Complexity of a Modelling Exercise: A Discussion of the Role of Computer Simulation in Complex System Science

Shifting the emphasis from a model *to a modeling* task, *which involves both a computer model and a modeler, we ask what makes a problem complex. We propose that a modeling task can be seen as a set of questions-and-answers, nested at multiple levels. The role of the modeler then lies in posing the questions and choosing the best procedure to answer them, while the role of the model lies in answering the questions, via algorithmic, thus logically simple, procedures. Within this framework, complexity is broadly related to the number of question-answer levels involved in the process and the nature of the questions posed. Addressing this complexity depends crucially on the ingenuity and creativity of the modeler. This may lead to a view of complexity, which is no more observer-independent, but rather accounts for both historical and cultural development, that is the context of the problem at hand.* © *2008 Wiley Periodicals, Inc. Complexity 13: 21–28, 2008*

FABIO BOSCHETTI, DAVID MCDONALD, AND RANDALL GRAY

Fabio Boschetti, David McDonald and Randall Gray are at CSIRO–CMAR, Wembley, WA, Australia. (e-mail: fabio.boschetti@csiro.au).

1. INTRODUCTION

A fundamental *concept* underpinning complex system science (CSS) is that local interactions between relatively simple components can lead to considerably more complex nonlocal behaviors. A fundamental *conjecture* CSS attempts to study is that these local interactions are the drivers for processes like self-organization and emergence and, in turn, are responsible for the immense variety of structures, patterns, and phenomena we see in Nature. These two ideas can be found, in slightly different form, in most "manifestos" of CSS, which, implicitly or explicitly, associate CSS with an attempt to go beyond the constraints of the reductionist framework, which has guided most past and current scientific successes.

A natural question arising from this observation is how much of the current scientific arsenal can be saved by giving up reductionism; is the scientific method as we know it today suitable to study processes like self-organization, emergence, and adaptation? In his extensive body of work, Rosen [1, 2] gives a negative answer to those questions, and a similar conjecture is proposed in Kauffman [3].

© 2008 Government of Australia. Exclusive worldwide publication rights in the article have been transferred to Wiley Periodicals, Inc., Vol. 13, No. 6
DOI 10.1002/cplx.20215
Published online 25 February 2008 in Wiley InterScience
(www.interscience.wiley.com)

Along those lines, in the present work, we discuss the use of computer modeling and we ask what it means to carry out CSS by computer modeling and what kind of complexity can we study with this approach. We believe these are important questions, given that so much work in CSS is done with the aid of computer modeling and since much of the development of CSS was made possible by the availability of fast computing.

Both our approach and aim are less general than those presented by Rosen and Kaufman. In particular, we discuss (a) if and how computer modeling differs when it is applied to a complex rather than to a noncomplex problem, (b) what is the role of the modeler in overall complex-problem solving, and (c) whether common measures of process and model complexity match human perception of the difficulty of solving a "complex" problem by employing a computer model.

We argue that, in practice, only a small fraction of a CSS problem is addressed via computation. Rather, the choice of the problem, its formulation, the selection of the numeral tools to address it, the interpretation of the results and the choice of how to act in response to such results are all steps that involve the creativity and ingenuity of CS researchers. We believe that the complexity of a modeling exercise cannot be estimated without accounting for this crucial contribution and we propose some initial steps in this direction.

Our target audience consists of modelers in a multidisciplinary sense; consequently we aim to discuss this subject in a fairly informal manner, trying to avoid technical terms and referring the interested reader to the ample available literature.

2. BACKGROUND CONCEPTS

In this section, we introduce some of the definitions, assumptions, and concepts we discuss in the rest of the study.

2.1. Definitions and Measures of Complexity

Complex system concepts like self-organization, emergence, and complexity itself, often have different meaning to different researchers, both within the same field and among different scientific fields. For communication to be possible, these concepts need to be defined, at least broadly. A large number of definitions have been proposed in the literature (see http://bruce.edmonds.name for an extensive collection). In this study, we refer mostly to a set of information-theoretic definitions proposed in Prokopenko et al. [4], where extensive references for further reading can also be found.

