Evolutionary Economics and Social Complexity Science 27

Stanislaw Raczynski

Catastrophes and Unexpected Behavior Patterns in Complex Artificial Populations

Springer

Evolutionary Economics and Social Complexity Science

Volume 27

More information about this series at http://www.springer.com/series/11930

Stanislaw Raczynski

Catastrophes and Unexpected Behavior Patterns in Complex Artificial Populations

Springer

Stanislaw Raczynski
Faculty of Engineering
Universidad Panamericana
Mexico City, Distrito Federal, Mexico

ISSN 2198-4204 ISSN 2198-4212 (electronic)
Evolutionary Economics and Social Complexity Science
ISBN 978-981-16-2573-2 ISBN 978-981-16-2574-9 (eBook)
https://doi.org/10.1007/978-981-16-2574-9

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

Computer simulation is an important research tool for more than seven decades, and evolves together with the evolution of computer hardware. At the very beginning, the simulations have been used by some "gurus" and specialists. Then, more user-friendly software tools have been developed. In recent years, there are so many software packages and simulation languages, that nearly everybody can simulate anything. However, more professional simulation tasks cannot be separated from model building. To create valid models, the researcher must dominate several disciplines, like mathematics, physics, computer science, and others. Modeling and simulation require from the researcher certain interdisciplinary and generalized insight on complex systems to be modeled.

This book is a continuation of the research described in the book "Interacting Complexities of Herds and Social Organizations" (Raczynski S., Springer Nature, 2019). However, here we concentrate on the phenomena of catastrophes, unexpected, and spontaneous events that are neither explicitly coded in the model nor provoked by external impulses. A short reference to the known theory of catastrophes is done. The classic examples of catastrophes in that context refer rather to mathematical descriptions of events that occur while solving differential equations. An example of such catastrophe is described in Chap. 1. In this book we understand "catastrophes" in more general sense, as events that appear in models of complex populations. As for the complexity, there is not completely clear when a model is complex or not. Complexity should not be confused with the system size. Huge systems can be quite simple, and some small ones more complex. MacMillan dictionary defines complexity as "features of something that make it confusing or difficult to deal with." The Oxford Learner's Dictionary says that the complexity is "the state of being formed of many parts; the state of being difficult to understand." There are many similar definitions. In a book of Meyers "Mathematics of Complexity and Dynamical Systems" (Springer, 2011), we can read "Complex systems are systems that comprise many interacting parts with the ability to generate a new quality of collective behavior through self-organization, e.g. the spontaneous formation of temporal, spatial, or functional structures." The main topic of the present book is somewhat related to the last definition, namely to spontaneous formation of structures and generation of macro-behavior patterns. Comparing to populations of individuals in the real world, our models are rather simple. However, from the point of view of model building, they are not trivial and generate, like the complex systems, structures and interesting behavior patterns.

In Chap. 1, there are some remarks on the classic mathematical approach to catastrophes, and the overview of agent-based modeling. Chapter 2 is dedicated to the simulation package BLUESSS, used in the rest of the book. In Chap. 3, we discuss the model of an artificial society where the individuals consume resources, move and interact with each other. The simulations show unexpected events that appear spontaneously in the model, when it seems to reach a steady state. Chapter 4 deals with an extended prey–predator system of multiple species, where the model dynamics is quite different from the classic Lotka–Volterra continuous model. Chapter 5 contains simulations of possible catastrophic events in a stock market model. Chapter 6 describes a model of epidemics that is itself a kind of catastrophe.

Here, I would like to express some personal view on epidemic models. First of all, it should be emphasized that the presented model is *not* any attempt to simulate the recent epidemy of covid-19. As stated in Chap. 6, we should appreciate all effort done to simulate this epidemy, related in numerous publications. However, we should remember that one of the most serious errors in scientific research is to look for something that does not exist. To construct a simulation model, first we should know if the model exists at all. Many researchers forget this stage of model building. This may lead to wrong and confused results. In the case such as covid-19, this is extremely important because an invalid model can be used to take wrong strategical decisions about epidemy treatment and provoke millions of deaths. It is important to remember that dealing with new kind of disease with unknown dynamics, we cannot just modify any existing epidemics model, and use known modeling paradigms. Anyway, this is also a question of ethics in computer simulation profession.

