# Simulation-Based Data Acquisition

Fabian Lorig and Ingo J. Timm

**Abstract** In data science, the application of most approaches requires the existence of big data from a real-world system. Due to access limitations, non-existence of the system, or temporal as well as economic restrictions, such data might not be accessible or available. To overcome a lack of real-world data, this chapter introduces simulation-based data acquisition as method for the generation of artificial data that serves as a substitute when applying data science techniques. Instead of gathering data from the real-world system, computer simulation is used to model and execute artificial systems that can provide a more accessible, economic, and robust source of big data. To this end, it is outlined how data science can benefit from simulation and vice versa. Specific approaches are introduced for the design and execution of experiments and a selection of simulation frameworks is presented that facilitates the conducting of simulation studies for novice and professional users.

## 1 Introduction

Most data science approaches rely on the existence of big data that is acquired and extracted from real-world systems for further processing. However, for some analyses or investigations, real data might not be available. Potential reasons are, for instance, accessibility to or existence of the system of interest such that data cannot be acquired. Other possible restrictions are economical or time limitations that do not allow for the efficient extraction of required data. In other disciplines, similar

Fabian Lorig*
Malmö University, Department of Computer Science and Media Technology,
Internet of Things and People Research Center (IoTaP),
Nordenskiöldsgatan 1, 211 19 Malmö, Sweden, e-mail: fabian.lorig@mau.se

Ingo J. Timm
Trier University, Center for Informatics Research and Technology,
Behringstrasse 21, 54296 Trier, Germany, e-mail: itimm@uni-trier.de

challenges are addressed using computer simulation. Here, artificial systems serve as a substitute for real-world systems, which enable a more efficient, viable, and unlimited generation of synthetic data instead.

In many disciplines and domains, scientific advance increasingly relies on the application of simulation. It is used for the generation and validation as well as for the illustration and imparting of knowledge. To this end, simulation can be applied both as a scientific method in terms of simulation studies or as a practical tool for educational purposes [37]. Either way, individual models are required, which are configured and executed with respect to a specific purpose. In many fields of application, simulation models exist for different purposes and are often provided in domain-specific repositories, e.g., OpenABM [14] or CellML [19]. In addition, numerous simulation frameworks exist for different modeling paradigms, that facilitate the creation of new models, e.g., [9]. Many of these frameworks do not require advanced technical or programming skills such that they can be utilized by both novice and professional users from different domains and with different backgrounds.

The application of simulation is particularly reasonable when empirical studies or observations are too costly, inconvenient, time-consuming, dangerous, or generally impossible. Instead of investigating a real-world system, *cause-effect relationships* of this system are modeled and simulated. This allows for observing the behavior of the model or an individual mechanisms within the model under specific circumstances to confirm or refute assumptions or theories. For this purpose, the values of the model's exogenous variables (*inputs*) are systematically altered to observe the impact they have on the endogenous variables (*outputs*) that are used to measure the model's performance or behavior. By this means, large amounts of synthetic data can be acquired for the investigation of systems and phenomena using data science methods.

This chapter introduces simulation as a technique for the systematic acquisition of synthetic data in data science. Instead of generating a vast data basis by simulating all possible parametrizations of a model, this chapter presents techniques from the field of *data farming*, which enable the problem-related extraction of data in respect of a specific problem. By this means, simulation can help to address data science challenges that are especially associated with the volume of data. The resulting relationship between simulation and data science is bilateral: Simulation experiments enable the efficient acquisition of synthetic data for the use in data science and data science provides approaches for deriving insights from simulation models.

To outline advantages and opportunities simulation offers for data science, this chapter is structured as follows: Section 2 introduces simulation as method for modeling, executing, and investigating artificial systems. In Section 3, the relationship between simulation and data science is outlined to illustrate how simulation models can be used for the acquisition of synthetic data as part of the data science process. Different approaches for the systematic design and execution of experiments are presented in Section 4, with focus on the comparison of different data farming approaches in respect of data science needs. In Section 5, two free-to-use simulation frameworks are introduced, which facilitate the conducting of simulation experiments. Finally, the opportunities simulation offers for data science are summarized.

## 2 What is Computer Simulation?

The history of modern computer simulation starts in the 1940s, when the invention of the ENIAC general-purpose computer enabled scientists to automatically execute mathematical computations for solving numerical problems [39]. Nowadays, more than 70 years later, scientific progress often inherently relies on the use of simulation and research without simulation became unimaginable. Axelrod even introduced simulation as a third way of doing science besides deductive and inductive research, i.e., empirically and theory driven approaches [2]. Based on this classification of scientific advance, some authors postulate the emergence of *data-intensive science* as a fourth paradigm of research, with focus on large data sets from different sources [11]. A special emphasis lies on the strong connection between computational sciences such as simulation and data science. Due to the large amount of data that can be generated by means of simulation, a demand for dedicated techniques arises to explore and extract relevant information.