One of the most popular definitions of complexity is Kolmogorov's algorithmic complexity (also called Kolmogorov–Chaitin complexity, see Prokopenko et al. submitted and [5]). Given a data set, this is defined as the length (in bits of information) of the minimal program, which can reproduce the data. According to this definition, fully periodic data have low complexity since a very short program (which stores 1 period and outputs it indefinitely) can reproduce the entire data set exactly. Departures from periodic behavior towards randomness would require programs of increasing length and consequently display increasing algorithmic complexity. Because random data are by definition not predictable and consequently not compressible, they have maximum complexity according to Kolmogorov's definition: the only program that can reproduce it is a program which stores and outputs the data set itself.

This somehow contradicts our intuition about complexity, which is usually seen as something between order and randomness. To circumvent some of these problems, Crutchfield and Young [6] propose that complexity be characterized by the amount of information needed to perform *useful* "statistical" prediction. In other words, it measures the minimum number of statistically different configuration states we need to store in order to capture and reproduce the statistical properties of a time series, rather that the exact time series itself. As in the case of algorithmic complexity, little information is needed to capture the statistical properties of a simple periodic function. Unlike Kolmogorov's definition though, very little information is needed to statistically reproduce a random time series, since no amount of memory (effort) can help improving our predictive ability, i.e., an "optimal" prediction can be performed with zero memory (there is no point in storing the outcomes of roulette draws to bet on the next draw). Since these information-theoretic views of complexity are closely related to predictability and in particular to the amount of information required to achieve useful or optimal prediction, they can be applied to scientific modeling in a straightforward manner [7].

From an information-theoretic perspective, the information processing capabilities of an agent (be this a cell, a bacterium or a higher organism) are equivalent to a set of rules which allows the agent to take an action in response to an input signal or stimulus. These capabilities can be viewed as adaptations to the environment as a result of evolution or learning. In information-theoretical jargon, this is a set of instructions that is usually described as a "model" of the environment [8]. Thus, within this framework, model building is analogous to "understanding": a model displays and represents "understanding," since it allows for a prediction (a guess, or expectation) of what the environment will look like in the immediate future, according to which the agent can choose what action it should take next.

At the level of human knowledge, this model building may enable a scientist or engineer to predict the behavior of natural processes and to devise ways to modify them. According to this view, the concepts of information processing, computation, modeling and prediction are relevant to all levels

© 2008 Wiley Periodicals, Inc.
DOI 10.1002/cplx

of investigation of a complex system and encompass both the system and the observer trying to study it.

2.2. Formal Logic and Computation

The discussion that follows relies crucially on the equivalence between the workings of formal systems and computation [Prokopenko et al., submitted, 9–11]. Both start from a set of axioms and input data (defined a priori) and a set of inference rules (transformation rules and computer instructions). These generate outputs such as theorems and computational results.

As discussed in [12], in a formal system the truth of a statement is a property that depends only on the set of axioms and inference rules and the values assigned to its variables. If these are given, then every such statement (without unassigned variables) is inevitably true or false, and remains so for all possible scenarios and, consequently, confers no information about the real world. A typical example is the statement "I will get wet if I go outside," which may or may not be true depending on the weather conditions. If our colleague tells us that it is currently raining, then our variables for location and weather have been defined for us, and the statement is an inescapable consequence of the unalterable parts of the system: it may not correspond to reality if our colleague is wrong about the local weather, but it is true in the context of the system. If, on the other hand, we must peek out of the window to assign a value to the local weather variables (and we do so correctly), then our statement *does* give us information pertinent to the real world. Thus, it is the process of assigning values to the variables in a statement, and assessing whether the statement is true or not, which provides information about the real world. In contrast, a mathematical theorem is true independently of Nature's vagaries. Within an information-theoretic framework, from the definitions of algorithmic and statistical complexities we presented above, it thud follows that true statements are *transformations* of sentences containing information, not statements of new information.

The PCs on our desks are equivalent to a finite tape turing machine (TM), an abstract and general computational device commonly employed in theoretical computer science. Because the working of a TM is equivalent to that of a formal system [9], it follows that the running of any computer model *transforms* the information contained in the input via the coded algorithm, but does not *generate* information. Clearly, a model's output helps our finite mental capability to see consequences of what we code (which at times we cannot envisage), but its truth and relevance to the real world is limited to the truth and relevance of what the user codes and the input fed to the computation. No actual information about the real world is produced by a simulation. In a modeling exercise, information is thus generated solely by the writing of the code, the choice of the input *and the interpretation and comparison of the outputs to what we observe in Nature* which, in turn, tells us about the appropriateness of the rules we implement and the input we choose. This suggests that the modeler plays a crucial role in a CSS modeling exercise, which should be properly accounted for.