In Chap. 7, you can find a model of a growing organism that evolves and may reveal some degeneration like a cancer in a living body. There are some remarks on the DNA and RNA mechanism in living organisms. However, we do not pretend to simulate the extremely complex DNA functions. Instead, we use the abbreviation EOB to denote the genetic information included in each model agent (living cell). Further research on this kind of models could give us hints to better understand the growth of cells in living organisms.

Chapter 8 contains the description of a model of a population of agents that are working and obtain salaries. The work effort increases the common pool of goods in the society. The salary may depend on the amount and quality of work done by the agent. This may result in differences and inequality between agents. The amount of inequality is measured using the Gini inequality index. The increased income motivates the agent to intensify his work. This is a feedback that influences the inequality, the level of common goods and welfare. The dynamics of the population is simulated.

In Chap. 9, we simulate some examples of populations with waiting lines. It is pointed out that certain queuing models may reveal a considerable “catastrophic” and irregular growth of queues.

Finally, in Chap. 10, there are some remarks about possible simultaneous events. This is not a simulation of big populations, but rather of some special cases when the simultaneous events in the model may lead to ambiguity and wrong results. The general conclusion of this chapter is that the discrete-event models form a singularity in the space of models. An alternative paradigm named *semi-discrete events* is proposed.

Mexico City, Mexico Stanislaw Raczynski

Acknowledgements

An updated version of my article published earlier has been used in Chap. 10 of this book, as listed below.

"Simultaneous events, singularity in the space of models and chicken game," *International Journal of Modeling, Simulation, and Scientific Computing*, World Scientific Publishing Company, online-ready article, May 2021.

Contents

1 Catastrophes and Agent-Based Models 1
1.1 Catastrophes 1
1.2 Discrete Events and Agent-Based Models 4
1.2.1 Discrete Events and DEVS 5
1.2.2 Some Software Tools 6
1.2.3 The Event Queue 7
1.2.4 Agent-Based Models 9
References 10

2 BLUESSS Simulation Package 15
2.1 BLUESSS Concepts 15
2.2 BLUESSS Code, Program Structure 17
2.3 Inheritance 18
2.4 BLUESSS Code Generators, Continuous and Discrete Simulation Support 22
2.5 Variance Analysis 27
References 28

3 Behavior Patterns of an Artificial Society 29
3.1 Introduction 29
3.2 Agent-Based Models 31
3.3 Simulation Tool 32
3.4 The Model 33
3.4.1 Process *food* 33
3.4.2 Process *agent* 33
3.5 Simulations 37
3.5.1 Experiment 1 39
3.5.2 Experiment 2 (unexpected behavior) 41
3.5.3 Experiment 3—aggression 43

3.5.4 Experiment 4—Work 45
3.6 Conclusions 48
References 48

4 Extended Prey–Predator Model 51
4.1 Introduction 51
4.2 Continuous Model 53
4.2.1 Simple Simulation 53
4.2.2 Uncertainty and Differential Inclusions 55
4.3 Agent-Based Simulation 58
4.3.1 General Remarks 58
4.3.2 Simulation Tool 59
4.4 The Model 60
4.4.1 Resources and Agent Types 60
4.4.2 Implementation, Processes, and Events 61
4.4.3 The Food Resource 61
4.4.4 Agents 62
4.4.5 Process Static 64
4.4.6 Process Control 65
4.5 Simulations 65
4.5.1 Basic Simulation Mode 67
4.5.2 Incoming Agents, Low System Isolation 69
4.5.3 Gregarious Effect 69
4.5.4 Group Forming—Simulation Mode 2 69
4.5.5 A Slow Catastrophe; Static Agents 71
Appendix—Model Data Specification 74
The Food Resource 75
Agents 75
Static Agents (SA) Mode 76
References 76

