of this test (instead of the test's own W statistic), as it is an easily interpretable

11/20

345 measure. The null-hypothesis of this test is that the data set, or sample, was obtained from a normally
346 distributed population. If the p-value is greater than a predetermined significance level α, usually 0.01 or
347 0.05, then the null hypothesis cannot be rejected. Conversely, a p-value less than α implies the rejection
348 of the null hypothesis, i.e. that the sample was not obtained from a normally distributed population.

RESULTS

350 A total of 30 replications, $r = 1, \ldots, 30$, were performed with NetLogo 5.1.0 for each combination of
351 model sizes (Table 3) and parameters sets (Table 4). Each replication r was performed with a PRNG seed
352 obtained by taking the MD5 checksum of r and converting the resulting hexadecimal string to a 32-bit
353 integer (the maximum precision accepted by NetLogo), guaranteeing some independence between seeds,
354 and consequently, between replications. The list of seeds is provided in Table S1.

Determining the steady-state truncation point

356 Using Welch's method, we smoothed the averaged outputs using a moving average filter with $w = 10$.
357 Having experimented with other values, $w = 10$ seemed to be a good compromise between rough and
358 overly smooth plots. Figure 3 shows results for model size 400 and both parameter sets. Following the
359 recommendations described in section 'Methodology', we select the steady-state truncation point to be
360 $l = 1000$ for parameter set 1, and $l = 2000$ for parameter set 2. These are round values which appear to
361 occur after the transient stage. Other model sizes produce similar results, apart from a vertical scaling
362 factor, which means that these values of l are also applicable in those cases.

Analyzing the distributions of focal measures

364 The six statistic summaries for each f.m., namely mean, sample variance, p-value of the SW test, skewness,
365 histogram and Q-Q plot, are shown in Tables S2.1 to S2.10 (available as supplemental information) for all
366 model size and parameter set combinations. The number of bins in the histograms is set to the minimum
367 between 10 (an appropriate value for a sample size of 30) and the number of unique values in the data set.
368   Much of the information provided in Tables S2.1 to S2.10, namely the p-value of the SW test, the
369 skewness, and the Q-Q plots, is geared towards continuous distributions. However, f.m.'s taken from *arg*
370 *max* and *arg min* operators only yield integer (discrete) values, which correspond to specific iterations. The
371 same is true for *max* and *min* of population outputs, namely P_i^s, P_i^w, and P_i^c. This can be problematic for
372 statistic summaries taken from integer-valued f.m.'s with a small number of unique values. For example,
373 the SW test will not be very informative in such cases, and cannot even be performed if all observations
374 yield the same value (e.g. *arg max* of P_i^c for 800@1, Table S2.4). Nonetheless, distributional properties of
375 a f.m. can dramatically change for different model size and parameter set combinations. For example,
376 for parameter set 2, observations of the *arg max* of P_i^c span many different values for model size 200
377 (Table S2.7), while for size 1600 (Table S2.10) they are limited to only three different values. Summary
378 statistics appropriate for continuous distributions could be used in the former case, but do not provide
379 overly useful information in the latter. In order to maintain a consistent approach, our discussion will
380 continue mainly from a continuous distribution perspective, more specifically by analyzing how closely
381 a given f.m. follows the normal distribution, though we superficially examine its discrete nature when
382 relevant.

Distribution of focal measures over the several size@set combinations

384 In the next paragraphs we describe the distributional behavior of each f.m., and when useful, repeat in a
385 compact fashion some of the information provided in Tables S2.1 to S2.10.

386 $\max P_i^s$: The SW p-value is consistently above the 5% significance level, skewness is usually low and
387 with an undefined trend, and the Q-Q plots are mostly follow the $y = x$ line. Although there are borderline
388 cases, such as 800@1 and 1600@2, the summary statistics show that the maximum prey population f.m.
389 generally follows an approximately normal distribution.