2.3. Forward and Inverse Modeling

Let's assume that we work with a model F of a physical process and we consider the model to be a black box. The process which, given a certain input i, determines an output o, ($o = F[i]$) represents our understanding of cause-effect relations. By changing the input i, we consequently change the output o; that is, we can control the output o, by acting on i, as we satisfy an operative definition of causality [13, 14], which allows us to timidly circumvent known philosophical problems. Since F respects our perception of the "arrow of time" (cause leads to an effect), it is usually called a "forward" model.

Many engineering and scientific problems however ask questions like "what is the cause of this effect?" ("what caused this flood?," "what causes this disease?," "what can stop the next pandemic?"). This implies a hypothetical model I which, given an effect (o) as input, gives a possible cause as an output ($i = I[o]$). Because I reverses the "arrow of time," it is usually called an inverse model.

Unfortunately, inverse models such as I can be written explicitly for only a very small set of forward models F. This is true not only for closed-form models but also for purely numerical models. As a result, most inverse engineering and scientific problems need to be solved by iterative methods using forward models in which sets of inputs i are tested until a reasonable match between $F[i]$ and the expected output, o, is found. The procedure, which allows us to recover i from F and o is called inversion, optimization, or regression, depending on the discipline, and we refer to it generally as inverse modeling.

3. COMPLEXITY, MODELING, AND UNDERSTANDING

We can summarize the previous sections as follows:

1. a computer model is equivalent to a formal system; that is, it is a closed system whose dynamics and evolution is fully determined by the set of acceptable initial conditions and transformation rules;
2. this ensures that there is an equivalence between a model and its output; in principle, an output can be compressed into the model, since we can reconstruct the output (or at least its statistical features) by simply running the model. This equivalence is what the definitions of algorithmic and statistical complexities are based upon;

3. employing the "Occam's razor" principle, this model represents the best "understanding" of the output/process since it is the most compact representation of the process, which generates the data and can be used for prediction (provided we have the "right" model, of course).

Ideally, we can thus apply these three steps to the working of a scientist. We can imagine:

1. measuring a natural process by collecting data arising from it;
2. reconstructing a model which is able to reproduce the measured data with a specified accuracy and then
3. "measuring" the size of the model; if the model is "large" the natural process is described as "complex," otherwise it is described as "simple."

Does this capture our intuition of the challenges faced by a scientist in addressing a complex problem? In our opinion, it does not. There are a number of reasons for our concern:

1. Consider some classes of models that have been extensively used in CSS research. Cellular automata are supposed to generate emergent patterns, sand-pile algorithms are supposed to generate self-organization, scale-free networks generate "small worlds" behaviors, and genetic algorithms, neural networks, ant colony algorithms and simulated annealing (among others) are supposed to provide learning and adaptation. Nevertheless, all of these models are very "small," containing only a few simple rules and, according to algorithmic and statistical complexity definitions, they would be classified as "simple." However, they have long been considered as stereotypical examples of "complex" models, and humans find it hard to predict the behavior of these "simple" models. This suggests that there is a mismatch between the information-theoretic concept of complexity and shortcomings in the ability of humans to "understand."
2. Most information-theoretic measures, which are based on statistical principles, require processes and data to be stationary. It is unclear that our understanding of self-organization, adaptation, and emergence are compatible with this assumption.
3. Information-theoretic measures of complexity also do not account for the time it may take for a scientist to solve problems. The concept of computation time as a measure of complexity is employed in the definition of "logical-depth," [15] roughly, the time which a TM would take to carry out a specific task. Once again, it is not obvious that the human perception of complexity matches this measure; a code may take a very long time to generate large amounts of easily-predictable data, while other fast computations may appear to be very complex (a chaotic process for example).

Most important for our discussion is that these measures do not account for the process, which an agent/scientist needs to employ in order to reconstruct the model; rather, they assume that the *shortest* model is already available. In fact, when dealing with algorithmic complexity, not only a method which *ensures* the reconstruction of the shortest model is unavailable, but also, given a model, we cannot even prove that such model is the shortest possible (Prokopenko et al., submitted). Regarding statistical complexity, a method to reconstruct the shortest model is available [16], but the resulting model is provably shortest only given the language we use to measure the data, represent the data and code the model. In Crutchfield [8], step improvements in the modeling capabilities of an agent are linked to the discovery of new, more powerful, languages. How such discovery might happen is not known, nor is it known whether such discovery is computable at all. It thus appear to us that the concept of complexity needs to account for the role of the modeler, not only because the modeler is necessary in reconstructing a model and interpreting it, but also because the model itself cannot be modeler-independent, since the modeler will determine the language used in the reconstruction.

