Simulation Modelling Practice and Theory 83 (2018) 201–212

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Lessons learned on development and application of agent-based models of complex dynamical systems

Richard A. Williams [a,b]

[a] Department of Management Science, Lancaster University, Lancaster, UK
[b] Data Science Institute, Lancaster University, Lancaster, UK

ARTICLE INFO

Article history:
Available online 11 November 2017

Keywords:
Agent-based modeling
Model development
Simulation-based experimentation
Complex dynamical systems

ABSTRACT

The field of agent-based modelling (ABM) has gained a significant following in recent years, and it is often marketed as an excellent introduction to modelling for the novice modeller or non-programmer. The typical objective of developing an agent-based model is to either increase our mechanistic understanding of a real-world system, or to predict how the dynamics of the real-world system are likely to be affected by changes to internal or external factors. Although there are some excellent ABMs that have been used in a predictive capacity across a number of domains, we believe that the promotion of ABM as an 'accessible to all' approach, could potentially lead to models being published that are flawed and therefore generate inaccurate predictions of real-world systems. The purpose of this article is to use our experiences in modelling complex dynamical systems, to reinforce the view that agent-based models can be useful for answering questions of the real-world domain through predictive modelling, but also to emphasise that all modellers, expert and novice alike, must make a concerted effort to adopt robust methods and techniques for constructing, validating and analysing their models, if the result is to be meaningful and grounded in the system of interest.

1. Introduction

Through the ever-continuing advancements in hardware and software technologies, the scale and sophistication of modelling tools continues to increase. In particular, the focus on developing agent-based modelling (ABM) tools over the last two decades, has resulted in several standardized libraries emerging (see [1,2] for discussion). These have allowed researchers with subject matter expertise (but not necessarily computer science expertise or progamming experience) to quickly develop models of their real-world systems of interest, and to perform cause-and-effect experiments through computer simulations. The underlying conceptual logic of the ABM approach is relatively simple for a newcomer to grasp, because real-world entities, such as a person or a biological cell, can have a straightforward mapping to a computational analogue known as an agent.

ABMs can provide a powerful means by which to formulate a model that replicates the known characteristics and behaviour(s) of a real-world system, or to explore and experiment on the dynamics and underlying mechanistic behaviour(s) of the system, where individual agents are autonomous and respond to the simulated environment [3]. The behaviour of agents is governed by pre-defined agent rules, and even the simplest agent rules can give rise to complex population level phe-

E-mail address: r.williams4@lancaster.ac.uk

https://doi.org/10.1016/j.simpat.2017.11.001
1569-190X/

nomena. It is the study of the resultant population behaviours with respect to the *heterogeneous and autonomous* individuals that is one of the key advantages of agent-based modelling [4], and has led to its rapid rise in adoption over the past decade [5–7]. For instance, the agent-based approach has been used in a diverse range of disciplines including the modelling of: diseases in biology [8]; financial markets [9] and the British banking sector [10] in economics; the movement of households and emergent segregation of communities within a reimplementation of Shelling's Bounded Neighbourhood Model [11] and violent crime [12] in social science; the effects of communication technologies on virtual project team performance [13] in management science; and seasonal impacts in agriculture [14]. We believe that this has been driven primarily by the ability of agent-based modelling and simulation to investigate questions regarding the causal relationships and mechanistic underpinnings to system dynamics, which traditional modelling techniques, such as differential equation-based, system dynamics (system-based models), and discrete event simulation (process-based models) cannot address [3,16]. This is supported by Epstein [4] who states that *"agent-based models provide computational demonstrations that a given microspecification is in fact sufficient to generate a macrostructure of interest."* In addition, the popularity of an agent-based approach has further risen through the increase in the number of software development frameworks for agent-based modelling and simulation [17].

Computational models can provide an interpretation (be that mechanistic or dynamic) of the underlying real-world system data upon which the model is constructed [18–20]. As per all types of models, agent-based models are simplifications of the real-world systems upon which they are based. In the course of computational model design and development, decisions are made regarding the interpretation of real-world system data and how this should be translated into a form that can be used within a computational model. It is usually intractable (from both a computational and domain knowledge perspective) to represent every aspect of the real-world system, for all but the most simplistic of case studies [21]. In addition, an inherent, but often forgotten aspect of simulation studies is that the definitive mechanisms and values of various spatio-temporal interactions within complex dynamical systems are not currently known. This necessitates the incorporation of assumptions into the model, which adds a layer of abstraction between the computational model and the real-world system under study. It is therefore essential that the effects of these decisions on the simulation results are fully understood, as any assumptions made during the design and development of the computational model, could have critical impact on the simulation response [22,23]. However, as discussed by Alden et al. [24], it is rare that a published model is accompanied by an in-depth description and justification of the abstraction level taken during model design, and assumptions made during model development.

Developing, running and analysing ABMs can be a challenge because even simple ABMs, such as the predator-prey model [25], can generate complex emergent behaviours. We believe that the majority of novice modellers (including domain experts who have begun to use ABMs within their research) are not always aware of the varying aspects of uncertainty that are inherent to simulation results, especially those of stochastic dynamical systems, or indeed the underlying ABM development frameworks. This occurs in the most part, from an absence of training in software development during the years of formal education that researchers in diverse fields such as biology, economics, and the social sciences, undergo at university. This is emphasised by Wooldridge [26] who states that the development of agent-based models is a process of experimentation, but that the experimental process tends to encourage developers to forget that they are actually developing software, and thus the core activities of software engineering (e.g. requirements analysis, specification, design, verification, and testing) are often forgotten. In fact, it has previously been argued that the ABM approach has been deemed inadequate for scientific use due to the perceived lack of engineering rigour [27]. This is in part, due to the weak validation and verification that has been performed on a number of the published models over the past decade, which makes the results and predictions from these models hard to trust [28]. We believe that with increasing experience of developing ABMs of complex dynamical systems, these software engineering principles can be acquired and strengthened over time however.

As highlighted above, many ABM environments allow a user to quickly instantiate models and produce results, however, we believe that this introduces the risk of inexperienced modellers incorrectly interpreting their simulation results just as easily as they construct their models, which supports Bonabeau [5] who argues that *"although ABM is technically simple, it is also conceptually deep."* (see [29] for a primer on modelling and simulation-based systems engineering). This introduces risks into projects that aim to use ABMs in a predictive capacity regarding behaviours and dynamics of a real-world system. We believe that these risks fall into three main categories around: lack of detailed design and specifications; analysis of simulation results; along with calibration and uncertainty analysis. These are expanded below:

No Discernible Specifications: Due to space limitations of journal articles, which do not always allow online supporting information, published models are not always supported by a functional specification or formalised conceptual model of the real-world system under study. As such the reader is sometimes left feeling uncertain whether the ABM has represented the real-world system appropriately. Without a functional specification (conceptual model), it is unclear how validation can be performed, as the specification provides context for judgements on the acceptability of interpolation or extrapolation of simulation results with respect to the real-world system of interest [30]. Similarly, we have also found that technical specifications, which link the conceptual model to the computational model are also sometimes omitted. As there are always constraints (and indeed different abstraction levels taken) regarding how a modeller chooses to implement specific real-world mechanisms, the functional and technical specifications are pivotal in setting the context in which the simulation data should be interpreted. After all, different abstractions and implementations, may produce different results.

Analytics Methods: A large number of ABM development and simulation frameworks do not provide any corresponding analytics tools, and thus require the research team to develop their own. Conversely, some of the more accessible simulation frameworks for non-Computer Scientists provide a number of methods for live analytics within the tools, whereas in reality,

other tools would be better suited to process the simulated data. These latter tools incur the risk of individuals trusting in the immediacy of the data output from their simulation, and may lead to false conclusions being made from single run experiments. Non-ABM specific tools such as the statistical analysis software Matlab or R provide a more comprehensive, purpose built utility for data processing, but leave the modeller to develop the analytic scripts themselves.