390 $\arg\max P_i^s$: This f.m. follows an approximately normal distribution for smaller sizes of parameter set 1,
391 but as model size grows larger, the discrete nature of the data clearly stands out. This behavior is more
392 pronounced for parameter set 2 (which yields simulations inherently larger than parameter set 1), such
393 that, for 1600@2, all observations yield the same value (i.e. 70). Table 8 shows, using histograms, how
394 the distribution qualitatively evolves over the several size@set combinations.
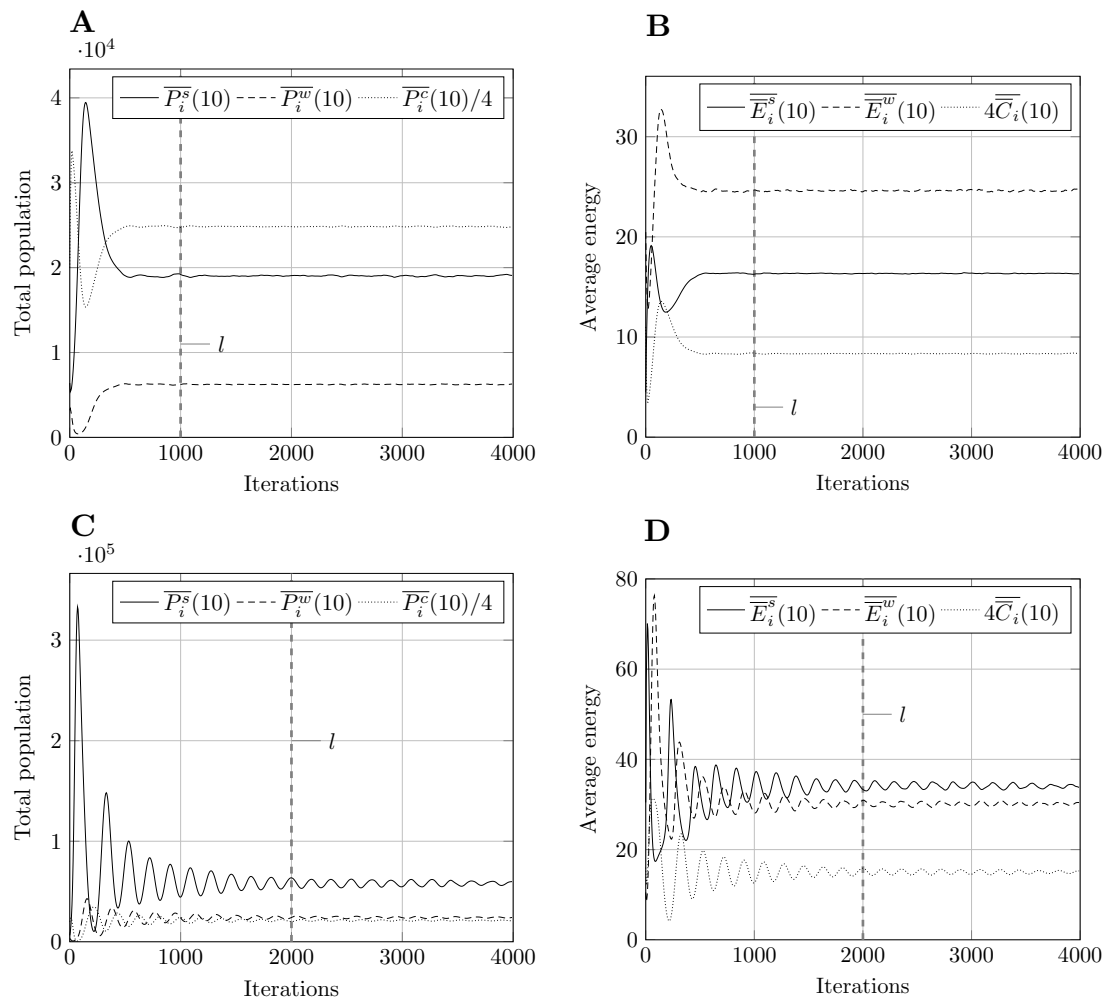
Figure 3. Moving average of outputs for model size 400 with $w = 10$. Other model sizes produce similar results, apart from a vertical scaling factor. The dashed vertical line corresponds to iteration l after which the output is considered to be in steady-state. (A) Population moving average, param. set 1. (B) Energy moving average, param. set 1. (C) Population moving average, param. set 2. (D) Energy moving average, param. set 2.

$\min P_i^s$: Two very different behaviors are observed for the two parameter sets. In the case of parameter set 1, this f.m. has a slightly negatively skewed distribution, with some p-values below the 0.05 significance threshold, but is otherwise not very far from normality (this is quite visible in some histograms). However, for parameter set 2, the data is more concentrated on a single value, more so for larger sizes. Note that this single value is the initial number of prey, which means that, in most cases, the minimum number of prey never drops below its initial value.

$\arg\min P_i^s$: This f.m. follows a similar pattern to the previous one, but more pronounced in terms of discreteness, namely for parameter set 1. For parameter set 2, sizes 100 and 200, the distribution is bimodal, with the minimum prey population occurring at iteration zero (i.e. initial state) or around iteration 200, while for larger sizes, the minimum always occurs at iteration zero.

$\overline{P_i^s}^{ss}$: The prey population steady-state mean seems to generally follow a normal distribution, the only exception being 400@2, in which some departure from normality is observed, as denoted by a SW p-value below 0.05 and a few outliers in the Q-Q plot.
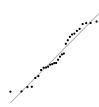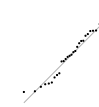
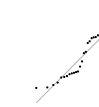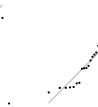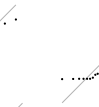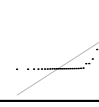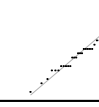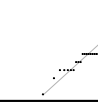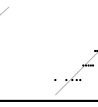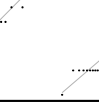$S^{ss}(P_i^s)$: For most size@set combinations this f.m. does not present large departures from normality. However, skewness is always positive.

13/20

Table 8. Histograms for the several size@set combinations of the $\arg\max P_i^s$ f.m..

$\max P_i^w$: This f.m. presents distributions which are either considerably skewed or relatively normal. The former tend to occur for smaller model sizes, while the latter for larger sizes, although this trend is not totally clear. The 800@2 sample is a notable case, as it closely follows a normal distribution, with a symmetric histogram, approximately linear Q-Q plot, and a SW p-value of 0.987.