Finally, it is not at all obvious that the information-theoretic interpretation of "understanding" (that is, model-building) applies to human understanding, or our perception of it. Among many definitions of the concept of "human understanding," a particularly insightful one is the "ability to translate," [17] which implies the ability to express and employ rules within different languages and contexts; so far, automated systems (algorithms) have proved particularly ineffective at tasks of this kind, which suggests that measures used to evaluate the complexity of algorithms may not be suitable to estimate the human perception of the complexity of a problem.

We attempt to address some of these concerns in the following sections.

4. STUDYING A COMPLEX SYSTEM VIA NUMERICAL MODELING

In this section, we consider a hypothetical complex system model (CSM) and draw some conclusions about the way it is commonly used in CSS research. To represent a stereotypical CSM, we imagine that our CSM is an agent-based model in which a relatively large set of agents interact with one another by using simple local rules. We also imagine that the purpose of the modeling exercise is to study the global pattern arising from the resulting dynamics, which, as is commonly assumed, results in self-organization and emergent behaviors.

Following Section 2.3, the CSM is a forward model, which takes the local rules and an initial configuration as input i and generates an emergent behavior and self-organized dynamics as output o, $o = \text{CSM}[i]$. Also, because the CSM is an algorithm, the discussion at Section 2.2 applies, and we infer that any dynamical behavior in o is implicitly determined by i and the CSM. Indeed, no matter how complex, large, apparently unpredictable and surprising o is, its entire information content can be compressed into i and the CSM (Prokopenko et al., submitted).

The discussion in Section 2.2 seems to rule out the possibility of novelty or creative processes solely via numerical modeling (see also [14, 17]). It also points out that, from a purely logical perspective, no difference exists between modeling a complex and a non complex system, as long as the modeling is carried out on some machine which is equivalent to a finite tape TM, as is the case for the computers we typically use.

It is also obvious from the previous discussion that the execution of a numerical model is equivalent to the running of a mechanical device (indeed, the very first computers were mechanical instruments), and it is normally accepted in the CSS community that a mechanical device, no matter how *complicated*, is not *complex*. This highlights a sort of contradiction: in trying to simulate a complex process, we resort to using a tool that is not commonly viewed as complex. This contradiction may be resolved by one or more of these statements:

1. either the natural processes we want to study are not complex, in which case our entire view of complexity or approach to CSS needs to be revised;
2. or our models will never simulate the very features that make natural processes complex; this also would imply a considerable revision of the way we carry out CSS;
3. or the roots of what we perceive as complex do not lay solely in the process itself, but also in the effort required by us to understanding it.

Some of the issues related to points 1 and 2 have been addressed in [14] and [7] and we refer the reader to that work and the references listed therein. Here we address point 3.

Consider the standard scenario under which our CSM could be used in CSS research. Supposedly, the scientist has some expectation of what kind of global behavior arising from a natural process he/she intends to model or which kind of local rules he/she wants to study. We imagine the following steps:

1. the CSM is written and some input i is chosen;
2. the CSM is run and the output o (an image, an animation, numerical data, etc.) is obtained;
3. the user analyses the output, often visually and subjectively (sometimes numerically), and expresses a judgement on whether the output matches the expectations (possibly a pattern seen in Nature which he/she wants to reproduce or a result of the local rules which he/she judges to be "realistic" or "interesting");
4. if the outcome is not satisfactory, or something unexpected in the output sparks new insights, the user may decide to change the input i or the rules in the CSM and go back to 2.

This process, the writing and "tuning" of the model, is actually an iterative inversion process as described in Section 2.3. The "non mechanical" steps are 1, 3, and 4, in which the user is required to use ingenuity and expertise to design or adjust the code, to interpret the results, to judge the output realism or accuracy (by analyzing its inner patterns) and to select further input for testing. Step 2, the running of the CSM, is the mechanical one, the one in which no information is generated, but only transformed according to pre-defined rules. It is also clear that the only "logically nontrivial" information we can recover from the exercise in steps 1–4 is the "inverse" deduction of the suitable input and the suitable transformation rules in the CSM, not the generation of the patterns in the output, since they contain no information which is not already contained in i and the CSM.