Calibration and Uncertainty Analysis: Calibration is an important process when developing agent-based models of complex dynamical systems. Like other modelling paradigms, agent-based modellers need to calibrate the *baseline* (normal state) behaviour for their simulation, in order to align the dynamics and (specifically for ABM) emergent behaviours of the computational model with those of the real-world system under study. This can be a time and resource intensive activity that requires close collaboration between modeller and domain expert, the latter being key to ensure simulation results are well-grounded in the target domain, and provide an acceptable baseline behaviour upon which the results of subsequent simulation-based experimentation can be compared. It is surprising how many published models lack discussion of calibration to real-world system dynamics (at a minimum) or uncertainty analysis of the calibrated behaviours, but then go on to use the results of simulation-based experimentation in a predictive capacity. We conjecture that this may result from the increased use of GUI-driven tools, which in our opinion are not always built to accommodate the kind of exploratory analysis needed to calibrate models of complex dynamical systems with emergent behaviours to known baseline dynamics. Recently, there has been considerable work around the area of calibration and uncertainty analysis of agent-based models: Read et al. [21,31] present an approach for calibrating complex dynamical systems that accounts for the underlying uncertainties inherent to the real-world system and the stochastic agent-based model itself; Lamperti et al. [32] use machine learning surrogates as a way to build meta-models of the agent-based model in order to tackle parameter space exploration; Dosi et al. [33] apply Kriging meta-modelling to explore the parameter space and to develop a linear unbiased model of their complex (high-dimensional and non-linear) simulation model; Barde and van der Hoog [34] present a proof-of-concept empirical validation method, which uses a three step process of Nearly-Orthogonal Latin Hypercube sampling to generate a set of parameter combinations, Markov Information Criterion (MIC) to score the simulated data, and stochastic kriging to develop a surrogate model of the MIC response surface.

This paper will present a number of key lessons learned from our experiences in developing agent-based models of complex dynamical systems around three main case studies: Experimental Autoimmune Encephalomyelitis, an animal model of the human disease Multiple Sclerosis, that utilises complex intercellular interactions [35,36]; the IL-1 stimulated Nuclear Factor-kappa B (NF-κB), which is crucial to normal immune system function and utilises a complex intracellular signalling pathway [15,37]; and the propagation of conflict within the social networks of large software implementation projects [38]. Section two will discuss how we can instill faith in our agent-based models of complex dynamical systems, through the need for a principled approach to design and development, and how adherence to a modelling and simulation development framework (in our case the CoSMoS process) can promote the adherence to good software engineering principles, such as uncertainty analysis, validation and verification, and rigorous statistical analysis of simulation results so that we are able to appropriately interpret results from the computational model in the context of the real-world domain. Section three will provide a discussion of the importance of a principled approach to design and development of agent-based models when they are used to investigate complex dynamical systems, in particular the mechanistic interactions that give rise to the emergent behaviour(s) of the system.

2. How can we instill faith in our agent-based models of complex dynamical systems?

The need for a principled approach to modelling and simulation of complex dynamical systems was the emphasis of a multidisciplinary research programme funded by the Engineering and Physical Sciences Research Council (EPSRC grants EP/E053505/1 and EP/E049419/1, between 2007–2011). This programme was titled the Complex Systems Modelling and Simulation (CoSMoS) Project,[1] and resulted in the CoSMoS process [39]. This process provides a framework of leading practice for developing and using simulations to explore complex systems, and is comparable to project lifecycle methodologies used in industry. The CoSMoS process is organised around phases, which contain a set of products (deliverables), and associated activities. It has three phases: the Discovery phase, which establishes the scientific basis of the project, identifies and models the domain of interest, and formulates scientific questions; the Development phase, which produces the simulator; and the Exploration phase, which uses the simulator for simulation-based experimentation, the results of which are used to explore the scientific questions defined previously. Along with these phases, there are key products associated with CoSMoS projects: Domain Model, Platform Model, Simulation Platform, and Results Model (Fig. 1).

This section will discuss a number of lessons learned from our experiences with agent-based modelling of biological and social system case studies. The section is structured around the phases of the CoSMoS process, which begins with the discovery phase where we investigate the real-world domain and develop a domain model, before progressing to the development phase, where the ABM is developed and tested, and finally ending with the exploration phase, where we perform simulation-based experimentation to answer questions of the real-world complex system.

[1] www.cosmos-research.org.

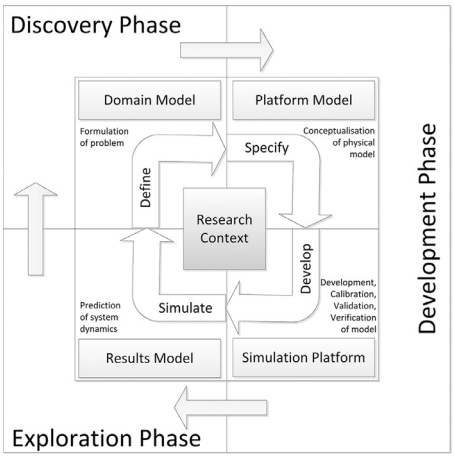

Fig. 1. The CoSMoS process advocates an iterative lifecycle, consisting of three separate phases (discovery, development and exploration), and creation of four key project artefacts (domain model, platform model, simulation platform, and results model). The discovery phase focuses on formulation of the problems to be investigated through use of the computational model, resulting in creation of a functional specification of the required real-world behaviour to be simulated (domain model). The development phase focuses on transforming the domain model into a technical specification (platform model) specific to the programming language(s) and computer architectures to be used, and actual development of the computational model (simulation platform), including calibration, validation and verification. The exploration phase focuses on the simulation-based experimentation to investigate the real-world problems of interest, and the generation of predictions (documented in the results model), which facilitate the generation of novel hypotheses for subsequent testing in the real-world arena. Reproduced under CC BY 4.0 from [40].

2.1. Discovery phase

Before investing significant time and effort in computational model development, we need to ensure that the problem we as modellers believe we need to solve, is actually a recognised and legitimate problem that needs solving in the real-world. The discovery phase of the CoSMoS process focuses on this formulation of the problem from the real-world domain, and the subsequent formal definition of this into a conceptual model, which has been termed a *domain model* in the CoSMoS process. The domain in this context represents the real-world system, or part of the system due to an appropriate abstraction level. Similarly, the research context in the CoSMoS process relates to the overall context and scope of the computational model. The research context is of paramount importance, and can be thought of as the underlying thread of knowledge, scope and assumptions, that runs throughout the entire CoSMoS project. The scope, abstraction level, assumptions and constraints that are presented within the domain model, provide the context behind how simulation results should be validated, interpreted and evaluated [41].

The domain model represents our understanding of the system being studied, and is in effect an abstract representation of the real-world system of interest (the domain), which documents our understanding of the system into explicit statements relating to assumptions, constraints and definitions of the underlying mechanisms and data, or indeed the structure of components and their relationships/interactions within the system [42]. Andrews et al. [43] advise that as the domain model focuses on the real-world system, it should be free of any simulation language or hardware platform bias, as the decisions on appropriate tools and technologies should be left until the full scope of the functional requirements for the model are captured.