Simulation is often utilized when the application other approaches is too costly, time-consuming, or cannot deal with the investigated system's complexity [3]. For instance, when analyzing crisis of the banking system, it might be necessary to investigate and understand the fractional-reserve banking mechanisms that allow banks to grant credits as well as its consequences to the banking system itself. In this example, experimentation with the real-world system is impossible as it might expose the financial market to unforeseeable threads. Likewise, the creation of a banking market under laboratory conditions that can be safely used for experimentation is not feasible due to financial and pragmatic reasons. The real-world system is also too complex to be analyzed by means of numerical approaches because of the large number of heterogeneous and independently-acting market participants. Thus, Law proposes simulation as technique of choice [18].

The conducting of a simulation usually consists of two distinct yet mutually dependent tasks: *model building* and *experimentation*. Hence, the corresponding discipline is also referred to as *Modeling & Simulation* (M&S). As this chapter addresses the practical application of simulation for means of data acquisition, the focus lies on the experimentation part of M&S and it is assumed that a suitable simulation model already exists. Comprehensive introductions on the building of simulation models are, for instance, provided by Bonabeau [4], Carson & John [5], and Sokolowski & Banks [36].

With respect to the conducting of simulation experiments, a *black box* perspective on the model is often sufficient (cf. Figure 1). Here, the inner states and mechanisms of the simulation model are not considered and only the *input-output behavior* is investigated [41]. *Inputs* represent exogenous factors that affect the model's behavior such as uncontrollable environmental influences or control variables. *Outputs* of the model are those variables that can be used for observing and assessing the behavior or performance of the model. Simulation is often used to examine the relationship between inputs and outputs, i.e., which particular inputs influence specific outputs or how certain values of inputs minimize or maximize outputs.

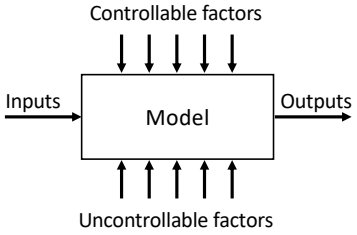



Fig. 1: Black box view on the input-output behavior of simulation models [24].

To analyze the relationship between the model's inputs and outputs, experiments must be systematically executed to generate suitable data and to gradually exploit the model's *response surface* [20]. For this purpose, *data farming* techniques can be applied, which pursue an approach that takes place before *data mining* [13]. While data mining focuses on the discovery of patterns in data sets, data farming starts one step prior to this and targets the generation of relevant data on the model's behavior. Referring to agricultural farmers, relevant data for the analysis is deliberately "grown" and data samples can be drawn from different parts of the model's response surface to selectively assess the quality of data. Data miners, in contrast, can neither influence the quality of the data set nor generate more data. Still, both approaches depend on each other. On the one hand, data farming must provide suitable data sets that can be further processed by means of data mining and other data science techniques. On the other hand, data science provides approaches that allow for deriving information and knowledge from data that was generated by simulation models.

In many scientific publications, mutual benefits are outlined that emerge from the combination of simulation and data science. Feldkamp et al., for instance, combine data farming and visual analytics to investigate the relationship between inputs and outputs of simulation models [8]. Following the *knowledge discovery in databases* process, the authors make use of a data farming approach to acquire data on the model's behavior which is then further analyzed via clustering and visual analytics to identify influential inputs of the model.

Conversely, simulation is also applied as technique in data science, e.g., as part of predictive analytics to validate the used models or to generate sample data of a system's behavior [27]. It is also utilized as independent application, e.g., in data analytics for addressing big data challenges [35]. To this end, Shao et al. demonstrate different applications of simulation in manufacturing and emphasize how data can be generated, which is required for the analysis of domain-specific data analytics applications [35]. Especially the combination of both disciplines allows the user to overcome existing shortcomings. Costs of data processing and acquisition can be reduced when applying data farming to artificial simulation models. Additionally, data points that are missing in the data set can easily be substituted by observations from the model. Finally, for the generation of simulation models, the need to understand all possible cause-effect relationships within the real-world system is decreases as relevant mechanisms can be learned from data.

## 3 Computer Simulation for the Acquisition of Data

After introducing computer simulation as method for the generation of synthetic data and presenting approaches that combine simulation and data science, this section outlines the methodological relationship between simulation and data science. It is illustrated how simulation can be integrated into the process of data acquisition as required in data science. Especially, the use of simulation as a technique for the generation and acquisition of synthetic raw data is addressed.

According to O'Neil and Schutt, the *data science process* starts from the real world [26]. Here, data exists in a variety of forms and contexts such that raw data might be directly acquired or observed from systems within this world. In consecutive steps, the goal of data science approaches is then to process and clean the raw data to prepare them for further analysis. Yet, the acquisition of raw data from the real world is not always feasible or possible. Among other things, this might be due to limited access to the system of interest, the required amount of time and money, or the existence of the system.