5 Stock Market: Uncertainty and Catastrophes 79
5.1 Introduction 79
5.2 Continuous Model 80
5.2.1 Model Equations 81
5.2.2 Differential Inclusion Solver 84
5.2.3 The Reachable Set Example 85
5.3 Agent-Based Model 87
5.3.1 Processes and Events 89
5.3.2 Other Elements 92
5.4 Simulation Tool: BLUESSS Implementation 93
5.5 The Simulations 94
5.5.1 Experiment 1 95
5.5.2 Experiment 2 96
5.5.3 Experiment 3 98

5.6 Final Remarks . . . 100
References . . . 100

6 Epidemics . . . 103
6.1 Introduction . . . 103
6.2 Continuous Models . . . 104
6.2.1 Susceptible-Infectious-Removed Models . . . 104
6.2.2 Differential Inclusions and Uncertainty . . . 107
6.2.3 Examples of Reachable Sets . . . 107
6.3 Agent-Based Model . . . 112
6.3.1 General Concepts . . . 112
6.3.2 The Model . . . 113
6.4 Simulations . . . 115
6.4.1 Fast Propagation . . . 115
6.4.2 Lower Trip Frequency . . . 116
6.4.3 Near Cities . . . 118
6.4.4 Long Epidemics—Adverse Conditions . . . 120
6.5 Conclusion . . . 121
References . . . 121

7 Growing Organism and Cancer . . . 123
7.1 Introduction . . . 123
7.2 Agent-Based Modeling . . . 124
7.2.1 How It Works . . . 124
7.2.2 Simulation Tool . . . 125
7.3 The Model . . . 126
7.3.1 Overview . . . 126
7.3.2 Implementation . . . 129
7.4 Simulations . . . 132
7.4.1 The Growth of the Organism and Aging . . . 132
7.4.2 Infection and Defense . . . 132
7.4.3 The Cancer . . . 134
7.5 Conclusion . . . 135
References . . . 136

8 Work, Salary, and Gini . . . 139
8.1 Introduction . . . 139
8.2 Inequality and the Gini Coefficient . . . 141
8.3 Agent-Based Modeling . . . 142
8.4 Simulation Tool . . . 143
8.5 The Model . . . 144
8.5.1 Work, Income, and Object Function . . . 144
8.5.2 Implementation . . . 146
8.5.3 Simulations . . . 147
References . . . 152

9 Waiting Lines 155
9.1 Introduction 155
9.2 Queuing Model Generator (QMG) 157
9.2.1 Overview 157
9.2.2 QMG Blocks 157
9.2.3 Additional Entity Actions: The SVOP Function 160
9.3 Simulations 162
9.3.1 Simple Model, Useless Statistics 162
9.3.2 The Bus Stop Paradox 164
9.3.3 Queue and Server Chain 165
9.3.4 Conveyors with Feedback 168
9.4 Conclusion 169
References 171

10 Simultaneous Events, Semi-Discrete Events, and Chicken 173
10.1 Introduction 173
10.2 The Chicken Game 175
10.3 Simulation and Model Convergence 177
10.4 Three Body Collision 177
10.4.1 Compliance Collision 179
10.4.2 Elastic Collision: Compliance Zero 180
10.5 Conclusion 181
References 181

Index 183

About the Author

Stanislaw Raczynski received his master's degree from the Academy of Mining and Metallurgy (AGH) in Krakow, Poland, his Ph.D. and habilitation degree from the same academy, in the area of control theory and optimization methods. He was the head of the Computer Center of the AGH and of the Systems Analysis Group at the AGH. Dr. Raczynski worked as a researcher in the International Research Group in Moscow, USSR and participated in the activities of the European Workshop on Industrial Computer Systems. He was a visiting professor at the National University of Mexico and then a professor at the Panamericana University in Mexico City. Dr. Raczynski also has been the International Director of the Society for Computer Simulation. He wrote three books on computer simulation (Wiley UK, BrownWalker Press, and Springer Nature) and has more than 140 articles and papers published in professional journals and conference proceedings.