One of our recent modelling projects involved a complex dynamical system from biology and focused on the IL-1 stimulated Nuclear Factor-kappa B (NF-κB) signalling pathway. Briefly, NF-κB is a collective term for a family of inducible dimeric transcription factors, and as such is an essential intracellular messenger, which in conjunction with its signalling pathway connects various extracellular stimuli to the induction of gene expression, cell proliferation, cell differentiation, and cell survival. We have recently published our domain model in [40], and learned a number of valuable lessons on the process and modelling techniques used within the modelling and simulation lifecycle. These are defined below:

Lesson 1: Close collaboration between modeller and domain expert is required to ensure a solid foundation of the real-world system. Development of the domain model is an iterative process, where the various views of the system are refined until they meet the abstracted view of the real-world system for the purposes of the simulation project. In our case, this meant the domain model was refined until agreement was gained between the modeller and domain expert.[2] We adopted the *deep curation* approach [44] for each domain model, which used manual curation of relevant facts and

[2] We have found that collaboration between a modeller (software skilled computer scientist) and domain expert provides for a higher quality ABM, although understand that this is not always possible, and that there are times where a field researcher also has the necessary software development skills himself/herself, in particular those that have traversed disciplinary boundaries or have interdisciplinary backgrounds.

information from the published literature, experimental datasets (gained through wet-lab single-cell analysis by experienced Cell Biologists), and interactions with the domain expert, to develop an initial model in a quasi-top-down manner.

Of particular note to an agent-based approach is that the domain model should document our understanding of: the mechanistic behaviours of system components, which will become our agents in the computational model; the heterogenous dynamics of these components (see statistical analysis of NF-κB dynamics in our domain model [40]); and how the interactions between them lead to the emergent behaviour of the system - these spatio-temporal, heterogenous, and mechanistic underpinnings are suited to an agent-based approach, and indeed we conjecture that the ability to model these aspects is the main strength of ABMs over other approaches (e.g. differential equations, discrete event simulations, or process modelling). This iterative approach, with close liaison between modeller and domain expert, ensures that an extended period of time is set aside for comprehensive exploration of the real-world domain before development of the computational model, and furthermore ensures the modeller has sufficient understanding of the domain before design and implementation of the resulting simulation-based experimentation.

Lesson 2: A variety of diagrammatic notations and statistical techniques are required to develop a comprehenisve conceptual model of a complex dynamical system. For instance, the Unified Modelling Language [45] was useful for defining the communication and sequence of interactions between the various components of the NF-κB signalling pathway, the sequence of activities within the system, and the low-level changes in state of components within the system. Unfortunately, it was found to have deficiencies in defining the network structure of the system, the hierarchical containment of individual components, and the overall system's view that links high-level system behaviours to low-level interactions between components, which were all overcome through using informal free-form diagrams (termed *cartoon* diagrams by biologists). In addition, UML diagrams were unable to model the temporal dynamics or to define the key rates, ratios and physical attributes of the system, which required the use of descriptive tables and various statistical techniques [40]. For instance, univariate and multivariate statistical analysis of the data was found to provide a complementary view to the domain model that UML could not capture on its own. Through visualising the data as histograms, it quickly became evident that the data was highly skewed, and that any assumptions to a normal distribution, which requires a symmetrical distribution around the modal value, would be incorrect. Due to the uncertainty about direction of difference, two-tailed χ^2 goodness of fit tests were therefore used to identify the type of distribution that the data approximated to, which in our case was a negative binomial distribution. Cluster analysis and principal component analysis were also used to investigate whether there were any *hidden* groupings within the data points. This was important, because the single-cell analysis data would be used to design the spatio-temporal and interaction dynamics of the individual agent behaviours within our ABM.

Lesson 3: A top-down approach is more intuitive than a bottom-up approach when modelling complex dynamical biosystems. We used a top-down approach for modelling the NF-κB signalling pathway, which used three distinct hierarchical levels of the system: overall system-wide behaviours and dynamics, component-level interactions, and dynamics of individual components. We believe that developing the domain model using the increasing level of details that comes from a top-down approach is more intuitive than a bottom-up approach when modelling the complex emergent behaviours inherent to biological systems. This is because the concept of expanding functional aspects of complex biological systems (such as the NF-κB signalling pathway) through increasing the detail of lower level interactions, e.g. from their system-level properties, through to component interactions and individual component dynamics, is more intuitive than the ability to conceptualise system-wide behaviours from individual component-level dynamics by reporting them first, and then building up levels of the functional hierarchy. We do accept that this may be discipline specific however, and are aware of examples, such as economics [46–48], where a bottom-up approach is favoured due to the inability of top-down designed ABMs to represent consistently the micro and macro aspects of reality in their domains of interest.

Lesson 4: Validation is not just for the computational model. The iterative approach to developing the domain model, and the close collaboration between the modeller and domain expert provides an ability to use a number of validation techniques during model formulation. Unhelkar [49] advocates a number of key validation and verification activities that may be used with UML models. In addition, although Balci [50] provided an extensive review of verification and validation techniques that are suitable for computational model development and simulation-based experiments, we were able to use a subset of these techniques to validate our domain model of the NF-κB signalling pathway, which comprised: audits by the senior software engineer to ensure that the modelling adheres to established practices; desk checking by the modeller to ensure that individual diagrammatic and statistical models are correct, complete, consistent and unambiguous; face validation by the domain expert to compare the complete domain model against their detailed understanding and judgment of the real-world biological system; and structured walkthroughs by the whole group (modeller, senior software engineer and domain expert) to detect and document faults.

2.2. Development phase

The development phase of the CoSMoS process focuses on translating the domain model into a technical specification (termed *platform model*) based on the particular programming language/simulation framework to be used, and development of the actual computational model (termed *simulation platform*).

As per the domain model, the platform model may be a collection of informal notes relating to relevant aspects of the domain, but may also include informal sketches (such as cartoons), more formal diagrams (such as those produced with UML), mathematical equations, scientific constants (e.g. rate constants), and physical descriptors (such as size, quantity, lo-

cation, and speed) of simulation components and functions. Unlike the domain model, the key differentiator of the platform model is that it is required to include an implementation specific focus and should therefore document programming language specific requirements and/or workarounds, which may be required due to the programming language or computer architecture constraints during development of the simulator. The platform model also has a wider scope, outside of the specific simulation platform, in that it also includes the concept of instrumentation, which allows observation of system dynamics (through for example a visual front-end); the extraction of simulation data from the simulation platform; and analysis of this data as the basis for the *results model* (key deliverable of the exploration phase).

The third product of a CoSMoS project, is the simulation platform, which *"encodes the platform model into a software and hardware platform upon which simulations can be performed"* [39]. Whereas the platform model is the implementation specific abstraction of the domain model, the simulation platform is the instantiation of this in actual code. Our most recent ABMs of biological [37] and social [38] systems were developed using the Flexible Large-scale Agent-based Modelling Environment (FLAME) [51]. FLAME agents are defined within an XML template, which specifies the agent attributes and internal (X-Machine) states, and requires an additional set of C files, where the rule-based functionality associated with agent interactions is defined. Following model specification within these templates, the FLAME toolset parses the XML and C code to generate executable simulation code. The FLAME toolset also provides a simulation engine that manages the execution of simulations, and the interactions between the X-Machine agents (through a centralised message board). Time is discretized into *time-steps*, and within every time-step the individual X-Machines iterate through their internal (computational) states, which may culminate in interactions with other X-Machines, and updates to their (simulated) real-world domain states.

We have learned a number of valuable lessons following the development of our platform model and simulation platform, which are defined below:

Lesson 5: Crucial to ensure separation of the conceptual model of the real-world system versus the technical specification of the ABM. Although advocated by the CoSMoS process and several others [18,29,41], we learned independently, the need for the abstracted view of the real-world system to be kept seperated from (and not combined with) the technical specification of the computational model. This separation ensures the abstracted view of the real-world system and the technical specifications of the computational system remain discrete models, and thus aims to minimise confusion (between the modeller and domain expert) during the development of the computational model around what aspects of the programming code relate to real-world functional requirements, and what aspects are necessary as technical workarounds due to constraints of the specific programming frameworks being used (e.g. in our case: Java MASON [35,36], Communicating Stream X-Machines [52,53], and FLAME [37,38]). As such, we believe the process of platform modelling to be an integral part of the development lifecycle for computational models of complex dynamical systems, and that its key value (in our projects) was in providing an unambiguous technical specification for the resultant computational model.