Here, benefits become apparent that simulation holds for data science: The use of M&S allows for the creation of an artificial system, which might be a suitable alternative to investigating a system in the real world. Compared to real-world systems, modeled systems can be executed and investigated with slowed or accelerated speed. They are not subject to access restrictions and the initial state of the system can be restored at any time and at no expense. Moreover, artificial systems do not necessarily require the existence of the underlying real-world system, which additionally allows for the generation and investigation of fictive and theoretical systems or effects.

As intended by the design science process, the conventional data acquisition process relies on the collection and export of data from a real-world system, e.g., from the data warehouse of a company or other sources of big data. Gained raw real-world data is then processed, cleaned, explored, and quantitatively analyzed to derive qualitative insights that can be used as basis of a decision-making process.

In contrast to this, the simulation-based data acquisition approach extends and partially replaces this conventional approach of data acquisition (cf. Figure 2). Instead of accessing big data in the real world, only a specific set of small data (*real system data*) is acquired [22]. This includes information that is required for the model building process, e.g., information on the system's mechanisms, involved entities, process flow, and further specifications. Moreover, data that is required for the calibration and parametrization of the model is extracted. Besides real system data, experience and knowledge of domain experts is also required for the model building. After verifying and validating the developed model, simulation experiments are conducted and data farming approaches applied to systematically generate synthetic big data that is required for the application of further design science methods.

Considering trends that come along with the digitalization of our society, e.g., *Internet of Things* (IoT), the potentials of simulation-based data acquisition can be illustrated. This especially includes the use of artificial data for the evaluation of innovative technologies. For instance, Renoux and Klügl outline how agent-based simulation of inhabitants in smart homes can be used to gather realistic artificial

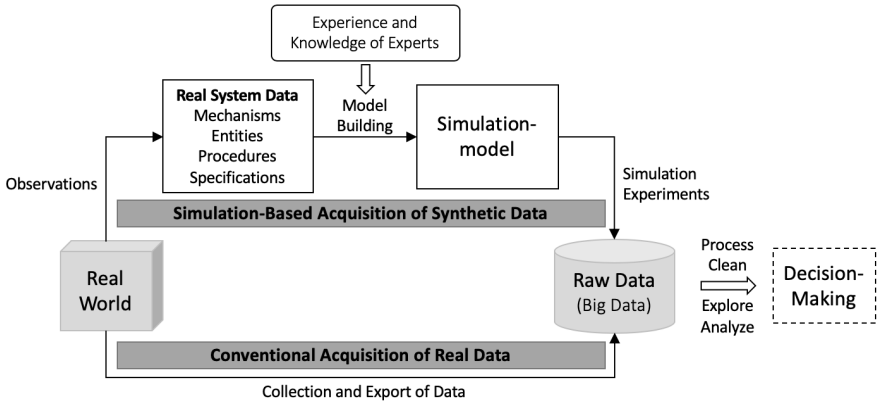



Fig. 2: Simulation-based and Conventional Data Acquisition.

sensor data on human behavior [29]. Such data can then be used to test augmented living algorithms or to identify patterns for learning rules on activities of inhabitants of smart homes. To this end, the authors also refer to *OpenSHS*, a simulator which can be used to extrapolate small data into big data with the aim of testing and evaluating IoT models using smart home data [1].

## 4 Design and Execution of Experiments

To enable and facilitate data farming on a simulation model, standardized *experimental designs* are used to derive experiment plans, which define all experiments that must be executed [32]. In this section, different experimental designs are presented that can be used for the systematic acquisition of data with respect to data science needs. This goes beyond the recommendation of big data only ("the more, the better") but also aims at the heterogeneity of the used or generated data, i.e., how adequately and evenly the data points cover all parts of the investigated response surface. To ensure the systematic investigation of the model's parameter space and the generation of heterogeneous data, simple *factorial designs* are introduced first. Especially for models with a great number of inputs, the suitability of basic factorial designs is often limited due to the combinatorial explosion of parametrizations that are suggested by the experiment plan. To overcome this limitation, this section also introduces more advanced *fractional factorial designs*. In contrast to basic factorial designs, fractional factorial designs investigate the model's parameter space more efficiently by reducing the number of simulated model parametrizations.

In *experimental design* terminology, exogenous inputs of a model are referred to as *factors*, that can be used to control the model during the experimentation. Each factor is defined by a set of admissible qualitative or quantitative values (*levels*), which it can

take. In a manufacturing model, examples of potential factor levels might be simple logical values, e.g., factor *AutomatedAllocation* that can either be *true* or *false*, a set of discrete levels, e.g., factor *QueuingDiscipline* which can take levels *FIFO*, *LIFO*, and *SPT*, or a range of numerical values, e.g., factor *NumberOfMachines* for which all whole numbers between 1 and 15 are admissible [33].