Chapter 1
Catastrophes and Agent-Based Models

Abstract Some general remarks on the term *catastrophe* are presented. A short recall of the concept of catastrophes in mathematics is given and examples are provided. The catastrophes discussed in this book are understood in some more general terms, quite different from the mathematical point of view. We rather discuss unexpected events that occur during computer simulation and that have not been explicitly inserted in the model algorithm and code. The main modeling tool used here is the agent-based modeling. An overview of this methodology and related software is provided in this chapter.

Keywords Catastrophe theory · Discrete-event simulation · DEVS · Agent-based modeling

1.1 Catastrophes

The Merriam-Webster dictionary defines *catastrophe* as "A momentous tragic event ranging from extreme misfortune to utter overthrow or ruin." Other meanings are defined as "A violent and sudden change in a feature of the earth" or "the final event of the dramatic action especially of a tragedy." The word *catastrophe* is also used in somewhat more general sense, denoting a sudden change in the behavior of an object under investigation, without the "tragic" attribute. Thus, we can consider of being catastrophe an earthquake, as well as a discontinuity in the trajectory of a computer-simulated dynamic system. This last meaning of catastrophe has been adopted by mathematicians to denote strange or irregular behavior of mathematical objects. This meaning was the subject of the theory of catastrophes developed by Thom (1975) and Zeeman (1976). In mathematics, catastrophes are closely connected to the *bifurcation* phenomenon when a small change of a model parameter may cause a sudden qualitative change in its behavior, see Luo (1997).

A good example in the theory of catastrophes is the cusp catastrophe. Consider the following function:

S. Raczynski, *Catastrophes and Unexpected Behavior Patterns in Complex Artificial Populations*, Evolutionary Economics and Social Complexity Science 27,
https://doi.org/10.1007/978-981-16-2574-9_1

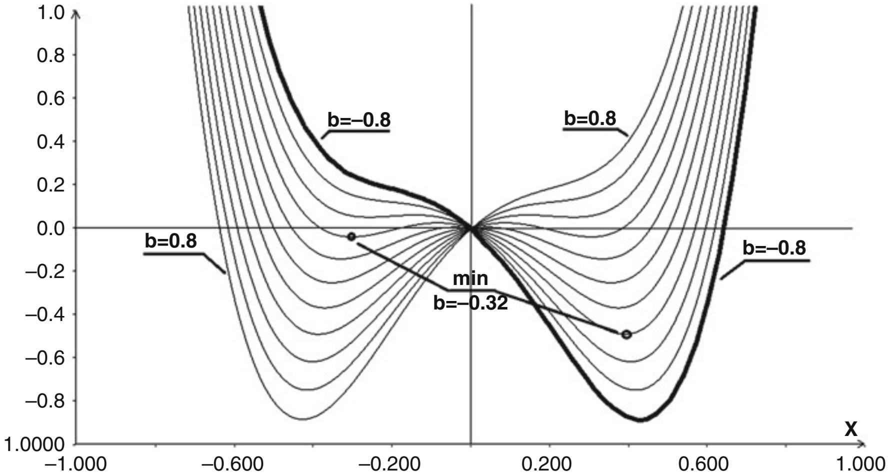

Fig. 1.1 Cusp catastrophe; Plots of function *V(x)* for different values of *b*

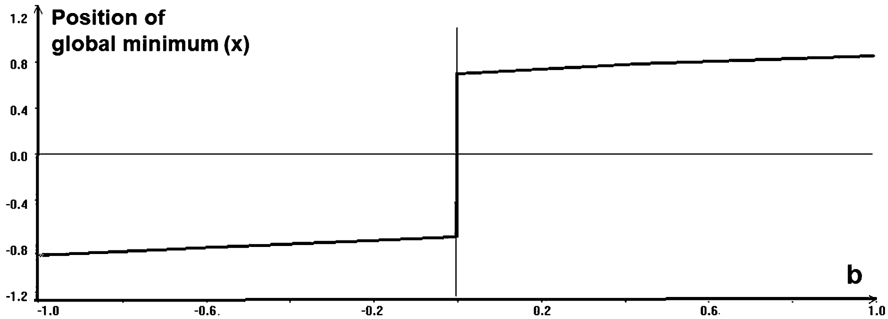

Fig. 1.2 Position *P(x)* of the global minimum of *V(x)* as function of parameter *b*

$$V(x) = x^4 + ax^2 + bx \tag{1.1}$$

Figure 1.1 shows the graphs of this function for $a = -1$, b changing from -0.83 to 0.83, and $-1 < x > 1$. The graph of the function is continuous, and its shape changes continuously with respect to a and b. It might appear that the other properties of *V(x)* should also change continuously. However, this is not true.