This was particularly important for our ABM of the NF-κB signalling pathway because: a number of assumptions were made on how real-world dynamics of the system would be represented in the model, such as the movement of agents within the internal structures of a biological cell environment, and the probabilistic (and often reversible) interactions of agents following their movement into the interaction boundary zone of a complementary agent; mappings between real-world biological states and internal computational model states were defined; technical workarounds were developed due to the modelling framework chosen (in our case FLAME); and constraints were encountered, which needed overcoming, such as the need to reduce the number of agents from circa 500,000 in the real-world domain to 5,000 in the ABM to ensure simulation-based experimentation was tractable. In addition, we were required to develop a number of peripheral computational tools (termed *instruments*) in order to run simulations and analyze the resultant data: Python scripts to allow generation of XML starting parameter files for running simulations in FLAME; Ruby, MS DOS batch, and Unix shell scripts, to allow submission of simulation runs to computing resources; Python and R scripts to analyze the data and automatically generate various graphs; a visualization front-end to provide an animated view of system dynamics over time; and Python and Matlab scripts to analyze the output data for statistical significance and effect magnitude.

Furthermore, regarding the validity of the platform model, we believe that the ODD (Overview, Design Concepts, and Details) protocol developed by Grimm et al. [54,55] may be useful for ensuring a comprehensive platform model is developed, and in a standardised way. Indeed, Stepney [56] has discussed how the CoSMoS process is ODD protocol compliant, and has provided a mapping between the ODD protocol components and CoSMoS project documentation. In addition, the ODD protocol has recently been updated to define the *decision making* that takes place by humans within complex social and environmental systems, which should standardise the way that human decision making is modelled within ABMs of complex systems that involve humans (e.g. social systems, economics, epidemiology).

Lesson 6: Important to ensure sufficient time for understanding the intricacies (and limitations) of the specific modelling framework in order to develop suitable workarounds. Two main technical issues were encountered during the development of our recent ABM of the NF-κB signalling pathway, which were due to the background design decisions taken during the development of the FLAME simulation framework and the associated mechanisms for processing the internal state transitions of the X-Machines [51,57]. The first issue related to the efficient handling of basal dissociation of the NF-κB-IκBα inhibited complex back to free NF-κB agents and IκBα agents. The second issue related to the use of a pseudo-random number generator (PRNG), and the need to set an associated *seed value* at the simulation level.

The first issue around basal dissociation related to the computational workaround needed for simulations to run efficiently. Briefly, upon binding of IκBα with NF-κB to form the inhibited complex, instead of the IκBα agent having its 3D cartesian (X, Y, Z) coordinates set at each time-step to mirror that of the relevant NF-κB agent, we instead chose to update

the status of the NF-κB agent to be that of the inhibited (bound) complex, and removed the IκBα agent from the system in order to improve the efficiency of simulation runs. This results in a reduction in the total number of individual agents within the system, but as the new status of the NF-κB agent models that of the inhibited complex, it can be argued that the total number of NF-κB and IκBα *molecules* within the system from a biological perspective remains consistent, and therefore respects the principle of the conservation of mass [58]. Functionality to provide a small degree of basal dissociation of the complex following its formation, required the updating of the NF-κB agent's state from *inhibited* to *free*, via a straightforward update to the X-Machines internal memory, but it was necessary to introduce *new* IκBα agents back to the simulation to model the release of inhibitor back in to the system. The simplest approach to provide this functionality would be to use a global counter, initially set to the highest agent ID of the set of IκBα agents at the start of a simulation. Unfortunately, as FLAME was developed to run across multiple clusters using parallel processing functionality, the concept of a *global mutable parameter* does not exist, as this would make the running of simulations inefficient due to the issue of concurrency locking. We expended significant effort in identifying this problem and the subsequent development of an efficient workaround.

The second issue relates to setting a seed value in order to use a PRNG. The theory behind setting a seed value for a PRNG is that you set once per simulation run, and use the random number functionality many times [59,60] within the simulation run. Unfortunately, as discussed above, due to the parallel nature of FLAME, there is no ability to set or change a global constant from within a simulation run, as it would be inefficient for this to propagate across the multiple nodes within the computer cluster/grid. As such, not only can you not update a global variable, such as total counts, but you are hindered from setting the PRNG seed at the global level. You therefore encounter the issue of either not being able to set your own PRNG seed value, and thus relying on the seeds generated using the system clock, or to develop a workaround. Once we had identified the fact that the only way to set the PRNG seed value was within the definition of an agent (the C functions file), we were able to create a workaround where a dummy agent was created, whose sole purpose was to set the PRNG seed value in the first time-step of a simulation run, with subsequent removal from the simulation at time-step two - effectively killing the agent.

Lesson 7: Essential to engage with domain expert and to develop robust mapping between real-world dynamics and computational model to ensure calibration process is fit-for-purpose. Before simulation-based experimentation can be performed, the agent-based model must be calibrated so that baseline behaviours and system dynamics are established. This calibrated baseline is crucial to ensure that the resulting system behaviours and dynamics that emerge following perturbations applied to the computational model during simulation-based experimentation, are emerging due to the intervention, and not as an artefact of model implementation [61]. Our ABMs of EAE [35,36] and NF-κB [37] were calibrated against published literature and the domain expert's understanding of the complex dynamical system of interest. The contribution of the domain expert in this calibration activity has been essential to ensure the respective computational models were not only grounded in the domain, but also adhered to the scope of the domain and platform models, and that the individual agents appropriately modelled the heterogeneous dynamics of data points from a population of their component type in the real-world system (e.g. the negative binomial distribution of NF-κB). As discussed by Kirschner et al. [62], there are several approaches for estimating the parameter values of computational models during the calibration process: 1) direct experimental determination of a parameter; 2) simultaneous estimation of several parameters at once by fitting experimental data to a model; and 3) estimation of a parameter based on known values for a similar system.

The calibration process that we utilised to align the behaviour of our ABMs with that of the underlying biological system used a mixture of the three approaches discussed by Kirschner for parameter value estimation. Environmental parameter values were approximated from the literature; a number of parameter values regarding agent interactions were arbitrarily set, such as the interaction radius within which agents need to enter before they are eligible for probabilistic binding, and the delay applied to nuclear receptors following translocation of an agent before they may translocate another agent (with respect to the NF-κB ABM); the parameters that we believe are fundamental to the emergent behaviour were estimated through a process of varying parameter values during multiple simulation runs, until simulation dynamics approximated (via qualitative curve-fitting) to those of the data from the real-world complex system.

Stepney [63] advocates the need for a translation step in ascertaining the desired simulation results that are required to calibrate computational models. For example, the domain model may comprise both parameter values and experimental data (d_i) derived from the real-world complex system, which following scientific analysis yields domain results data (d_r). To move to the simulated world, the domain model data needs to be translated to appropriate simulation results values s_i, which can be documented in the platform model. A simulation experiment, given input data s_i, and using the simulation parameters, functions and methods (T_{ds}) called within the computational model during simulation runs, will produce raw simulation results data s_r. Through the use of multiple simulation replicates, this raw simulation results data is then processed to generate simulation output data (s_o), which could comprise of the median averages of the individual s_r data. The calibration exercise is to adjust the simulation parameters and translation functions T_{ds} to achieve an approximation of domain results data to simulation results data ($d_r \sim s_r$). The relationship between domain and simulation results does not need to be exact, because a certain degree of variation is required to produce the stochasticity inherent to complex systems, be they biological or social; it does need to be statistically similar, or show qualitative agreement however.