$\arg\max P_i^w$: Interestingly, for parameter set 1, this f.m. seems to follow a uniform distribution. This is more or less visible in the histograms, but also in the Q-Q plots, because when we plot uniform data points against a theoretical normal distribution in a Q-Q plot we get the "stretched-S" pattern which is visible in this case (Table 9). For parameter set 2, the distribution seems to be more normal, or even binomial as the discreteness of the data starts to stand-out for larger model sizes; the only exception is for size 100, which presents a multimodal distribution.

Table 9. Q-Q plots for the several size@set combinations of the $\arg\max P_i^w$ f.m..

$\min P_i^w$: The minimum predator population seems to follow an approximately normal distribution, albeit with a slight positive skewness, except for 800@1, which has negative skewness.

$\arg\min P_i^w$: This f.m. displays an approximately normal distribution. However, for larger simulations (i.e. mainly for parameter set 2) the discrete nature of the data becomes more apparent.

$\overline{P_i^{w\text{ss}}}$: The steady-state mean of predator population apparently follows a normal distributions. This is confirmed by all summary statistics, such as the SW p-value, which is above 0.05 for all size@set combinations.

$S^{\text{ss}}(P_i^w)$: Departure from normality is not large in most cases (200@2 and 800@2 are exceptions, although the former due to a single outlier), but the trend of positive skewness is again observed for this statistic.
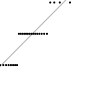
$\max P_i^c$: The maximum available cell-bound food seems to have a normal distribution, although 400@2 has a few outliers which affect the result of the SW p-value (which, nonetheless, is above 0.05).

$\arg\max P_i^c$: The behavior of this f.m. is again quite different between parameter sets. For the first parameter set, the discrete nature of the underlying distribution stands out, with no more than three u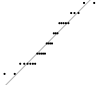nique values for size 100, down to a single value for larger sizes, always centered around the value 12 (i.e. the maximum available cell-bound food tends to occur at iteration 12). For the second parameter set, distribution is almost normal for sizes above 200, centered around iteration 218, although its discreteness shows for larger sizes, namely for size 1600, which only presents three distinct values. For size 100, most values fall in iteration 346, although two outliers push the mean up to 369.5.

439 $\min P_i^c$: This f.m. displays an apparently normal distribution for all model sizes and parameter sets, with
440 the exception of 800@1, which has a few outliers at both tails of the distribution, bringing down the SW
441 p-value barely above the 5% significance level.

442 $\arg\min P_i^c$: In this case, the trend is similar for both parameter sets, i.e. the distribution seems almost
443 normal, but for larger sizes the underlying discreteness becomes apparent. This is quite clear for parameter
444 set 2, as shown in Table 10, where the SW test p-value decreases as the discreteness becomes more visible
445 in the histograms and Q-Q plots .

Size Stat.	100	200	400	800	1600
SW	0.437	0.071	0.062	0.011	< 0.001
Hist.					
Q-Q					

Table 10. Three statistical summaries for the several sizes of the $\arg\min P_i^c$ f.m. for parameter set 2. Row 'SW' contains the SW test p-values, while the corresponding histograms and Q-Q plots are in rows 'Hist.' and 'Q-Q', respectively.

446 $\overline{P_i^c}^{ss}$: For this f.m. there is not a significant departure from normality. The only exception is for 800@1,
447 but only due to a single outlier.

448 $S^{ss}(P_i^c)$: Like in previous cases, the steady-state sample standard deviation does not stray too far from
449 normality, but consistently shows a positive skewness.

450 $\max\overline{E_i^s}$: For sizes 100 and 200 of both parameter sets, the maximum of the mean prey energy presents a
451 positively skewed, lognormal-like distribution. For larger sizes, distributions tend to be more normal-like.
452 This trend is clear when analyzing how the p-value of the SW test and the skewness vary for the several
453 size@set combinations, as shown in Table 11, namely for sizes 100 and 200, where the former is smaller
454 while the absolute value of the latter is larger.

Set	Stat.	Size				
		100	200	400	800	1600
1	SW	0.159	0.012	0.625	0.672	0.555
	Skew.	0.679	0.961	0.521	-0.123	0.196
2	SW	< 0.001	0.038	0.515	0.702	0.337
	Skew.	1.80	1.07	-0.327	-0.216	0.389

Table 11. p-values for the SW test (row 'SW') and skewness (row 'Skew.') for the several size@set combinations of the $\max\overline{E_i^s}$ f.m..

455 $\arg\max\overline{E_i^s}$: For parameter set 1, the distribution is approximately normal for smaller sizes, with the
456 underlying discreteness becoming apparent for larger sizes, centering around iteration 49. For parameter
457 set 2, the data set revolves around a limited set of unique values (centered at iteration 16), following a
458 poisson-like distribution, except for size 100, which displays a bimodal behavior.

459 $\min\overline{E_i^s}$: This f.m. seems to follow an approximately normal distribution.

460 $\arg\min\overline{E_i^s}$ In the case of parameter set 1, this f.m. has distributions with a single value: zero. This
461 means that the minimum mean prey energy occurs at the initial state of the simulation. From there
462 onwards, mean prey energy is always higher. The situation is notably different for the second parameter
463 set, where minimum mean prey energy can occur at several different iterations centered around iteration
464 88. Distribution seems to be binomial or Poisson-like.

$\overline{\overline{E}}_i^{s\,\text{ss}}$: Although the histograms are not very clear, the Q-Q plots and the p-values from the SW test suggest that this f.m. follows a normal distribution.

