Noname manuscript No.
(will be inserted by the editor)

Computing and Experiments

A methodological view on the debate on the scientific nature of computing

Viola Schiaffonati · Mario Verdicchio

the date of receipt and acceptance should be inserted later

Abstract The question about the scientific nature of computing has been widely debated with no universal consensus reached about its disciplinary status. Positions vary from acknowledging computing as the science of computers to defining it as a synthetic engineering discipline. In this paper we aim at discussing the nature of computing from a methodological perspective. We consider, in particular, the nature and role of experiments in this field, whether they can be considered close to the traditional experimental scientific method or, instead, they possess peculiar and unique features. We argue that this experimental perspective should be taken into account when discussing the status of computing. We critically survey how the experimental method has been conceived and applied in computing, and some open issues that could be tackled with the aid of the history of science, the philosophy of science, and the philosophy of technology.

Keywords Computing · Computer Science and Engineering · Experiments · Experimental method

V. Schiaffonati
Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, piazza Leonardo da Vinci 32, 20133, Milano, Italy
Tel.: +39-02-23993622
Fax: +39-02-23993411
E-mail: viola.schiaffonati@polimi.it

M. Verdicchio
Università degli Studi di Bergamo, Dipartimento di Ingegneria, Viale Marconi 5, 24044, Dalmine (BG), Italy
Tel.: +39-035-2052358
Fax: +39-035-2052310
E-mail: mario.verdicchio@unibg.it

1 Introduction

The debate around the nature of computing has always been a major concern since the beginning of the discipline. This debate presents a variety of positions.

Computing[1] is defined as the science of *computers* and related phenomena [44], but also as the natural science of *procedures* [47], or the *artificial science* of phenomena related to computers [45] [48]. Moreover, computing is considered as the *study* (and not the science) of *algorithms* and related phenomena [36], of *computational processes* [9], of information [31]. Alternatively, it is characterized as a synthetic *engineering* discipline [6] or as a new species of engineering [38].

Computing is considered as dealing with computers, algorithms, processes, procedures, information, depending on the choice made by who is providing the definition. Moreover, it can be considered a science, or a study, a corpus of knowledge, a discipline. Other labels that may be used to refer to it are: computing science, informatics, computer engineering, information science.

The attention to names and definitions clearly reflects a struggle for status that has been characterizing computing since its birth. Whether computing is considered science (even of a special kind) or engineering, it is a matter of fact that this characterization moves around its relationship with science. In time, the interdisciplinary nature of computing has been widely recognized and, accordingly, it is now defined partly as scientific, partly as mathematical, and partly as technological [56].

But "what's in a name?" According to [6], the 'science' in 'computer science' is a misnaming, implying that researchers in computing (also known as computer scientists) have to accept a hierarchy in which natural scientists are more respected than engineers. On the contrary, in [30] it is claimed the scientific status of computing, given its uniqueness. And even if computing is not a natural science in the traditional sense as physics or biology, it is natural in the sense that it studies naturally (and also artificially) occurring information processes [17].

Although the debate around its nature is nowadays less radical, and the discipline is considered as a special kind of science and of engineering or, in other words, as a discipline comprising both a scientific and an engineering part, still its status is discussed in terms of similarities and differences with, in the best cases, a naive definition of science or, in the worst cases, an incorrect one.

If we consider the so-called Denning report [13], the ACM 'official' view on the discipline, the investigation of a phenomenon by adopting an experimen-

[1] While recognizing the relevant difference between the theoretical and practical ends of the computing spectrum, introducing a taxonomy is beyond the scope of this work. We use here the term 'computing' to refer to the relevant field in general in the same sense of [14], although some readers may be more familiar with the terms 'computer science' or 'computer science and engineering'. We find 'computing' a better fit for the purposes of this work because it does not come with any reference to or hint at the stances that emphasize the scientific or engineering nature of this field.

tal scientific method is reduced to an oversimplified list of processes (form a hypothesis, construct a model and make a prediction, design an experiment and collect data, analyze the results). Even when Denning, one of the most influential opinion makers in this debate, argues in favor of the scientific status of computing, his vision of science boils down to "the quest to understand what is so about the world. Through observation and experimentation, scientists seek to discover recurrent phenomena. They formulate models to capture recurrences and enable predictions, and they seek to validate or refute models through experiments" [14]. The idea here is that, since much computing conforms to these ideals, it is indeed a science. Advocating the scientific character of the discipline goes hand in hand with the shifting of the focus from computers to computation and with the consideration of the latter as a domain distinguished from and with equal status as physics, society, and biology. As these domains are the subject of physical, social, and life sciences, respectively, so is computation the topic of computing [14].

Within this debate on the disciplinary nature of computing our work aims at adding some steps by investigating such question with an analysis of the nature of its experiments, thus adding a methodological point of view to the ongoing discourse.

The attempt to characterize computing in terms of its method calls for the analysis of its relationship with mathematics and the role the latter discipline has in the former. Although mathematics constitutes one of the foundations of computing, we argue that computing has a strong empirical character and is not just a branch of mathematics and, thus, it should not adopt only deductive methods.