Lesson 8: Essential to understand the heterogeneity within the real-world and computational systems. Read et al. [21] and Alden et al. [64] advise that there are two sources of uncertainty within computational models of real-world domains that require analysis before the results of simulation-based experimentation may be used within a predictive capacity. The first source, *epistemic* uncertainty, arises because our knowledge of the real-world system that we are modelling

is incomplete, and therefore we do not have complete assurance that the parameter values derived through the literature, the domain experts professional opinion, and the calibration process, are wholly accurate [65]. The second source, *aleatory* uncertainty, arises from the inherent stochasticity in systems, be that the computational model itself, or the underlying real-world system [65].

As discussed previously, one of the strengths of using an agent-based approach is the ability to generate stochasticity at the individual agent level. As such, sources of randomness can be applied to the relevant places within the model, as opposed to a more general noise term used by differential equation models, which is often added arbitrarily to an aggregate equation. With this in mind, it is important to understand whether simulation results are affected by uncertainty in the computational and real-world systems. Statistical techniques rely on the fact that all data required for model validation can be collected from the system, and that the results of simulation can be compared with the results from the real-world system when the model is run with the *same* input data that derive the real system. Due to the stochasticity inherent to complex dynamical systems, we need to ensure that comparisons between the model and system outputs are made in the context of the uncertainty within the real-world and computational systems.

Within our modelling project of the NF-κB signalling pathway, we performed a number of statistical techniques on the biological (single-cell) data to understand the heterogeneous nature of the real-world system (see the statistical techniques section of our domain model paper [40]), along with sensitivity and uncertainty analysis on the parameters within our computational model, to understand epistemic uncertainty within the ABM, which highlighted that a small number of parameters displayed robust, yet fragile, dynamics. In addition, we performed aleatory uncertainty analysis to investigate the *noise* due to the stochastic nature of the agent-based model following setting of PRNG seed values for each replicate simulation run. Such aleatory uncertainty can severely affect our ability to compare experimental results against reference dynamics (e.g. calibrated control dynamics) and thus interpret the simulation results with respect to the real-world domain [66]. Our analysis indicated that a minimum of 175 replicate simulation runs were required in order to generate stable median averages of simulation data, mitigate the stochastic effects introduced through setting different PRNG seed values, and thus develop confidence that simulation results are representative of the *experimental* condition(s) on which the simulation was run, and not an artefact of our computational model that is caused by the specific PRNG seed value [37].

Lesson 9: Need for thorough verification and validation of the ABM. Verification and validation are two fundamental processes within software engineering in general, and computational model development in particular, as underpinned by the IEEE 1012 system and software verification and validation standard for software quality assurance [67]. For a high-level explanation of the difference between them, we refer to Boehm's maxim *"validation is building the right system, whereas verification is building the system right"* [68], with Balci [50] providing a comprehensive discussion of validation, verification, and testing techniques for simulation studies.

Balci [69] lists a taxonomy of validation and verification and categorises into 6 perspectives: informal, e.g. inspections, reviews and walkthroughs; static, e.g. data flow analysis, semantic analysis, structural analysis, and syntax analysis; dynamic, e.g. graphical comparisons, sensitivity analysis, visualization, and stress testing; symbolic, e.g. cause-effect graphing, and path analysis; constraint, e.g. boundary analysis, unductive assertions, and assertion checking; and formal, e.g. induction, inference, and proof of corrections. By comparison, Sargent [70] discusses three distinct types of validation (conceptual model validation, operational validation, and data validity) along with verification of the computational model.

We used a number of the verification and validation techniques described by Balci within the lifecycle of our modelling projects. Our experiences confirm that of Gurcan et al. [28], who argue that a number of the traditional techniques for validation and verification are easily transferable to agent-based models, and of Klugl [71], who suggests that the main problem with the process of validating agent-based models of complex dynamical systems, is the lack of available empirical data at the level of the agent interactions. We are however cognisant that ABMs belonging to different fields are designed and developed at different levels of abstraction, and will need to be validated in accordance with disciplinary norms. For example, we are aware that validating ABMs within the disciplines of economics and accounting/finance has been problematic for a significant period of time, but has recently gained considerable attention from the modelling community [72–74]. In addition, Axtell and Epstein [75] advise that there are four levels upon which ABMs can be validated:

- **Level 0:** The model is a *caricature* of reality, as established through the use of simple graphical devices (e.g. allowing visualization of agent notation);
- **Level 1:** The model is in *qualitative agreement with empirical macro-structures*, as established by plotting, say, distributional properties of the agent population;
- **Level 2:** The model produces *quantitative agreement with empirical macro-structures*, as established through on-board statistical estimation routines; and finally,
- **Level 3:** The model exhibits *quantitative agreement with empirical micro-structures*, as determined from cross-sectional and logitudinal analysis of the agent population.

2.3. Exploration phase

The exploration phase of the CoSMoS process focuses on simulation-based experimentation (using the computational model), and translating the resultant simulation data back to the real-world domain. These results are documented within the *Results Model*, which provides an understanding of the real-world complex system that has been gained through simu-

lation and analysis of any causal dynamics emerging from the experimental conditions that were used. We have learned a key lesson following simulation-based experimentation with our ABMs.

Lesson 10: Need for rigorous analysis of simulation data. Our ABMs have taken an abstracted view of the relevant complex dynamical system in terms of both the agent types, and indeed the numbers of the respective agents. As such, we cannot directly compare an executing model to the real-world data, and have therefore utilised the process advocated by Stepney [63] in order to translate the real-world data into an appropriate range of numbers for calibration of the simulation platform. We therefore believe that a rigourous framework for calibration, uncertainty analysis, and interpretation of simulation results, is essential to ensure that the model adequately relates back to the underlying real-world domain, and furthermore that we can have confidence that the results of simulation-based experimentation are reflective of the experimental conditions and not a mere random occurence due to the stochastic nature of the model.

As with all ABMs that utilise stochastic behaviours and set different PRNG seed values for each replicate, there is significant aleatory uncertainty within our models. Such aleatory uncertainty can severely affect our ability to compare experimental results against reference dynamics. This being the case, aleatory uncertainty analysis allows us to investigate the uncertainty that is introduced through the use of PRNGs, and to calculate a minimum number of replicate simulations to generate a stable median average of the simulation results. The calculation of the minimum number of replicates, and its use within simulation-based experimentation to generate a stable median, allows us to mitigate stochastic effects and thus develop confidence that our simulation results are representative of the condition(s) on which the simulation was run, and not an artefact of our computational model that is caused by any specific PRNG seed value.

Our aleatory uncertainty analyses have utilised the Kolmogorov-Smirnov (KS) test [76] and the Vargha and Delaney non-parametric A-Test [77] (a modified form of the Mann and Whitney U-Test [78]). The KS-Test is used to understand the *statistical significance* of differences between two distributions, and the A-Test is used to understand the effect magnitude (which we term *scientific significance*) of differences between two distributions. In our analyses, we used the KS-Test and A-Test (KS-Test p-value less than 0.05 and A-Test score < 0.29 or > 0.71) to interpolate when a stable median average was gained, to calculate the minimum number of replicates needed for simulation-based experimentation. As discussed above, we discovered that 175 simulation replicates were required for our NF-κB ABM, and similarly that 1,000 simulation replicates were required for our EAE ABM, in order to mitigate stochastic effects from the use of different PRNG seed values, and thus develop confidence that simulation results are representative of the condition(s) on which the simulation was run. We also utilised the KS-Test and A-Test to investigate whether there is statistical significance and scientific significance between the median average distributions of simulation data from experimental conditions and control dynamics. We require both to be significant in order to infer causal relationships between the simulation data and changes to the simulation setup (e.g. parameter values with respect to calibrated control dynamics).