$S^{\text{ss}}(\overline{E}_i^s)$: This f.m. does not seem to stray much from normality, except in the case of 1600@1 and 200@2, which are affected by outliers. The tendency for the steady-state sample standard deviation statistic to show positive skewness is again confirmed with these observations (800@1 being the exception).

$\max \overline{E}_i^w$: The maximum of mean predator energy follows an approximately normal distribution, though for 100@1 there are a few replications which produce unexpected results.

$\arg\max \overline{E}_i^w$: In most cases, this f.m. approximately follows a normal distribution. There are several exceptions though. For the second parameter set and sizes above 400, the f.m. starts to display its discrete behavior, following a Poisson-like distribution. Less critically, an outlier "ruins" normality for 100@1.

$\min \overline{E}_i^w$: Apart from a few outliers with some parameter combinations, this f.m. generally seems to follow a normal distribution.

$\arg\min \overline{E}_i^w$: Perhaps with the exception of 100@1 and 200@1, the iteration where the minimum of mean predator energy occurs seems best described with a discrete, Poisson-like distribution.

$\overline{\overline{E}}_i^{w\,\text{ss}}$: This f.m. generally follows a normal distribution. However, 1600@1 shows a salient second peak (to the right of the histogram, also visible in the Q-Q plot), affecting the resulting SW p-value, which is below the 1% significance threshold.

$S^{\text{ss}}(\overline{E}_i^w)$: This f.m. follows a positively skewed unimodal distribution, in the same line as the steady-state sample standard deviation of other outputs. Note the outlier in 200@2, also observed for the $S^{\text{ss}}(P_i^w)$ f.m., which is to be excepted as both f.m.'s are related to predator dynamics.

$\max \overline{C}_i$: The samples representing the maximum of the mean C state variable are most likely drawn from a normal distribution. Most histograms are fairly symmetric (which is corroborated by the low skewness values), the Q-Q plots are generally linear, and the SW p-value never drops below 0.05 significance.

$\arg\max \overline{C}_i$: For smaller model sizes this f.m. follows a mostly normal distribution, but as with other iteration-based f.m.'s, the underlying discreteness of the distribution starts to show at larger model sizes, especially for the second parameter set.

$\min \overline{C}_i$: For most size@set combinations, the minimum of the mean C state variable seems to be normally distributed. Nonetheless, a number of observations for 400@2 yield unexpected values, making the respective distribution bimodal and distorting its normality (though the respective SW p-value does not drop below 0.05).

$\arg\min \overline{C}_i$: Like in some previous cases, this f.m. displays different behavior depending on the parameter set. For the first parameter set, practically all observations have the same value, 10, which means the minimum of the mean C state variable is obtained at iteration 10. Only model sizes 100 and 200 have some observations representing iterations 11 and/or 12. Parameter set 2 yields a different dynamic, with an average iteration of 216 approximately (except for size 100, which has an average iteration of 373.3 due to a few very distant outliers). While sizes 200 and 400 follow an approximately normal distribution, larger sizes seem to be more fit to be analyzed using discrete distributions such as Poisson or binomial.

$\overline{\overline{C}}_i^{\text{ss}}$: This f.m. follows an approximately normal distribution. While most size/parameter combinations have a few outliers, only for 800@1 is the existing outlier capable of making the SW test produce a p-value below the 5% significance threshold.

$S^{\text{ss}}(\overline{C}_i)$: Although passing the SW normality test (p-value > 0.05) in most cases, we again note the positive skewness of the steady-state sample standard deviation samples, suggesting that distributions such as Weibull or Lognormal maybe a better fit.

Statistics-wise distribution trends

Table 12 summarizes the descriptions given in the previous section. It was built by assigning an empirical classification from 0 to 5 to each f.m. according to how close it follows the normal distribution for the tested size@set combinations. More specifically, individual classifications were determined by analyzing

the information provided in Tables S2.1 to S2.10, prioritizing the SW test result (i.e. if the *p*-value is above 0.01 and/or 0.05) and distributional discreteness (observable in the Q-Q plots). This classification can be used as a guide to whether parametric or non-parametric statistical methods should be used to further analyze the f.m.'s or to compare f.m.'s of different PPHPC implementations. The last row shows the average classification of individual outputs for a given statistic, outlining its overall normality.

X_i Stat.	$\max\limits_{0\leq i\leq m} X_i$	$\arg\max\limits_{0\leq i\leq m} X_i$	$\min\limits_{0\leq i\leq m} X_i$	$\arg\min\limits_{0\leq i\leq m} X_i$	$\overline{X}^{\text{ss}}$	S^{ss}
P_i^s	●●●●●	●●○○○	●●◐○○	○○○○○	●●●●●	●●●●◐
P_i^w	●●●●○	●◐○○○	●●●●●	●●●○○	●●●●●	●●●●◐
P_i^c	●●●●●	◐○○○○	●●●●●	●●●○○	●●●●●	●●●●◐
$\overline{E}_i^s$	●●●●○	●○○○○	●●●●●	◐○○○○	●●●●●	●●●●◐
$\overline{E}_i^w$	●●●●●	●●●○○	●●●●●	◐○○○○	●●●●●	●●●●○
$\overline{C}_i$	●●●●●	●●◐○○	●●●●●	○○○○○	●●●●●	●●●●◐
Stat. wise	●●●●◐	●●○○○	●●●●◐	●○○○○	●●●●●	●●●●◐

Table 12. Empirical classification (from 0 to 5) of each f.m. according to how close it follows the normal distribution for the tested size@set combinations. The last row outlines the overall normality of each statistic.