Factorial designs "cross" the levels of the factors to investigate all possible factor-level combinations [24]. In other words, if a model consist of two factors $A$ and $B$ with $a$ respectively $b$ levels, the *cartesian product* $A \times B$ is applied which results in a set of $a * b$ possible parametrizations of the model. Compared to the conventional *one-factor-at-a-time* method, where only one factor is changed and tested in each experiment, factorial designs allow for the investigation of interactions between factors as multiple factors are tested at the same time [15].

Factorial designs are usually defined via the number of factors ($k$) and the number of levels ($m$). Examples for common factorial designs are $2^k$, $3^k$, or the general $m^k$ design. A $2^k$ factorial design is well-suited for the investigation of models with a smaller number of binary factors or factors with a limited number of levels. However, as both the number of factors and levels per factor increase, the number of resulting parametrizations also increases exponentially. This results in a combinatorial explosion of data points that are suggested by the experiment plan. For instance, the $2^k$ factorial design of a model with 10 factors and only 5 levels per factor consists of almost 10 million individual parametrizations (cf. Figure 3). It also must be considered that many simulation models consist of stochastic components that for example represent real-world variations of processing times. The simulation of each parametrization must then be replicated multiple times to estimate the underlying probability distribution. Thus, Sanchez and Wan [33] suggest not to apply $m^k$ designs in case the number of factors or levels exceeds 10.

| No. of factors | $10^k$ factorial | $5^k$ factorial | $2^k$ factorial |
|---|---|---|---|
| 1 | 10 | 5 | 2 |
| 2 | $10^2 = 100$ | $5^2 = 25$ | $2^2 = 4$ |
| 3 | $10^3 = 1,000$ | $5^3 = 125$ | $2^3 = 8$ |
| 5 | 100,000 | 3,125 | 32 |
| 10 | 10 billion | 9,765,625 | 1,024 |
| 20 | *don't even* | 95 trillion | 1,048,576 |
| 40 | *think of it!* | 9100 trillion trillion | 1 trillion |

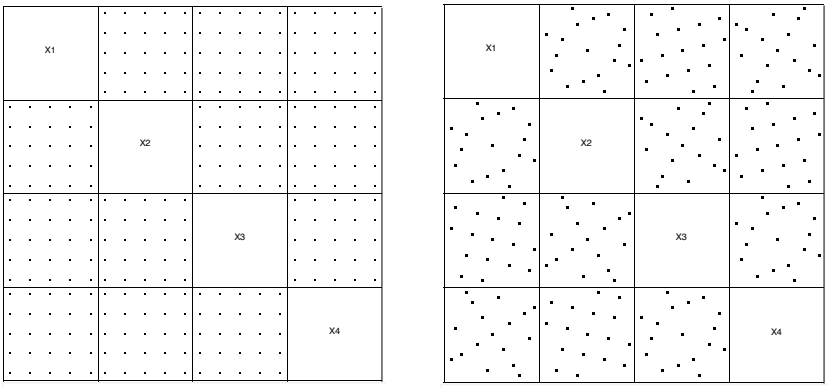
Fig. 3: Data Requirements for Factorial Designs [33].

Data generated by applying $m^k$ designs allows for the identification of interactions between two and more factors. By confounding these interactions, the efficiency of $m^k$ designs can be increased as only a *fraction* of the intended parametrizations needs to be executed. Resulting $m^{k-p}$ fractional factorial designs generate an experiment

plan which consists of a subset of parametrizations from the respective $m^k$ design. Hence, the larger $p$ is chosen, the less data but also information is generated [18].

Other examples of fractional designs that are well-suited for greater numbers of factors and levels are *Nearly Orthogonal Latin Hypercubes* (NOLH) and approaches that combine different designs, e.g., *FFCSB-X* [34]. According to Sanchez, NOLH have good space-filling properties for $k \leq 29$, meaning that all parts of the simulation model's parameter space have the same probability of being investigated and require a considerably lower amount of data points. She also illustrates, that while a $5^{10}$ design consists of almost 10 million data points, 33 parametrizations are sufficient for a NOLH design of 10 factors. Furthermore, reducing the number of required experiments allows for the execution of a sufficient number of replication per parametrization to investigate the distribution of the results as well as for the execution and combination of multiple NOLH designs.

Figure 4, illustrates the space-filling properties of NOLH by comparing the resulting coverage of the parameter space to a $m^k$ factorial design. For each possible combination of two inputs $x_1$ to $x_4$, all investigated factor-level combinations are visualized. In the scatterplot matrix of the $5^4$ design, the grid-like shape of the investigated tuples can be observed. Accordingly, the parts of the parameter space that fall between the grid cells are never analyzed. Thus, the scatterplot matrix of a specific $m^k$ design will always be the same for a specific model. In contrast to this, the matrix of a NOLH design consist of a random permutation of all possible tuples in accordance with certain restrictions that ensure the coverage of the parameter space. Hence, the tuples that are suggested by the NOLH design are distributed randomly such that any possible factor-level combination might be suggested by the design.