We believe that rigourous statistical underpinnings, such as those used within our modelling projects, are not an *added extra*, but are in fact an essential part of ABM development and analysis of simulation-based experimental results. We also believe that without such rigour, it will be hard for the field of computational modelling to gain credibility with the wider scientific community.

3. Discussion

Real-world systems are complex, with dynamical behaviours and characteristics that result from a highly connected set of interaction networks that function through time and space. One of the main strengths of the computational systems approach, is that it focuses on three key properties of complex systems: 1) system structures, 2) system dynamics, and 3) system control [79]. Agent-based modelling and simulation can be used to test our theories of the underlying mechanics of component interactions within systems and their resulting dynamics. Specifically, the use of ABMs can facilitate the micro-to-macro mapping of systems [4] through the relaxing of assumptions or altering of the mechanistic interactions at the individual agent-level, to investigate the emergent behaviour at the system-level. As such, they are a low-cost way for hypothesis generation (of the real-world system) and directing experimental design to test these (within the abstracted computational model). The ABMs thus become a scientific instrument for investigation [80], and therefore require a principled approach to their use, which is akin to the scientific method, to ensure they are fit-for-purpose.

Due to a simulation being the executable form of a computational model, the process of developing the model and encoding the real-world *input* data into the simulation, introduces many design decisions and assumptions that need to be visible to all users and beneficiaries of the model, so that the simulation results generated through simulation-based experimentation can be appropriately interpreted in relation to the real-world domain of interest. Credibility of the results from simulation-based experimentation depends on a plethora of activities and approaches from software engineering, in particular, model correctness and the accurate formulation of problems that need to be solved/investigated. Therefore, a principled approach to design and development should be adopted, such as the CoSMoS framework, along with rigorous quality assurance through verification, validation and testing (see Fachada et al. [81] for discussion of statistical tests that can be used on ABM simulation outputs). In addition, although agent-based modelling and simulation is a powerful tool for increasing our understanding of the mechanistic behaviours of complex dynamical systems, caution should be applied when using simulation results in a predictive capacity. This is because all ABMs are ultimately abstractions of the real-world systems that they capture, and this separation must be appreciated in the interpretation of simulation results [64]. In particular, the affects of uncertainty within the computational model (aleatory uncertainty) and our understanding of

Table 1
The Ten Lessons Learned. The ten lessons learned during the design, development, and application of our agent-based models of complex dynamical systems. The scope column reflects the fact that we consider some lessons to be generalisable to all agent-based modelling projects, and some to be specific to the particular objectives and/or research questions of the project. For instance, we are aware that although some of our lessons may be essential in the natural and physical sciences due to the multi-disciplinary teams, they may not be essential to all disciplines where it may be acceptable for the modeller to also act as the domain expert.

Number	Lesson	Phase	Scope
Lesson 1	Close collaboration between modeller and domain expert is required to ensure a solid foundation of the real-world system.	Discovery	Specific
Lesson 2	A variety of diagrammatic notations and statistical techniques are required to develop a comprehensive conceptual model of a complex dynamical system.	Discovery	General
Lesson 3	A top-down approach is more intuitive than a bottom-up approach when modelling complex dynamical biosystems.	Discovery	Specific
Lesson 4	Validation is not just for the computational model, but should also be performed on the conceptual model.	Discovery	General
Lesson 5	Crucial to ensure separation of the conceptual model of the real-world system versus the technical specification of the ABM.	Development	General
Lesson 6	Important to ensure sufficient time for understanding the intricacies (and limitations) of the specific modelling framework in order to develop suitable workarounds.	Development	General
Lesson 7	Essential to engage with domain expert(s) and to develop robust mapping between real-world dynamics and computational model to ensure calibration process is fit-for-purpose.	Development	Specific
Lesson 8	Essential to understand the heterogeneity within the real-world and computational systems.	Development	General
Lesson 9	Need for thorough verification and validation of the ABM.	Development	General
Lesson 10	Need for rigorous analysis of simulation data.	Exploration	General

the real-world system (epistemic uncertainty) need to be considered when using simulation-based experimentation for the purpose of making predictions of the real-world complex system under study.

Finally, we would like to emphasise that the need for a principled approach to design and development of ABMs that model complex dynamical systems cannot be understated. We are aware that modern simulation frameworks have made the development of ABMs relatively easy, even for domain experts, however through first-hand experience in our ABM development projects, we are also aware how easy it can be to inadvertently misinterpret simulation data, and thus make incorrect predictions on real-world system dynamics. As such, we would like to share these ten lessons learned during the design, development and application of our ABMs of complex dynamical systems, and summarise these in Table 1.

Acknowledgments

We are grateful to the three anonymous reviewers who provided helpful and constructive feedback to enhance the paper. In addition, we thank Dr Kieran Alden (University of York) and Dr Daniel Moyo (Data Scientist at Holiday Extras) for their comments on an early draft of this paper.

Funding: This work was in part funded by a Lancaster University Early Career Small Grant and a Management and Business Development Fellowship awarded by the Economic and Social Research Council, and jointly funded with the Society for the Advancement of Management Studies and the United Kingdom Commission for Employment and Skills (SAMS-ESRC-UKCES MBDF; Grant No. ES/L002612/1). The funders had no involvement in study design; development of the underlying ABMs, their analysis and interpretation of results; in the writing of this manuscript, or the decision to submit this manuscript for publication.

References

[1] N. Gilbert, S. Bankes, Platforms and methods for agent-based modeling, Proc. Natl. Acad. Sci. 99 (suppl 3) (2002) 7197–7198.
[2] S.F. Railsback, S.L. Lytinen, S.K. Jackson, Agent-based simulations platforms: review and development recommendations, Simulation 82 (9) (2006) 609–623.
[3] P.-O. Siebers, C.M. Macal, J. Garnett, D. Buxton, M. Pidd, Discrete-event simulation is dead, long live agent-based simulation!, J. Simul. 4 (2010) 204–210.
[4] J.L. Epstein, Agent-based computational models and generative social science, Complexity 4 (5) (1999) 41–60.
[5] E. Bonabeau, Agent-based modeling: methods and techniques for simulating human systems, PNAS 99 (suppl.3) (2002) 7280–7287.
[6] C.M. Macal, M.J. North, Tutorial on agent-based modeling and simulation, J. Simul. 4 (3) (2010) 151–162.
[7] C.M. Macal, M.J. North, Introductory tutorial: agent-based modeling and simulation, in: A. Tolk, S.Y. Diallo, I.O. Ryzhov, L. Yilmaz, S. Buckley, J.A. Miller (Eds.), Proceedings of the 2014 Winter Simulation Conference, 2014, pp. 6–20. Savannah, Georgia, USA.
[8] R.B. Greaves, M.N. Read, J. Timmis, P.S. Andrews, J.A. Butler, B.O. Gerckens, V. Kumar, In silico investigation of novel biological pathways: the role of CD200 in regulation of t-cell priming in experimental autoimmune encephalomyelitis, BioSystems 112 (2013) 107–121.
[9] T. Lux, M. Marchesi, Volatility clustering in financial markets: a microsimulation of interacting agents, Int. J. Theor. Appl. Finance 3 (4) (2000) 675–702.
[10] P. Garnett, A tipping point in 300 years of banking? a conceptual simulation of the british banking system, Nat. Comput. 14 (2015) 25–37.
[11] A.A. Dodson, S. Stepney, E. Uprichard, L. Caves, Using the CoSMoS approach to study schelling's bounded neighbourhood model, in: S. Stepney, P.S. Andrews (Eds.), Proceedings of the 2004 Workshop on Complex Systems Modelling and Simulation, Luniver Press, New York, NY, USA, 2014, pp. 1–12.
[12] S.P. Wilcox, Agentizing the social science of crime, in: S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, M. Fu (Eds.), Proceedings of the 2011 Winter Simulation Conference, IEEE, Phoenix, Arizona, USA, 2011, pp. 333–344.
[13] V. Sahasrabudhe, S. Kanungo, R. Iyer, Understanding the impact of communication technologies on virtual team performance: an agent-based simulation model, in: S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, M. Fu (Eds.), Proceedings of the 2011 Winter Simulation Conference, IEEE, Phoenix, Arizona, USA, 2011, pp. 321–332.