The *max* and *min* statistics yield mostly normal distributions, although care should be taken when the maximum or minimum systematically converge to the same value, e.g. when they occur at iteration zero. Nonetheless, parametric methods seem adequate for f.m.'s drawn from these statistics. The same does not apply to the *arg max* and *arg min* statistics, which show a large variety of distributional behaviors (including normality in some cases). Thus, these statistics are better handled with non-parametric techniques. The steady-state mean typically displays distributions very close to normal, probably due to central-limit-theorem type effects, as described by Law (2007) for mean or average-based f.m.'s. Consequently, parametric methods will most likely be suitable for this statistic. Finally, f.m.'s based on the steady-state sample standard deviation display normal-like behavior, albeit with consistently positive skewness; in fact, they are probably better represented by a Weibull or Lognormal distribution. While parametric methods may be used for this statistic, results should be interpreted cautiously.

DISCUSSION

In this paper, the PPHPC model is completely specified, and an exhaustive analysis of the respective simulation outputs is performed. Regarding the latter, after determining the mean and variance of the several f.m.'s, we opted to study their distributional properties instead of proceeding with the classical analysis suggested by simulation output analysis literature (i.e. the establishment of c.i.'s.). This approach has a number of practical uses. For example, if we were to estimate c.i.'s for f.m.'s drawn from the steady-state mean, we could use *t*-distribution c.i.'s with some confidence, as these f.m.'s display an approximately normal distribution. If we did the same for f.m.'s drawn from the steady-state sample standard deviation, the Willink (2005) c.i. would be preferable, as it accounts for the skewness displayed by these f.m.'s. Estimating c.i.'s without a good understanding of the underlying distribution can be misleading, especially if the distribution is multimodal. The approach taken here is also useful for comparing different PPHPC implementations. If we were to compare *max* or *min*-based f.m.'s, which seem to follow approximately normal distributions, parametric tests such as the *t*-test would most likely produce valid conclusions. On the other hand, if we compare *arg max* or *arg min*-based f.m.'s, non-parametric tests, such as the Mann-Whitney U test (Gibbons and Chakraborti, 2011), would be more adequate, as these f.m.'s do not usually follow a normal distribution.

However, the scope of the PPHPC model is significantly broader. For example, in (Fachada et al., 2015b), PPHPC is reimplemented in Java with several user-selectable parallelization strategies. The goal is to clarify which are the best parallelization approaches for SABMs in general. A *n*-sample statistical test is applied to each f.m., for all implementations and strategies simultaneously, in order to verify that these do not yield dissimilar results. In (Fachada et al., 2015a), PPHPC is used for presenting a novel

17/20

549 model-independent comparison technique which directly uses simulation outputs, bypassing the need of
550 selecting model-specific f.m.'s.
551   The PPHPC model is made available to other researchers via the source code, in addition to the
552 specification presented here. All the data analyzed in this paper is also available as supplemental data.
553 PPHPC can be used as a pure computational model without worrying with aspects like visualization and
554 user interfaces, allowing for direct performance comparison of different implementations.

CONCLUSION

556 In this paper, we presented PPHPC, a conceptual model which captures important characteristics of
557 SABMs. The model was comprehensively described using the ODD protocol, a NetLogo canonical
558 implementation was reported, and simulation outputs were thoroughly studied from a statistical perspective
559 for two parameter sets and several model sizes. While many ABMs have been published, proper model
560 description and analysis is lacking in the scientific literature, and thus this paper can be seen as a guideline
561 or methodology to improve model specification and communication in the field. Furthermore, PPHPC
562 aims to be a standard model for research in agent-based modeling and simulation, such as, but not limited
563 to, statistical model comparison techniques, performance comparison of parallel implementations, and
564 testing the influence of different PRNGs on the statistical accuracy of simulation output.