(a) $5^4$ Factorial Design (b) NOLH Design

Fig. 4: Scatterplot Matrices for a) $5^4$ Factorial Design and b) NOLH Design with 4 factors in 17 runs [33].

An example of a sophisticated design that combines different more basic designs is FFCSB-X. Here, *CSB-X* is applied after using fractional factorial design to estimate the direction of the factors' effects. It pursues a *divide-and-conquer* approach to determine those factors that have the greatest effect on the model's behavior. According to Sanchez and Wan, FFCSB-X is more efficient than CSB-X and can be applied for models with more than 1,000 factors and with a large number of discrete or even continuous factor levels [33]. Yet, the application of this approach is challenging as it requires advanced simulation knowledge and is not pre-implemented in standard simulation frameworks. Other more basic designs are often available in ready-to-use packages, e.g., via the R Archive Network (CRAN) or MATLAB.

Regardless of the used design, it must be ensured that a sufficient number of replications is executed for stochastic models to precisely measure the performance of the model [30]. For each execution of the model, the made observation of the model's output can be considered as a sample drawn from an unknown probability population. Thus, a sufficient number of experiments must be executed such that the sample mean ($\tilde{x}$) can be used to adequately estimate the population mean ($\mu$). For this purpose, Hoad et al. suggest the use of confidence intervals [12].

Summarizing, factorial designs are well-suited to gather data on smaller models and to gain insights on interactions between the model's factors. When the application of factorial designs is limited, fractional factorial designs provide more efficient experiment plans that can handle a greater number of factors and levels. Yet, the reduced amount of data might also results in a reduced amount of information that can be gained from the simulation data. A detailed overview and comparison of different experimental designs is provided by Sanchez and Wan [33], Kleijnen et al. [16], and Montgomery [24]. However, with respect to the combination of simulation-based data acquisition and data science approaches, the execution of a great number of experiments as well as the quantity and controllability of generated data set might no longer be a showstopper. This is, because data science provides more sophisticated and dedicated analytical approaches.

## 5 Simulation Frameworks and Toolkits

To apply experimental plans that were generated using factorial designs, the simulation model under investigation must be executed. Usually, simulation models are developed using commercial or free-to-use simulation frameworks rather than proprietary software developments. This facilitates the model building process, as commonly used modeling constructs or domain-specific formalisms are provided by these frameworks and can be applied out of the box. Moreover, a runtime environment is provided that enables the user to easily execute the model with a specific parametrization or to automatically observe the model's behavior under different parametrizations. To this end, scaling and parallelization of experiments as well as logging and first visualizations of output data are further functionalities that are often provided by such frameworks.

This section introduces *NetLogo* and *Repast Simphony* as related modeling environments that are applicable for novice as well as for professional simulation users. Both frameworks are especially well-suited for building and executing *agent-based models* in which actions of and interactions between individual entities (*software agents*) are investigated, e.g., in economic markets [10] or social networks [31]. In contrast to other modeling paradigms, e.g., *system dynamics* or *microsimulation*, the autonomous and proactive decision behavior of each individual is in focus, which allows for the investigation of a system's behavior on a microlevel [7]. With respect to the simulation of agent-based models, the focus of this section also lies on the execution assistance *BehaviorSpace* that is provided by NetLogo. It facilitates the systematic execution of simulation experiments to examine how different factor levels influence the agents' behavior [38].

Of both frameworks, NetLogo is the one that is more suitable for novice users. It is lightweight, makes use of the functional and procedural *Logo* programming language, provides a user interface for the development and execution of the model, and facilitates the export and visualization of observed data (cf. Figure 5). Moreover, NetLogo's model library consist of a great number of sample models that can be downloaded and modified as required.

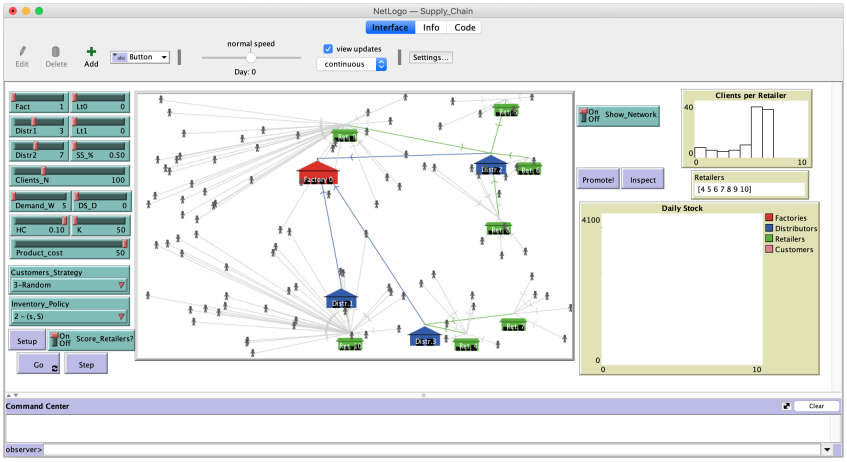



Fig. 5: Interface of the NetLogo simulation framework.