Consider two properties: (a) The number of local minima and (b) The position of the global minimum with respect to the variable x. These two properties change abruptly when b changes. Observe that the curve for $b = -0.8$ has one local (and global) minimum, while the curve for $b = -0.32$ has two local minima. The position of the global minimum changes, with discontinuity at $b = 0$, as shown as the function *P(x),* in Fig. 1.2.

Now, if we use the function *V(x)* as a part of a mathematical model of a dynamic system, we may observe a catastrophic change of model behavior like the loss of stability, for even very small changes of parameter *b* oscillating around zero. For example, if *V(x)* is used as a potential function in the equation

$$\frac{dx}{dt} = -\frac{dV(x)}{dx}$$

then the solution can become ambiguous or unstable. Such catastrophes may also occur in ODE models of heartbeat and nerve impulse, or in the behavior of stock exchange market.

The catastrophes in social systems are frequently related to conflict and aggression. In Kadar et al. (2019) we can find a good review of this approach. The authors describe an analytic model that includes violations of the moral order, principles and ideologies, breaches of moral norms, conflict, and aggression. The model reflects "the relationship between various methodologies through which one can examine morality and the moral order in the context of language conflict and aggression."

In multiple publications of Konrad Zacharias Lorenz (for example, Lorenz (1964, 1981, 2002), we can find an exhaustive source of works about aggression, social system behavior and related catastrophes, known as Lorenzian Theory of Catastrophes. Lorenz is recognized as the founder of ethology science. He asserts that "centrally coordinated sequences of endogenously generated impulses, have evolved phylogenetically and are very resistant to any individual modification. A fixed motor pattern's most important distinction from motor patterns that are not fixed and, simultaneously, the cogent argument for its being genetically programmed, consists in its taxonomic distribution." The Lorenzian theory can be used in advanced agent-based models of social behavior. The catastrophes result from the appearance of aggression in the groups of individuals. Lorenz provides the bases to create models of catastrophes of these types that perhaps could be used to predict future catastrophic events (Weaver 1980).

The aggression in the Lorenzian theory can be treated as a kind of catastrophe. For more results on this approach to catastrophes see Kim (1976) and Baron (1977). For newer works on aggression simulation, see, for example, Danaf et al. (2015).

Computer simulation of aggression is not new. Coe (1964) presents a report on computer simulation of aggression simulated on IBM 650 computer, machine with rotating drum memory of up to 4000 words. The simulation programs running on such machines have been already quite complex. Of course, the huge exponential growth of hardware in present days makes it possible to carry out greater and more sophisticated simulations. However, the complexity and capability of the software depend not only on the power of used hardware. Simulation programs that ran on IBM650 or ICL1900 in the sixties were not trivial.

We do not intent to provide a strict definition of *catastrophe*, different from these mentioned above. Roughly speaking, we denote as "catastrophic" events that occur in the model behavior that can abruptly change the model state or the behavioral pattern, like the presence and magnitude of oscillations, if any. This includes also other, relevant and unexpected events along the model trajectory. By "unexpected" we mean events that occur spontaneously and which were not explicitly, or in other, hidden way, implanted into the model while it was created.

There are many other examples of mathematical catastrophes provided in the literature. However, this, classic mathematical catastrophe theory is not the topic of the present book. Here, we use the discrete-event, agent-based models, mainly the models of artificial populations of moving objects, instead the models given by ordinary (ODE) or partial differential equations. The ODE models appear in the methodology of system dynamics (SD-models) that has been widely used and applied in modeling of a variety of real systems, during the last six decades (Forrester 1958). Our point is that not everything that happens in the real world can be modeled using the ODE or SD methodology. Thus, we rather use the agent-based models, where all the necessary model specification is reduced to a logical description of object behavior and simple arithmetic, with no mathematics at all. In the post-mortem model analysis, the only mathematics is the eventual use of statistics to analyze the results.