[14] G. Ziervogel, M. Bithell, R. Washington, T. Downing, Agent-based social simulation: a method for assessing the impact of seasonal climate forecast applications among smallholder farmers, Agric. Syst. 83 (1) (2005) 1–26.
[15] R.A. Williams, An agent-based model of the IL-1 stimulated Nuclear Factor-kappa B signalling pathway, University of York, York, UK, 2014 PhD thesis.
[16] R.A. Williams, J. Timmis, E.E. Qwarnstrom, Computational models of the NF-κB signalling pathway, Computation 2 (4) (2014) 131–158.
[17] S.J.E. Taylor, Agent-based modeling and simulation, OR Essentials, Palgrave Macmillan, 2014.
[18] M. Pidd, Computer Simulation in Management Science, John Wiley & Sons Inc, Chichester, UK, 2006.
[19] N. Gilbert, Agent-based models, Vol. 7 of Quantitative Applications in the Social Sciences, SAGE Publications, Thousand Oaks, California, 2008.
[20] A.M. Law, Simulation Modeling and Analysis, McGraw Hill Education, New York, NY, USA, 2014.
[21] M. Read, P.S. Andrews, J. Timmis, V. Kumar, Techniques for grounding agent-based simulations in the real domaa case study in experimental autoimmune encephalomyelitis, Math. Compt. Model. Dyn. 18 (1) (2012) 67–86.
[22] F.A.C. Polack, T. Hoverd, A.T. Sampson, S. Stepney, J. Timmis, Complex systems models: engineering simulations, in: S. Bullock, J. Noble, R. Watson, M. Bedau (Eds.), Proceedings of the 11th International Conference on the SImulation and Synthesis of Living Systems (ALife), MIT Press, Winchester, UK, 2008, pp. 482–489.
[23] F.A.C. Polack, Arguing validation of simulations in science, in: S. Stepney, P.H. Welch, P.S. Andrews, A.T. Sampson (Eds.), Proceedings of the 2010 Workshop on Complex Systems Modelling and Simulation, Luniver Press, Odense, Denmark, 2010, pp. 51–74.
[24] K. Alden, P.S. Andrews, F.A.C. Polack, H. Veiga-Fernandes, M.C. Coles, J. Timmis, Using argument notation to engineer biological simulations with increased confidence, J. R. Soc. Interface 12 (2015). 20141059
[25] E. Tatara, M.J. North, T.R. Howe, N.T. Collier, J.R. Vos, An introduction to repast simphony modeling using a simple predator-prey example, in: D.L. Sallach, C.M. Macal, M.J. North (Eds.), Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects, Argonne National Laboratory and The University of Chicago, North American Association for Computational Social and Organizational Science, 2006, pp. 83–93. Chicago, Illinois, USA.
[26] M. Wooldridge, N.R. Jennings, Software engineering with agents: pitfalls and pratfalls, IEEE Internet Comput. (1999) 20–27.
[27] M. Wheeler, S. Bullock, E.A. Di Paolo, J. Noble, M. Bedau, P. Husbands, S. Kirby, A. Seth, The view from elsewhere: perspectives on ALife modeling, Artif. Life 8 (1) (2002) 87–100.
[28] O. Gürcan, O. Dikenelli, C. Bernon, A generic testing framework for agent-based simulation models, J. Simul. 7 (3) (2013) 183–201.
[29] D. Gianni, A.D. Ambrogio, A. Tolk, Modeling and simulation-based systems engineering handbook, Engineering Management, CRC Press, Boca Raton, Florida, 2015.
[30] MSCO, Conceptual Model Development and Validation, Department of Defense, (2012), http://www.msco.mil/documents/RPG/ST14_Conceptual_Model.pdf.
[31] M.N. Read, K. Alden, L.M. Rose, J. Timmis, Automated multi-objective calibration of biological agent-based simulations, J. R. Soc. Interface 13 (122) (2016). 20160543
[32] F. Lamperti, A. Roventini, A. Sani, Agent-based model calibration using machine learning surrogates, Laboratory of economics and management working paper, 2017. Institute of Economics, Scuola Superiore Sant'Anna, Pisa (Italy).
[33] G. Dosi, M.C. Pereira, M.E. Virgillito, On the robustness of the fat-tailed distribution of firm growth rates: A global sensitivity analysis, J. Econ. Interact. Coord. (2017) 1–21.
[34] S. Barde, S. van der Hoog, An empirical validation protocol for large-scale agent-based models, 2017, School of Economics Discussion Paper KDPE 1712, University of Kent.
[35] R.A. Williams, R. Greaves, M. Read, J. Timmis, P.S. Andrews, V. Kumar, iN silico investigation into dendritic cell regulation of CD8Treg mediated killing of Th1 cells in murine experimental autoimmune encephalomyelitis, BMC Bioinf. 14 (Suppl 6) (2013). S9.
[36] M. Read, P.S. Andrews, J. Timmis, R.A. Williams, R.B. Greaves, H. Sheng, M.C. Coles, V. Kumar, Determining disease intervention strategies using spatially resolved simulations, PLoS ONE 8 (11) (2013). E80506.
[37] R.A. Williams, J. Timmis, E.E. Qwarnstrom, Investigating IKK dynamics in the NF-κB signalling pathway using X-machines, in: Proceedings of the IEEE 2017 Congress on Evolutionary Computation, 2017. IEEE Xplore, Donastia-San Sebastian, Spain.
[38] R.A. Williams, Modelling conflict within the social networks of large multi-vendor software projects using communicating stream X-machines, in: Proceedings of the European Conference on Artificial Life 2015, MIT Press, York, UK, 2015, p. 79.
[39] P.S. Andrews, F.A.C. Polack, A.T. Sampson, S. Stepney, J. Timmis, The CoSMoS process, version 0.1: a process for the modelling and simulation of complex systems, Tech. Rep. YCS-2010-453, 2010.
[40] R.A. Williams, J. Timmis, E.E. Qwarnstrom, Statistical techniques complement UML when developing domain models of complex dynamical biosystems, PLoS ONE 11 (8) (2016). E0160834.
[41] B.S.S. Onggo, Towards a unified conceptual model representation: a case study in healthcare, J. Simul. 3 (2009) 40–49.
[42] S. Robinson, Conceptual Modeling for Discrete-Event Simulation, CRC Press, Boca Raton, Florida, 2011. Ch. Conceptual Modeling for Simulation: Definition and Requirements, 3–30.
[43] P.S. Andrews, S. Stepney, T. Hoverd, F.A.C. Polack, A.T. Sampson, J. Timmis, CoSMoS process, models, and metamodels, in: S. Stepney, P.H. Welch, P.S. Andrews, C.G. Ritson (Eds.), Proceedings of the 2011 Workshop on Complex Systems Modelling and Simulation, CoSMoS, Luniver Press, 2011, pp. 1–14.
[44] S. Ghosh, Y. Matsuoka, Y. Asai, K.-Y. Hsin, H. Kitano, Software for systems biology: from tools to integrated platforms, Nat. Rev. Genet. 12 (2012) 821–832.
[45] Object Management Group, Unified modeling language superstructure specification v2.4, http://www.omg.org/spec/UML/2.4/Superstructure.
[46] D. Delli Gatti, E. Gaffeo, M. Gallegati, G. Giulioni, A. Palestrini, Emergent macroeconomics: an agent-based approach to business fluctuations, New Economic Windows, Springer, Milan, Italy, 2008.
[47] D. Delli Gatti, S. Desiderio, E. Gaffeo, P. Cirillo, M. Gallegati, Macroeconomics from the bottom-up, New Economic Windows, Springer, Milan, Italy, 2011.
[48] G. Fagiolo, A. Roventini, Macroeconomic policy in DSGE and agent-based models redux: new developments and challenges ahead, J. Artif. Soc. Social Simul. 20 (1) (2017). 1.
[49] B. Unhelkar, Verification and validation for quality of UML 2.0 models, Systems Engineering and Management, Wiley Interscience, Hoboken, New Jersey, 2005.
[50] O. Balci, Validation, verification, and testing techniques throughout the life cycle of a simulation study, Ann. Oper. Res. 53 (1994) 121–173.
[51] S. Coakley, Formal software architecture for agent-based modelling in biology, University of Sheffield, 2007 Ph.d. thesis.
[52] L. Zhang, R.A. Williams, D. Gatherer, Rosen's (M,R) system in unified modelling language, BioSystems 139 (2016) 29–36.
[53] M. Palmer, R.A. Williams, D. Gatherer, Rosen's (M,R) system as an X-machine, J. Theor. Biol. 408 (2016) 97–104.
[54] V. Grimm, U. Berger, F. Bastiansen, S. Eliassen, V. Ginot, J. Giske, J. Goss-Custard, T. Grand, S.K. Heinz, G. Huse, A. Huth, J.U. Jepsen, C. Jorgensen, W.M. Mooij, B. Muller, G. Pe'er, C. Piou, S.F. Railsback, A.M. Robbins, M.M. Robbins, E. Rossmanith, N. Ruger, E. Strand, S. Souissi, R.A. Stillman, R. Vabo, U. Visser, D.L. DeAngelis, A standard protocol for describing individual-based and agent-based models, Ecol. Model. 198 (2006) 115–126.
[55] V. Grimm, U. Berger, D.L. DeAngelis, J.G. Polhill, S.F. Railsback, The ODD protocol: a review and first update, Ecol. Model. 221 (2010) 2760–2768.
[56] S. Stepney, P.S. Andrews, CoSMoS simulation experiment reproducibility and the ODD protocol, in: S. Stepney (Ed.), Proceedings of the 2013 Workshop on Complex Systems Modelling and Simulation, CoSMoS, Luniver Press, 2013, pp. 93–108.
[57] S. Coakley, R. Smallwood, M. Holcombe, From molecules to insect communities - how formal agent-based computational modelling is uncovering new biological facts, Sci. Math. Jpn. (2006) 765–778. Online e-2006.
[58] D.A. Beard, E. Babson, E. Curtis, H. Qian, Thermodynamic constraints for biochemical networks, J. Theor. Biol. 228 (2004) 327–333.