Beside its ease of use for model building, NetLogo also provides assistance functionalities that facilitate the design and conducting of experiments. In *BehaviorSpace*, individual experiments can be configured as combination of different factor levels that shall be investigated. Referring to the $m^k$ design, one or many distinct levels or a range of levels can be selected for each factor such that BehaviorSpace deduces and executes all possible factor-level combinations. Moreover, the number of replications can be determined that must be executed for each distinct parametrization of the model. After designing an experiment, all runs can be automatically executed

and the respective results are logged into a CSV file. NetLogo enables the use of multiple CPU cores to distribute and parallelize the execution of the runs.

Repast Simphony is more comprehensive compared to NetLogo and provides a greater range of functionalities, which allows for the building and executing of more sophisticated simulation models. To import NetLogo models into Repast, the *ReLogo* language can be used, which is a NetLogo-inspired domain specific language for the development of agent-based models in Repast [28]. Besides Repast Simphony, that is mostly Java-based, a C++-based version (*Repast HPC*) exists, which is intended for the use on clusters and supercomputers. In this regard, a comprehensive and practical introduction to agent-based modeling is provided by Wilensky and Rand [40] who maintain and teach NetLogo.

NetLogo and Repast Simphony are only two examples of a large number of simulation frameworks. In their literature review, Kravari and Bassiliades provide an overview on different platforms that can be used for agent-based modeling [17]. Even though not all agent platforms are also simulation frameworks, most of them include functionalities that facilitate the execution and simulation of multiagent systems. Other prominent examples of agent simulation frameworks are *AnyLogic*, *MASON*, and *Swarm*.

In addition to agent-based simulation, there are also other established simulation paradigms. To efficiently model progress of time and to skip periods of time in which no relevant actions take place during simulation, the *discrete event* paradigm only calculates the next model state when specific predefined events take place. This is in contrast to continuous simulations, where time continuously progresses even if no actions take place. Franceschini et al. provide an overview of different frameworks that are suited for the simulation of discrete event systems [9].

It is noticeable that many of the introduced frameworks make use of the Java programming language. Yet, there are also extensions of existing systems or additional packages that enable simulation by means of other languages, e.g., Python, which might be more familiar to data scientists. Examples include DES [23], ManPy [6], or Repast Py [25]. Especially with regard to data science, Python is a programming language that is frequently used for extracting, cleaning, and analyzing data sets, e.g., *Pandas* for exploratory data analysis or *scikit-learn* for machine learning. This facilitates first steps in the application of simulation for data scientists, as they can make use of a programming language they most likely are familiar with.

## 6 Investigation of a Credit Market

To emphasize how simulation can be applied with respect to data science requirements, this section introduces a simple NetLogo simulation model. Besides the formulation of potential analysis goals (*hypotheses* [21]), this section elaborates on the implementation of the model as well as the possibilities the model provides for data farming. The outlined model consists of a banking market and is taken from Hamill and Gilbert's introduction to "Agent-Based Modelling in Economics" [10].

Especially in economics, the use of simulation is promising to investigate cause-effect relationships between different entities in the system, to analyze the mechanisms behind certain phenomena, or to examine the effect new rules or norms have on the system's behavior and resilience. In their book, Hamill and Gilbert introduce the banking market as an omnipresent system with sophisticated dynamics and partially unnoticed mechanisms. The authors especially stress on the credit system of *fractional reserve banking*, where banks can multiply money by granting credits, which are then used to pay money to third parties, which again potentially deposit the money in the bank. The deposited money can then be used to grant further credits that are smaller than the original credit. This mechanisms results in an recursive credit system where the bank gains money from earlier credits, which are deposited by third party retailers.

According to the authors, when analyzing such processes most approaches leave out partial or monthly repayments of granted loans. This is undesirable as from the bank's perspective as the repaid money can be used to grant further credits which has a large effect on the bank's potential loan volume. When thoroughly implementing such mechanisms, respective models can be used to investigate the stability of the banking market. Potential triggers of crises can be analyzed, i.e., solvency crisis and liquidity crisis, and strategies for preventing or handling crises can be evaluated, e.g., regulatory frameworks and standards such as the global and voluntarily regulatory frameworks Basel I - III. To this end, understanding the relationship between credit institutions, regulators, and households seems most relevant such that a potential questions that can be answered by means of simulation might be: *How does the borrower's budget affect the stability of fractional reserve banking?*

The presented model consists of one bank with an initial deposit of 1 million GBP and 10,000 households with an average monthly budget of 1,000 GBP. There are two kinds of loans, i.e., 25-year mortgages and 3-year consumer loans. According to the limitations of the regulators, the bank decides how much money to provide as credits to the households. The households then use the borrowed money to buy from other households who decide to deposit the money at the bank. This money is then again available to the bank and can be granted as further credits. Moreover, the borrowers of the loans repay on a monthly basis and the bank also uses this money to grant further credits.