REFERENCES

566 Axtell, R., Axelrod, R., Epstein, J. M., and Cohen, M. D. (1996). Aligning simulation models: a case
567   study and results. *Comput. Math. Organ. Theory*, 1(2):123–141.
568 Bigbee, A., Cioffi-Revilla, C., and Luke, S. (2007). Replication of Sugarscape using MASON. In Terano,
569   T., Kita, H., Deguchi, H., and Kijima, K., editors, *Agent-Based Approaches in Economic and Social*
570   *Complex Systems IV*, volume 3 of *Springer Series on Agent Based Social Systems*, pages 183–190.
571   Springer Japan.
572 Brown, L. D., Cai, T. T., and DasGupta, A. (2001). Interval estimation for a binomial proportion. *Stat.*
573   *Sci.*, 16(2):101–117.
574 D'Souza, R., Lysenko, M., and Rahmani, K. (2007). Sugarscape on steroids: simulating over a million
575   agents at interactive rates. In *Proc. of Agent 2007 Conf.*, Chicago, USA.
576 Edmonds, B. and Hales, D. (2003). Replication, replication and replication: Some hard lessons from
577   model alignment. *J. Artif. Soc. Soc. Simulat.*, 6(4).
578 Epstein, J. and Axtell, R. (1996). *Growing artificial societies: social science from the bottom up*. MIT
579   press.
580 Fachada, N. (2008). Agent-based simulation of the immune system. Master's thesis, Instituto Superior
581   Técnico, Universidade Técnica de Lisboa, Lisboa.
582 Fachada, N., Lopes, V. V., Martins, R. C., and Rosa, A. C. (2015a). Model-independent comparison of
583   simulation output. *arXiv*, 1509.09174. [cs.OH].
584 Fachada, N., Lopes, V. V., Martins, R. C., and Rosa, A. C. (2015b). Parallelization strategies for spatial
585   agent-based models. *arXiv*, 1507.04047. [cs.DC].
586 Gibbons, J. D. and Chakraborti, S. (2011). *Nonparametric statistical inference*. Springer.
587 Ginovart, M. (2014). Discovering the power of individual-based modelling in teaching and learning: The
588   study of a predator–prey system. *J. Sci. Educ. Technol.*, 23(4):496–513.
589 Goldsby, M. E. and Pancerella, C. M. (2013). Multithreaded agent-based simulation. In *Proceedings of*
590   *the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World*, WSC '13,
591   pages 1581–1591, Washington, D.C., USA. IEEE Press.
592 Grimm, V., Berger, U., DeAngelis, D., Polhill, J., Giske, J., and Railsback, S. (2010). The ODD protocol:
593   A review and first update. *Ecol. Model.*, 221(23):2760–2768.
594 Helbing, D. and Balietti, S. (2012). How to do agent-based simulations in the future: from modeling
595   social mechanisms to emergent phenomena and interactive systems design. In *Social Self-Organization*,
596   chapter Agent-Based Modeling, pages 25–70. Springer.
597 Hiebeler, D. (1994). The Swarm simulation system and individual-based modeling. SFI Working Paper:
598   1994-11-065.
599 Isaac, A. G. (2011). The ABM template models: A reformulation with reference implementations. *J.*
600   *Artif. Soc. Soc. Simulat.*, 14(2):5.

Kahn, K. (2007). Comparing multi-agent models composed from micro-behaviours. In Rouchier, J., Cioffi-Revilla, C., Polhill, G., and Takadama, K., editors, *M2M 2007: Third International Model-to-Model Workshop*, Marseilles, France.

Kelton, W. D. (1997). Statistical analysis of simulation output. In *Proceedings of the 29th Conference on Winter Simulation*, WSC '97, pages 23–30, Washington, DC, USA. IEEE Computer Society.

Law, A. M. (1983). Statistical analysis of simulation output data. *Oper. Res.*, 31(6):983–1029.

Law, A. M. (2007). Statistical analysis of simulation output data: the practical state of the art. In *Simulation Conference, 2007 Winter*, pages 77–83. IEEE.

Law, A. M. (2015). *Simulation Modeling and Analysis*. McGraw-Hill, fifth edition.

Lotka, A. (1925). *Elements of Physical Biology*. Williams and Wilkins.

Lysenko, M. and D'Souza, R. (2008). A framework for megascale agent based model simulations on graphics processing units. *J. Artif. Soc. Soc. Simulat.*, 11(4):10.

Lytinen, S. L. and Railsback, S. F. (2012). The evolution of agent-based simulation platforms: A review of NetLogo 5.0 and ReLogo. In *Proceedings of the Fourth International Symposium on Agent-Based Modeling and Simulation*.

Macal, C. and North, M. (2010). Tutorial on agent-based modelling and simulation. *J. Simulat.*, 4(3):151–162.

Macal, C. M. and North, M. J. (2008). Agent-based modeling and simulation: ABMS examples. In *Proceedings of the 40th Conference on Winter Simulation*, WSC '08, pages 101–112. Winter Simulation Conference.

Martin, B. T., Zimmer, E. I., Grimm, V., and Jager, T. (2012). Dynamic energy budget theory meets individual-based modelling: a generic and accessible implementation. *Methods Ecol. Evol.*, 3(2):445–449.

Merlone, U., Sonnessa, M., and Terna, P. (2008). Horizontal and vertical multiple implementations in a model of industrial districts. *J. Artif. Soc. Soc. Simulat.*, 11(2):5.

Müller, B., Balbi, S., Buchmann, C. M., De Sousa, L., Dressler, G., Groeneveld, J., Klassert, C. J., Le, Q. B., Millington, J. D., Nolzen, H., et al. (2014). Standardised and transparent model descriptions for agent-based models: Current status and prospects. *Environ. Model. Softw.*, 55:156–163.

Nakayama, M. K. (2008). Statistical analysis of simulation output. In *Proceedings of the 40th Conference on Winter Simulation*, WSC '08, pages 62–72. Winter Simulation Conference.

Ottino-Loffler, J., Rand, W., and Wilensky, U. (2007). Co-evolution of predators and prey in a spatial model. In *GECCO 2007*.

Park, H. M. (2008). Univariate analysis and normality test using SAS, Stata, and SPSS. Working Paper. The University Information Technology Services (UITS) Center for Statistical and Mathematical Computing, Indiana University.