In principle, algorithms which would, at least superficially, carry out steps 3 and 4, are abundant in the numerical optimization literature, and algorithms which reconstruct a CSM at point 1 have been proposed in the machine learning literature. However, this simply obscures, rather than removes, the role of the modeler. First the algorithms to carry out steps 1, 3, and 4 still need to be developed by the modeler. Second, the choice of algorithms will constrain the solution set and thus their applicability; third, once written, they are cannot be modified by the algorithm themselves; once the user has realized that the current architecture does not model the process appropriately, the algorithm must be re-written/modified, a step which requires the modeler intervention. In other words, optimization algorithms (even the ones inspired by natural processes like genetic algorithms) are still algorithms and as such undergo the constraints discussed in Section 2.2.

We can summarize our discussion so far as follows:

1. information-theoretic measures of model complexity do not match our perception of what makes a natural processes complex;
2. "understanding" a natural process is an inverse process;
3. this inverse process cannot be carried out solely by an algorithm, rather it requires the modeler intervention.

In the following section, we propose an approach towards a definition of complexity that accounts for these points.

5. WHAT MAKES A MODELING EXERCISE "COMPLEX"?

Here, we shift the question of complexity from the *model* to the *modeling exercise*, which involves a model and a modeler who uses the model to answer specific questions.

As we discussed in Section 2.3, within the context of a numerical inversion, solving a inverse problem can be seen as a (possibly iterative) process of asking and answering questions. The role of the modeler lies in posing the questions and devising a procedure to answer them; both require intuition and creativity. Once this has been done, answering the question (if possible) is then a mechanical process which is carried out by the model.

Within this framework, we propose that *the complexity of a modeling exercise is represented by the number of levels at which questions are asked and answered*. Here, we describe what we mean by levels. At a high level, any modeling exercise aims to answer a broad question ("what factors lead to a market crash?," "what decision will improve the resilience of this ecosystem?," "what is the square root of 10?," "what is the optimal scheduling for this process?"). Some problems, which we define as "simple," can be solved (within required numerical precision[1]) algorithmically in a single call of a numerical routine ("what is the square root of 10?"). Solving this problem does not involve an iterative inverse problem once the question is posed and a method to answer it (the algorithm) has been devised. We say this problem has level 1′ complexity.

Other problems cannot be solved in a single step and require a numerical iterative search of a parameter space via the use of a forward model. This search involves questions and answers at 2 levels. Let's consider the problem "what is the optimal scheduling for this process?" The high level question is given by the problem itself. The lower level questions are given by the individual runs of the forward model, which, given a candidate input, check whether the resulting schedule is optimal; each run of a forward model effectively answers the question "is the scheduling arising from this input parameter set good?," thus the modeler must devise a clever numerical procedure to judge how "good" a certain schedule is, given our expectations. We say this problem has level 2′ complexity.

Unlike in level 1′, in a level 2′ problem the modeler has a further role. First, he/she must pose the "global" question. Second, he/she must pose the lower level questions in the iterative search. Traditionally, this is done by the modeler employing his/her judgment to evaluate which configuration in the parameter space should be tested next. Alternatively, it can be done automatically, by using a numerical optimization routine [18], in which case the role of the user is to choose the suitable optimization routine and tune it for the problem at hand (which can be seen as an inverse problem itself).

Next, we find inverse problems for which the quality of the outcome cannot be judged numerically [19]. Artistic problems fall in this category [20], but so do many scientific and engineering problems [21]. An example of such questions may be "does this agent-based model generate an emergence pattern?" or "does it display self-organization?." Here, in addition to the questions mentioned in the previous paragraph, the user needs to judge (visually, for example) the global patterns arising from the forward model; expertise, experience, creativity and imagination in this further step are required to evaluate patterns, since *novelty and surprise are possible outcomes of this process*, and to possible develop instruments and/or procedures capable of carrying out such assessment. We say this problem has level 3′ complexity.

Finally, we consider problems like "what factors result in a market crash?" and "what decision will improve the resilience of this ecosystem?." In these problems, the modeler is required to carry out another crucial judgment in answering another level of questions like "is the numerical model I am using appropriate for this task?" "does it include all important processes which affect my results?." If the answer to these questions is negative, the modeler needs to intervene by modifying or augmenting the model itself. It should be clear than neither of these tasks (judging the appropriateness of the model and improving it) can be carried out by the model itself with current technologies [14] and both involve a creative intervention from the modeler. We say this problem has level 4′ complexity.