The described model can be used to conduct simulation experiments with different parametrizations of the banking system. Potential configurations that might be analyzed include different reserve ratios of the bank, the ratio of households that are borrowers and savers, or the amount of money the households spend for repayments. To investigate the resilience of the banking system under different circumstances, it can be simulated how the bank reacts to the absence of repayments in terms of profit and vulnerability.

To analyze different configurations of the model, the use of data farming approaches and experimental designs is reasonable. This allows for the systematic investigation of the model's parameter space and the identification of interactions between the model's factors as well as the overall impact of each factor. However, from a simulation perspective, the conducting of experiments is not sufficient for

the identification of circumstances that lead to resilient or fragile banking markets. Likewise, the analysis of real bank data is not satisfying as information from multiple bank crashes is required to derive patterns. Here, the potentials that emerge from combining simulation and data science can be highlighted. Based on a smaller amount of real system data, simulation allows for the generation of a large amount of artificial banking data for different policies of the bank, external regulations, and kinds of borrower. This set of big data can then be processed and explored to derive potential insights regarding the crisis resistance of the banking system. Without the use of simulation, data science methods would rely on real-world big data, which might not be accessible or exist at all, and result in limited possibilities to identify and analyze causal relationships in banking markets.

## 7 Conclusions

Limited access to real-world data imposes challenges on the acquisition of data and thus also on the application of data science techniques. To overcome a lack of real data, this chapter introduced the simulation-based acquisition of synthetic data. By modeling and executing an artificial system, limitations of big data acquisition are overcome and even fictive systems can become subject to data science approaches. To enable the efficient acquisition of synthetic data from simulation models, this chapter suggested data farming as a technique for the systematic extraction of data from the model's parameter space. Through this, the availability of real-world big data is no longer mandatory for the application of data science techniques and the observation of smaller and specific real system data is sufficient. Finally, this chapter outlined the methodological relationship between simulation and data science and illustrated how data science can benefit from the utilization of simulation.

## References

1. Nasser Alshammari, Talal Alshammari, Mohamed Sedky, Justin Champion, and Carolin Bauer. Openshs: Open smart home simulator. *Sensors*, 17(5):1003, 2017.
2. Robert Axelrod. Advancing the art of simulation in the social sciences. In *Simulating social phenomena*, pages 21–40. Springer, 1997.
3. Jerry Banks and Randall Gibson. Don't Simulate When... 10 Rules for Determining when Simulation Is Not Appropriate. *IIE solutions*, 29(9):30–33, 1997.
4. Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287, 2002.
5. II Carson and S John. Introduction to modeling and simulation. In *Proceedings of the 37th Winter Simulation Conference*, pages 16–23. Winter Simulation Conference, 2005.
6. Georgios Dagkakis, Ioannis Papagiannopoulos, and Cathal Heavey. Manpy: An open-source software tool for building discrete event simulation models of manufacturing systems. *Softw. Pract. Exper.*, 46(7):955–981, July 2016.
7. Paul Davidsson. Multi agent based simulation: beyond social simulation. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 97–107. Springer, 2000.