Radax, W. and Rengs, B. (2010). Prospects and pitfalls of statistical testing: Insights from replicating the demographic prisoner's dilemma. *J. Artif. Soc. Soc. Simulat.*, 13(4):1.

Railsback, S., Lytinen, S., and Grimm, V. (2005). *StupidModel and extensions: A template and teaching tool for agent-based modeling platforms*. Swarm Development Group.

Railsback, S., Lytinen, S., and Jackson, S. (2006). Agent-based simulation platforms: Review and development recommendations. *Simulation*, 82(9):609–623.

Razali, N. M. and Wah, Y. B. (2011). Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. *J. Stat. Model. Anal.*, 2(1):21–33.

Reynolds, C. (2006). Big fast crowds on PS3. In *Proceedings of the 2006 ACM SIGGRAPH Symposium on Videogames*, Sandbox '06, pages 113–121, New York, NY, USA. ACM.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34.

Rollins, N. D., Barton, C. M., Bergin, S., Janssen, M. A., and Lee, A. (2014). A computational model library for publishing model documentation and code. *Environ. Model. Softw.*, 61:59–64.

Sallach, D. and Mellarkod, V. (2005). Interpretive agents: A heatbug reference simulation. In *Proc. Agent 2005 Conf. Generative Social Processes, Models, and Mechanisms*, pages 693–705.

Sanchez, S. M. (1999). ABC's of output analysis. In *Simulation Conference Proceedings, 1999 Winter*, volume 1, pages 24–32, Phoenix, AZ, USA. IEEE.

Sargent, R. G. (1976). Statistical analysis of simulation output data. *SIGSIM Simul. Dig.*, 7(4):39–50.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples).

656   *Biometrika*, 52(3/4):591–611.

657   Shook, E., Wang, S., and Tang, W. (2013). A communication-aware framework for parallel spatially
658   explicit agent-based models. *Int. J. Geogr. Inf. Sci.*, 27(11):2160–2181.

659   Smith, M. (1991). Using massively-parallel supercomputers to model stochastic spatial predator-prey
660   systems. *Ecol. Model.*, 58(1):347–367.

661   Sondahl, F., Tisue, S., and Wilensky, U. (2006). Breeding faster turtles: Progress towards a NetLogo
662   compiler. In *Proceedings of the Agent 2006 conference on social agents, Chicago, IL, USA*.

663   Standish, R. K. (2008). Going stupid with EcoLab. *Simulation*, 84(12):611–618.

664   Tang, W. and Wang, S. (2009). HPABM: A hierarchical parallel simulation framework for spatially-explicit
665   agent-based models. *Trans. GIS*, 13(3):315–333.

666   Tatara, E., North, M., Howe, T., Collier, N., Vos, J., and Vos, J. (2006). An introduction to Repast modeling
667   using a simple predator-prey example. In Sallach, D., Macal, C., and M.J., N., editors, *Proceedings of
668   the Agent 2006 Conference on Social Agents: Results and Prospects*, volume ANL/DIS-06-7, pages
669   83–94. co-sponsored by the Argonne National Laboratory and The University of Chicago.

670   Volterra, V. (1926). Fluctuations in the abundance of a species considered mathematically. *Nature*,
671   118:558–560.

672   Welch, P. D. (1981). On the problem of the initial transient in steady-state simulation. IBM Watson
673   Research Center.

674   Wilensky, U. (1997). NetLogo wolf sheep predation model.

675   Wilensky, U. (1999). NetLogo.

676   Wilensky, U. (2004). NetLogo heatbugs model.

677   Wilensky, U. (2014). *NetLogo 5.1.0 User Manual*. Northwestern University, Evanston, IL, USA.

678   Wilensky, U. and Rand, W. (2007). Making models match: replicating an agent-based model. *J. Artif.
679   Soc. Soc. Simulat.*, 10(4):2.

680   Willink, R. (2005). A confidence interval and test for the mean of an asymmetric distribution. *Commun.
681   Stat. Theor. M.*, 34(4):753–766.

1

NetLogo implementation of the PPHPC model.