We summarize this proposed classification in Table 1.

6. DISCUSSION AND IMPLICATIONS

The classification proposed in the previous section is neither unique, nor observer independent; the same process will appear of different complexity to different scientists, according to their own expertise. Also, process complexity may change in time. Let's consider the "simple," complexity level 1, "$\sqrt{10}$?" problem. In our discussion we assumed this can be solved by a simple routine call. This is correct only provided we do not concern ourselves with what the routine does. In reality, a square root is itself calculated with an iterative procedure, albeit one which can be coded fully algorithmically, given a limited required accuracy [22]. Someone developed the algorithm,[2] using his/her own creativity and ingenuity. So is it really correct to

[1] *This requirement is essential, or we must admit that any problem leading to an irrational number would be unsolvable numerically.*

[2] *Actually several algorithms to possibly calculate a square root exist, see for example [22].*

TABLE 1

Classification Levels of Problem Complexity and Some Examples

Complexity Level	Type of Problems	Example
Level 1	Problems which can be solved by a single call of a routine/model	• $\sqrt{10}$? • "is file.dat stored in my computer?" • "what is the temperature today?"
Level 2	Inverse problems for which a "suitable" forward model is available and which can be automated by the choice of an appropriate optimization routine	• "what is the optimal scheduling for this process?" • "what aerodynamic design for this wing is optimal?" • "what type of fuel can make this engine more efficient?"
Level 3	Inverse problems for which a "suitable" forward model is available but cannot be automated by the choice of an appropriate optimization routine	• "does this agent-based model generate an emergence pattern?" • "does it display self-organization?" • "does this program generate nice pictures/music?"
Level 4	Inverse problems for which we are not sure a "suitable" forward model is available and in which the model may need to be modified	• "what factors result in a market crash?" • "what decision will improve the resilience of this ecosystem?" • "what is the best policy to address global warming?"

assign to it a level 1′ complexity? A similar consideration applies to the scheduling problem discussed above; from a purely computational perspective a code may contain both a forward complex model and a numerical inverse routine, thereby performing an iterative search in a way that may be transparent to the user. In both cases, an originally complex modeling task can be turned into a "simple" black-box operation, if we do not concern ourselves with what the code actually does. As a further example, Kaltwasser et al. [23] attempted to reproduce algorithmically the subjective evaluation of problems for which a numerical cost function could not be easily designed; in principle, this would also turn the iterative, subjective problem at level 3′ into a level 1′, "simple" problem.

In our opinion, this captures the process of cultural development and scientific understanding: the ability to "de-complexify" a process turns a previously "complex" problem into a "simpler" one, thereby climbing down the complexity ladder. This can be achieved either by a single individual or by the scientific community as a whole. According to this view, the $\sqrt{10}$? problem, which previously require lengthy calculations, now can be solved by pressing a button, without any need to concern ourselves with what the algorithm does; for us, today, $\sqrt{10}$? is not complex; at least is simpler than it was for previous generations.

This view raises the issue of how this understanding or "de-complexification" is transmitted culturally. In the information-theoretic formulations we discussed above, "understanding" implies ability to predict. The model, which can be used to carry out the prediction, represents "understanding." As we discussed above, we do not believe that ability to predict implies *ultimate* understanding, however we subscribe to the view that the ability to predict is a minimum requirement to show understanding. From this perspective, a scientist who has devised a model that predicts a process with a required precision can be said to have achieved a satisfactory understating of a process. More generally, we may say that another scientist who adopts such model has at least understood that the use of the model leads to prediction. This is perhaps a little too loose, we admit, but the difference between the model developer and the user may be dismissed for "simple" level 1 and 2 problems.

The identification of "understanding" with a model becomes weaker the higher we climb the "complexity" ladder, however. Although the algorithm which solves $\sqrt{10}$? can solve all other square root questions (for most practical purposes), this becomes less and less true for more complex problems. An ecological model built to study an Australian marine ecosystem, may not be suited to the study of an Antarctic marine ecosystem without suitable modification, and even less to study nonmarine ecosystems. The model will then require intervention and modification; to what extent this modification is possible, and to what extent this is helped by the existence of a starting model (which may have been built by generations of previous modelers) is not clear and may be an avenue for further research.