[59] S.C. Barry, How much impact does the choice of a random number generator really have? Int. J. Geogr. Inf. Syst. 25 (4) (2011) 523–530.
[60] K.P. van Niel, S.W. Laffan, There is no good excuse for a bad random number generator: a reply to Barry, Int. J. Geogr. Inf. Syst. 25 (4) (2011) 531–539.
[61] S. Evans, K. Alden, L. Cucurull-Sanchez, C. Larminie, M.C. Coles, ASPASIA: a toolkit for evaluating the effects of biological interventions on SBML model behaviour, in: M.C. Kullberg, J. Timmis (Eds.), PLoS Comput. Biol., 13, 2017. E1005351.
[62] D.E. Kirschner, S.T. Chang, T.W. Riggs, N. Perry, J.J. Linderman, Toward a multiscale model of antigen presentation in immunity, Immunol. Rev. 216 (2007) 93–118.
[63] S. Stepney, A pattern language for scientific simulations, in: S. Stepney, P.S. Andrews, M. Read (Eds.), Proceedings of the 2012 Workshop on Complex Systems Modelling and Simulation, Luniver Press, Orleans, France, 2012, pp. 77–103.
[64] K. Alden, M. Read, J. Timmis, P.S. Andrews, H. Veiga-Fernandes, M.C. Coles, Spartan: a comprehensive tool for understanding uncertainty in simulations of biological systems, PLoS Comput. Biol. 9 (2) (2013). E1002916.
[65] J.C. Helton, Uncertainty and sensitivity analysis for models of complex systems, in: T.J. Barth, M. Griebel, D.E. Keyes, R.M. Nieminen, D. Roose, T. Schlick (Eds.), Computational Models in Transport: Verification and Validation, Springer, Heidelberg, Germany, 2008, pp. 207–228.
[66] R.B. Harris, L.A. Maguire, M.L. Shaffer, Sample sizes for minimum viable population estimation, Conserv. Biol. 1 (1) (1987) 72–76.
[67] IEEE, 1012–2012 IEEE Standard for System and Software Verification and Validation, IEEE, DOI: 10.1109/IEEESTD.2012.6204026 (2012).
[68] B.W. Boehm, Verifying and validating system requirements and design specifications, IEEE Softw. 1 (1) (1984) 75–88.
[69] O. Balci, Principles and techniques of simulation validation, verification, and testing, in: C. Alexopoulos, K. Kang, W.R. Lilegdon, D. Goldsman (Eds.), Proceedings of the 1995 Winter Simulation Conference, 1995, pp. 147–154. Arlington, VA, USA.
[70] R.G. Sargent, Verification and validation of simulation models, in: Proceedings of the 2010 Winter Simulation Conference (*WSC*), IEEE, Baltimore, MD, USA, 2010, pp. 166–183.
[71] F. Klugl, A validation methodology for agent-based simulations, in: Proceedings of the 2008 ACM Symposium on Applied Computing, 2008, pp. 39–43. Association for Computing Machinery.
[72] P. Windrum, G. Fagiolo, A. Moneta, Empirical validation of agent-based models: alternatives and prospects, J. Artif. Soc. Social Simul. 10 (2) (2007). 8.
[73] M. Guerini, A. Moneta, A method for agent-based models validation, J. Econ. Dyn. Control 82 (2017) 125–141.
[74] S. Barde, A practical, accurate, information criterion for Nth order markov processes, Comput. Econ. 50 (2017) 281–324.
[75] R.L. Axtell, J.L. Epstein, Agent-based modeling: understanding our creations, Bull. Santa Fe Institute Winter (1994) 28–32.
[76] F.J. Massey, The Kolmogorov-Smirnov test for goodness of fit, J. Am. Stat. Assoc. 46 (253) (1951) 68–78.
[77] A. Vargha, H.D. Delaney, A critique and improvement of the CL common language effect size statistics of McGraw and Wong, J. Educ. Behav. Stat. 25 (2000) 101–132.
[78] H.B. Mann, D.R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, Ann. Math. Stat. 18 (1947) 50–60.
[79] T. Ideker, T. Galitski, L. Hood, A new approach to decoding life: systems biology, Annu. Rev. Genomics Hum. Genet. 2 (2001) 343–372.
[80] J. Bown, P.S. Andrews, Y. Deeni, A. Goltsov, M. Idowu, F.A.C. Polack, A.T. Sampson, M. Shovman, S. Stepney, Engineering simulations for cancer systems biology, Curr. Drug Targets 13 (12) (2012) 1560–1574.
[81] N. Fachada, V.V. Lopes, R.C. Martins, A.C. Rosa, Model-independent comparison of simulation output, Simul. Model. Pract. Theory 72 (2017) 131–149.