Figure 1. NetLogo implementation of the PPHPC model.
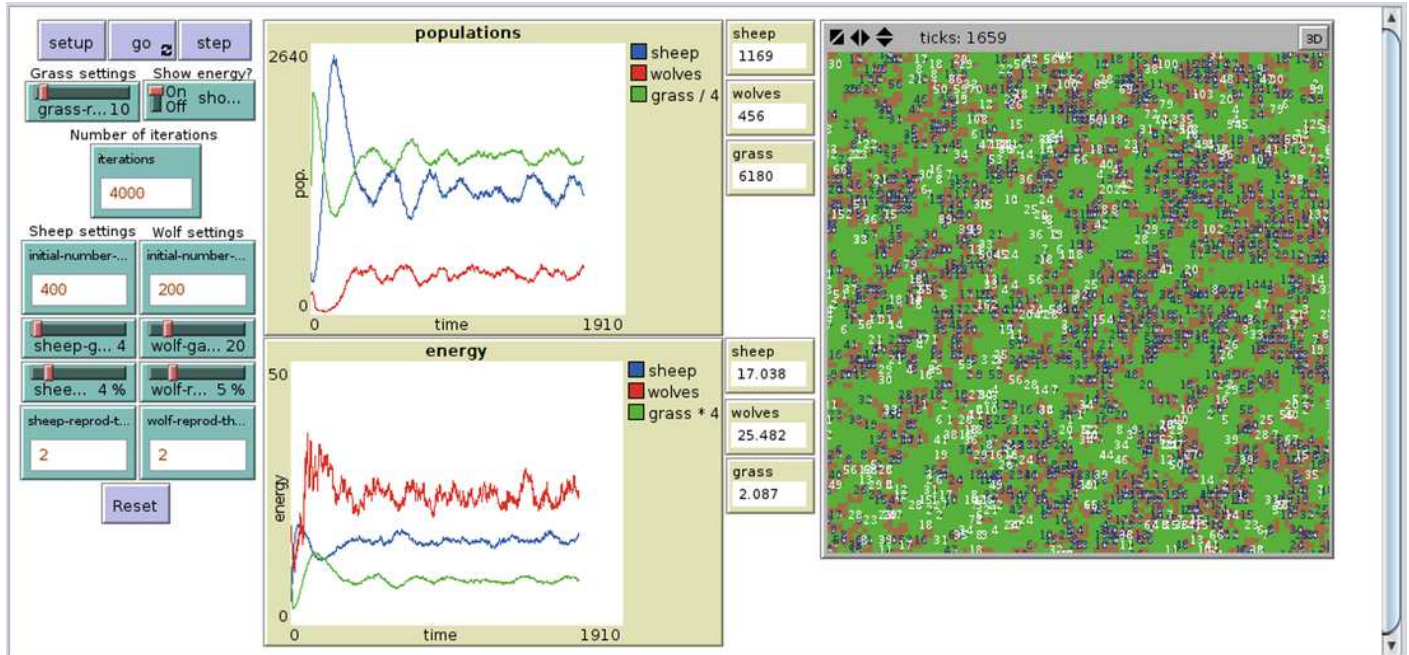
Figure 2(on next page)

Moving average of outputs for model size 400 with $w=10$

Moving average of outputs for model size 400 with $w=10$. Other model sizes produce similar results, apart from a vertical scaling factor. The dashed vertical line corresponds to iteration l after which the output is considered to be in steady-state. (A) Population moving average, param. set 1. (B) Energy moving average, param. set 1. (C) Population moving average, param. set 2. (D) Energy moving average, param. set 2.
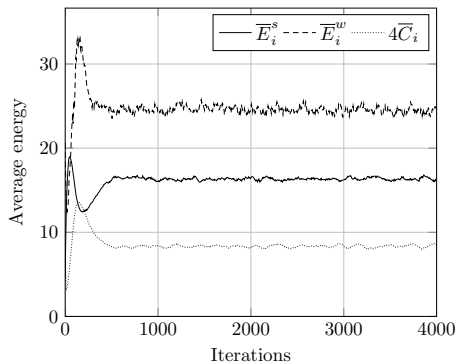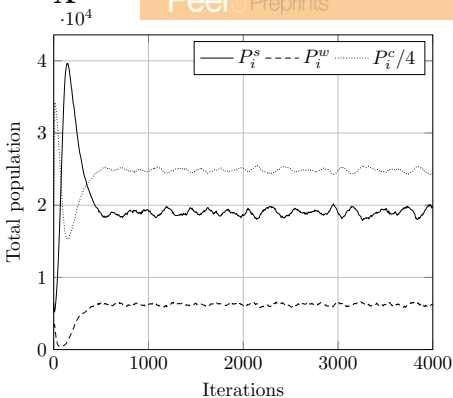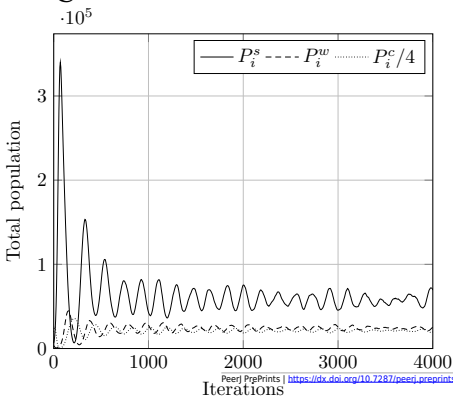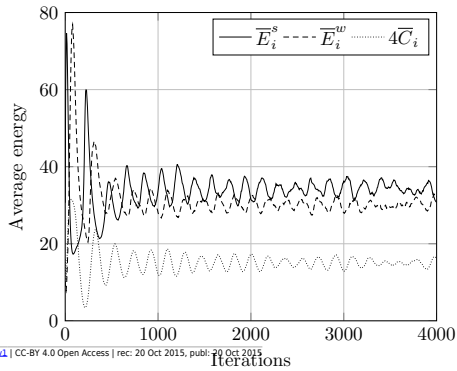
Figure 3(on next page)

Typical model output for model size 400

Figure 2. Typical model output for model size 400. Other model sizes have outputs which are similar, apart from a vertical scaling factor. P_i refers to total population, $\mean{E}_i$ to mean energy and $\mean{C}_i$ to mean value of the countdown state variable, C. Superscript s relates to prey, w to predators, and c to cell-bound food. P_i^c and $\mean{C}_i$ are scaled for presentation purposes. (A) Population, param. set 1. (B) Energy, param. set 1. (C) Population, param. set 2. (D) Energy, param. set 2.