7. CONCLUSIONS

Most real-world problems involving computation need to be solved via an inverse process, involving iterative trial-and-error runs of a forward model. This is not different from how most CSS problems are addressed, in which a modeler tests several input parameters and model architectures in order to generate a desired global outcome. Disregarding this observation, thereby focusing only on the computer model, leads to ignoring the role of the modeler in addressing a complex problem.

We propose switching the focus from the analysis of a model to the analysis of a modeling exercise and we argue that most of the complexity lies in the task the modeler must carry out, rather than merely in the computation involved.

We suggest that the complexity of a modeling task is related to the level and amount of intervention, in terms of ingenuity and creativity, which is required by the modeler, and that the content of the model represents the amount of complex knowledge which is transmitted within the scientific community.

ACKNOWLEDGMENTS

This research was carried out as a part of the CSIRO Emergence Interaction Task, http://www.per.marine.csiro.au/staff/Fabio.Boschetti/CSS emergence.htm. The authors wish to thank Dr. Nicky Griggs and Dr. Francis Pantus for useful comments on an early manuscript.

REFERENCES

1. Rosen, R. Essays on Life Itself; Columbia University Press: New York, 2000.
2. Rosen, R. Life Itself, a Comprehensive Inquiry into the Nature, Origin, and Fabrication of Life; Columbia University Press, 2001.
3. Kauffman, S. Investigations; Oxford University Press, 2000.
4. Chaitin, G. The Limits of Mathematics: A Course on Information Theory and Limits of Formal Reasoning; Springer: New York, 1997.
5. Chaitin, G.J. Information-theoretic limitations of formal systems. J ACM 1974, 21, 403–424.
6. Crutchfield, J.P.; Young, K. Inferring statistical complexity. Physical Rev Lett 1989, 63, 105–108.
7. Boschetti, F. Mapping the complexity of ecological models. Ecological Complexity, in press.
8. Crutchfield, J.P. The calculi of emergence: Computation, dynamics, and induction. Physica D 1994, 75, 11–54.
9. Turing, M.A. On computable numbers, with an application to the Entscheidungs problem. Proc Lond Mathematical Soc 1936, 42, 230–265.
10. Ord, T. Hypercomputation: Computing more than the turing machine. CoRR math.LO/0209332 (2002).
11. Penrose, R. The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics; Vintage: London, Melbourne, 1989.
12. Boschetti, F.; Gray, R. Emergence and computability. Emergence: Complexity and Organization 2007, 9, 120–130.
13. Pattee, H.H. Causation, control, and the evolution of complexity. In: Downward Causation; Anderson, P.B.; Christiansen, P.V.; Emmeche, C.; Finnemann, N.O, in press.
14. Pearl, J. Causality: Models, Reasoning and Inference; Cambridge, Mass: MIT Press, 2000; 384 p.
15. Bennett, C.H. Logical depth and physical complexity. In: The Universal Turing Machine, A Half-Century Survey; Herken, R.; Eds. Oxford University Press: Oxford, 1988; pp 227–257.
16. Shalizi, C.R.; Shalizi, K.L. Blind construction of optimal nonlinear recursive predictors for discrete sequences. In: Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference; Chickering, M.; Halpern, J., Eds.; AUAI Press: Arlington, Virginia. 2004; pp 504–511.
17. Hübler, A. Understanding complex systems: Defining an abstract Concept. Complexity 2007, 12, 5–9.
18. Tarantola, A. Inverse Problem Theory; Elsevier: Amsterdam, 1987; 613 p.
19. Takagi, H. Interactive evolutionary computation: Fusion of the capacities of EC: Optimization and human evaluation. Proc IEEE 2001, 89, 1275–1296.
20. Baluja, S.; Pomerleau, D.; Jochem, T. Towards automated artificial evolution for computer-generated images. Connection Sci 1994, 6, 325–354.
21. Boschetti, F.; Moresi, L. Interactive inversion in geosciences. Geophysics 2001, 64, 1226–1235.
22. Weisstein, E.W. Square Root. From MathWorld–A Wolfram Web Resource. Available at http://mathworld.wolfram.com/SquareRoot.html
23. Kaltwasser, P.; Boschetti, F.; Hornby, P. Measure of similarity between geological sections accounting for subjective criteria. Computer Geosci 2004, 31/1, 29–34.

DOI 10.1002/cplx