1.2 Discrete Events and Agent-Based Models

Looking at the annals of the development of computer simulation methodology, we can see a clear division between continuous and discrete modeling and simulation. As for the continuous simulation (Raczynski 2003) observe that the only machines that supported such kind of simulation were the analog computers of 1950s. In a digital computer nothing is continuous. So, any numerical method implemented on digital computer provides an approximation to the real continuous process. Of course, such methods have reached a high level of sophistication, and permit the simulation of continuous systems with a satisfactory accuracy.

The models used in continuous simulation are based on ordinary or partial differential equations. If the model time is discrete and the model state is a real variable, difference equations are used. The system dynamics methodology (SD) uses the ordinary (ODE) or difference equation models (Forrester 1958).

In the discrete simulation, both the model time and model state "jump" from one value to another in, theoretically, time interval zero. Some of the recent simulation tools also support combined simulation (see BLUESSS package, Chap. 2 of this book). BLUESSS is an object-oriented simulation package that carried out discrete-event simulation, but also supports continuous models given in form of the ODEs, Bond Graphs, or Signal Flow Graphs.

In Raczynski (2009, 2019, see also Chap. 9) you can find a description of a new simulation tool that uses the graphical user interface almost identical to that used by the system dynamics packages. This interface is used to define the model. However, behind the interface, the simulation is carried out using discrete-event method. This discrete simulation machine is transparent to the user and runs automatically.

1.2.1 Discrete Events and DEVS

By the *model time* we understand the time variable that is controlled by the simulation program during the simulation run. The *real time* represents the time of our (or computer) physical clock. For example, simulating the movement of a galaxy, we simulate several millions of model time years. On a fast computer, his simulation may take several minutes in the real time.

There are many real systems, where we can define *events* that consist in changing the state of the system. For example, the events may describe the start or the end of a service process, a birth or death of a model entity, or taking place in a waiting line. In many situations, such events can be considered to be executed in a very small interval of time, compared to the total length of model simulation time. The discrete-event simulation means that we suppose that the model events are discrete, i.e., they are accomplished within model time interval of length zero. This model simplification makes the simulations very fast.

In the *object-oriented* programming we declare several generic code segments called classes. According to these declarations, objects are created at the runtime. Each object is equipped with a data set, and several *methods* that perform operations on the data.

The Discrete Event Specification (DEVS) formalism is used to describe models in discrete-event simulation. In the DEVS formalism, an "atomic" model M is defined as follows (Zeigler 1987).

$$\begin{aligned} &M = < X, S, Y, \sigma_{\text{int}}, \sigma_{\text{ext}}, \lambda, \tau > \\ &\sigma_{\text{int}} : S \to S, \sigma_{\text{ext}} : Q \times S \to S, \lambda : Q \to Y, \end{aligned} \tag{1.2}$$

where X is the input space, S is the system state space, Y is the output space, σ_{int} is the internal state transition function, σ_{ext} is the external transition function, and Q is the "total state."

Atomic models can be coupled to form a coupled model. The coupled models can also be coupled in hierarchical way, to form more complex models. The coupled DEVS model is as follows.

$$\text{coupled DEVS} \equiv < X_{\text{self}}, Y_{\text{self}}, D, \{M_i\}, \{I_i\}, \{Z_{i,j}\}, \text{select} >$$

The sub-index *self* denotes the coupled model itself. D is a set of unique component references. The set of components is

$$\{M_i | i \in D\}$$

I_{i} is the set of influencees for I, $Z_{\text{i,j}}$ is the i-to-j output translation function. The *select* component defines the order of execution for simultaneous events that may occur in the coupled model. This component must be added to the model to avoid ambiguities in the simulation algorithm and to make the model implementation