8. Niclas Feldkamp, Sören Bergmann, and Steffen Strassburger. Visual analytics of manufacturing simulation data. In *Proceedings of the 2015 Winter Simulation Conference*, pages 779–790. IEEE Press, 2015.
9. Romain Franceschini, Paul-Antoine Bisgambiglia, Luc Touraille, Paul Bisgambiglia, and David Hill. A survey of modelling and simulation software frameworks using discrete event system specification. In *OASIcs-OpenAccess Series in Informatics*, volume 43. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.
10. Lynne Hamill and Nigel Gilbert. *Agent-based modelling in economics*. John Wiley & Sons, 2015.
11. Tony Hey, Stewart Tansley, Kristin M Tolle, et al. *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA, 2009.
12. Kathryn Hoad, Stewart Robinson, and Ruth Davies. Automated selection of the number of replications for a discrete-event simulation. *Journal of the Operational Research Society*, 61(11):1632–1644, 2010.
13. Gary E Horne and Ted E Meyer. Data farming: Discovering surprise. In *Proceedings of the 36th Winter Simulation Conference*, pages 807–813. Winter Simulation Conference, 2004.
14. Marco A Janssen, Lilian Na'ia Alessa, Michael Barton, Sean Bergin, and Allen Lee. Towards a community framework for agent-based modelling. *Journal of Artificial Societies and Social Simulation*, 11(2):6, 2008.
15. Jack PC Kleijnen. Design and analysis of simulation experiments. In *International Workshop on Simulation*, pages 3–22. Springer, 2015.
16. Jack PC Kleijnen, Susan M Sanchez, Thomas W Lucas, and Thomas M Cioppa. State-of-the-art review: a user's guide to the brave new world of designing simulation experiments. *INFORMS Journal on Computing*, 17(3):263–289, 2005.
17. Kalliopi Kravari and Nick Bassiliades. A survey of agent platforms. *Journal of Artificial Societies and Social Simulation*, 18(1):11, 2015.
18. Averill M. Law. *Simulation modeling and analysis*. McGraw-Hill series in industrial engineering and management science. McGraw-Hill Education, Dubuque, 5 edition, 2013.
19. Catherine M Lloyd, James R Lawson, Peter J Hunter, and Poul F Nielsen. The cellml model repository. *Bioinformatics*, 24(18):2122–2123, 2008.
20. Fabian Lorig. *Hypothesis-Driven Simulation Studies – Assistance for the Systematic Design and Conducting of Computer Simulation Experiments*. Springer, 2019.
21. Fabian Lorig, Daniel S Lebherz, Jan Ole Berndt, and Ingo J Timm. Hypothesis-driven experiment design in computer simulation studies. In *Simulation Conference (WSC), 2017 Winter*, pages 1360–1371. IEEE, 2017.
22. Anu Maria. Introduction to modeling and simulation. In *Proceedings of the 29th Winter Simulation Conference*, pages 7–13. IEEE Computer Society, 1997.
23. Norm Matloff. Introduction to discrete-event simulation and the simpy language. *Davis, CA. Dept of Computer Science. University of California at Davis. Retrieved on August*, 2(2009), 2008.
24. Douglas C Montgomery. *Design and analysis of experiments*. John wiley & sons, 2017.
25. Michael J North, Nicholson T Collier, and Jerry R Vos. Experiences creating three implementations of the repast agent modeling toolkit. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 16(1):1–25, 2006.
26. Cathy O'Neil and Rachel Schutt. *Doing data science: Straight talk from the frontline*. O'Reilly Media, Inc., 2013.
27. Huiyin Ouyang and Barry L Nelson. Simulation-based predictive analytics for dynamic queueing systems. In *Simulation Conference (WSC), 2017 Winter*, pages 1716–1727. IEEE, 2017.
28. Jonathan Ozik, Nicholson T Collier, John T Murphy, and Michael J North. The relogo agent-based modeling language. In *Simulation Conference (WSC), 2013 Winter*, pages 1560–1568. IEEE, 2013.
29. Jennifer Renoux and Franziska Klügl. Simulating daily activities in a smart home for data generation. In *Proceedings of the 2017 Winter Simulation Conference*. IEEE, 2017.

30. Stewart Robinson. *Simulation: the practice of model development and use*. Wiley Chichester, 2004.
31. Stephanie C. Rodermund, Fabian Lorig, Jan Ole Berndt, and Ingo J. Timm. An agent architecture for simulating communication dynamics in social media. In Jan Ole Berndt, Paolo Petta, and Rainer Unland, editors, *Multiagent System Technologies*, pages 19–37, Cham, 2017. Springer International Publishing.
32. Susan M Sanchez. Simulation experiments: better data, not just big data. In *Proceedings of the 2014 Winter Simulation Conference*, pages 805–816. IEEE Press, 2014.
33. Susan M Sanchez and Hong Wan. Work smarter, not harder: a tutorial on designing and conducting simulation experiments. In *Proceedings of the Winter Simulation Conference*, page 170. Proceedings of the 2012 Winter Simulation Conference, 2012.
34. Susan M Sanchez, Hong Wan, and Thomas W Lucas. Two-phase screening procedure for simulation experiments. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 19(2):7, 2009.
35. Guodong Shao, Seung-Jun Shin, and Sanjay Jain. Data analytics using simulation for smart manufacturing. In *Proceedings of the 2014 Winter Simulation Conference*, pages 2192–2203. IEEE Press, 2014.
36. John A Sokolowski and Catherine M Banks. *Principles of modeling and simulation: A multidisciplinary approach*. John Wiley & Sons, 2011.
37. Ingo J Timm and Fabian Lorig. A survey on methodological aspects of computer simulation as research technique. In *Proceedings of the 2015 Winter Simulation Conference*, pages 2704–2715. IEEE Press, 2015.
38. Seth Tisue and Uri Wilensky. Netlogo: A simple environment for modeling complexity. In *International conference on complex systems*, volume 21, pages 16–21. Boston, MA, 2004.
39. Stanislaw M Ulam. *Analogies between analogies: the mathematical reports of SM Ulam and his Los Alamos collaborators*, volume 10. Univ of California Press, 1990.
40. Uri Wilensky and William Rand. *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*. MIT Press, 2015.
41. Bernard P Zeigler, Tag Gon Kim, and Herbert Praehofer. *Theory of modeling and simulation*. Academic press, 2000.