One of the arguments, which we fully embrace against the idea that "demonstrations can take the place of experiments" [30], has been proposed in [50]. The idea that "instead of debugging a program, one should prove that it meets its specifications, and this proof should be checked by a computer program" [41] is an old dream of the field, reflecting its strong connections with mathematical research and methods. However, formal verification cannot prove that the behavior of a program corresponds to how the program was intended to behave. The existing gaps between models, algorithms, computer systems and the world are a strong argument for us to accept the empirical character of computing.

Some of the seminal works about the methodology of computing [8] revolve around abstraction. In [9] for example, although recognizing the indispensability of empirical methods, the author believes that, from a methodological point of view, the most probing question concerns the relation between mathematics and computing. So questions such as "Is there a sense in which computer science is experimental science?" remain unanswered even if programmatically stated.

We believe, however, that a methodological analysis in terms of experiments could be beneficial. Even if the definition of the scientific experimental method is extremely complex and far from being not controversial, there exists a shared tradition on what an experiment is and how experimental method

works that goes beyond all the controversies in the philosophy of science, which lie out of the scope of this work.

The same does not go for what science is in general and how it works: in this case the disagreement is far more evident and the issues about demarcation are still very controversial (see [25] for an introductive overview to the main positions within this debate). If we try to define a discipline as scientific only according to the objectives science is supposed to have (exploring, describing, predicting, and explaining phenomena), then some activities (such as astrology) would have to be considered scientific, because they adopt these aims. In particular, the capability to make predictions, which in many cases is considered a necessary and sufficient condition for science, does not ensure the 'scientificity' of a discipline, according to the peculiar method descended from the Scientific Revolution. Let us think, for example, of the Aristotelian-Ptolemaic system and its ability to make better predictions over the Copernican system for many years [53].

As the challenge of defining the concept of science is affected by such controveries, any attempt to investigate whether computing is a science or not will necessarily have to deal with these problems. We do not want to run away from a daunting enterprise, but we think that there is another path that enables us to start a discussion without immediately stepping into a conceptual quagmire. Instead of trying to define the disciplinary nature of computing in an absolute way, we rather intend to compare it with more traditional disciplines, and assume a methodological perspective by focusing such comparison on how the experiments in computing are characterized, in search for analogies and differences that are meant to help us shed light on the nature of this discipline.

This work is structured as follows: Section 2 is comprised of an overview on the the scientific experimental method and its general principles; Section 3 provides a critical review of what has been said about experiments in computing, and indicates some open issues that are worth tackling; finally, Section 4 concludes by shedding some light on future steps to take in this research path.

2 Science and Experiments

We intend to fill the gap left by the debate on the disciplinary identity of computing, which has traditionally neglected its relationship with the experimental method. Let us start by discussing how the experimental method has been shaped when modern science was born.

2.1 The Scientific Experimental Method

One of the traits that distinguishes science from other forms of investigation of the world is the adoption of a peculiar method based on experiments and originated during the Scientific Revolution of the XVII century. The scientific method, of course, is not a single, well-defined list of clear cut processes, but

it is a collection of approaches, techniques, and procedures to investigate the world in which, at least for natural sciences, experiments play a major role. It is common to refer to the scientific method as the *experimental method* to stress the centrality of experiments intended, since the Scientific Revolution, as sort of questions posed to nature. These questions are related to specific situations and deal with some parameters that are measured in the course of the experiments themselves.

Our modern conception of experiment has emerged during the Scientific Revolution. Since then, science has become experimental, where 'experimental' means based on experiments, which are more than simple collections of observations. An experiment can be seen as a *controlled experience*, namely as a set of observations and actions, performed in a controlled context, to support a given hypothesis. In general, while experiments are performed in controlled conditions, this does not hold for observations. For example, observing a drop of water through a microscope does not constitute an experiment. On the contrary, observing the same drop, after having colored it with a chemical reagent in order to evidence some microorganisms, can be considered an experimental procedure performed to test the behavior of the drop under some controlled circumstances.

Two issues seem central to grasp the very idea of experiment: the possibility of *controlling* some of the features of a phenomenon under investigation and the *purpose* for which an experiment is performed. Control deals with the idea that experiments consist in producing controlled circumstances, in which it can be assumed that a studied relationship does not depend on changes in non-controlled factors, either because they are constant or because the relationship under investigation has been made independent of the variations of these uncontrolled factors. In other words, in an experiment the phenomenon under investigation must be treated as an isolated object. This is clearly an artificial situation: it is assumed that other factors, which are not under investigation, do not influence the studied relationship. This is the reason for experiments being performed in the artificial conditions of laboratories, since in the real world it is not usually possible to control every factor that can influence the studied phenomenon [22].

There exists a long tradition (starting from Francis Bacon) that considers an experiment as a sort of question, posed in a specific situation and dealing with some parameters measured in the course of the experiment itself. Let us consider for example the experiment that allowed Galileo Galilei to discover the famous law of falling bodies, claiming that the distance traveled by a falling body is directly proportional to the square of the time it takes to fall. In this case the controlled parameters chosen by Galileo were the space covered by a falling body and the time employed to cover this space, in an ideal situation of absence of air (vacuum). The question associated to this experimental situation is the following: does a mathematical constant relation between the values of these quantities hold? The answer given by Galileo is not only that this constancy exists, but also that the covered space is proportional to the square of the time employed to cover it. Therefore, this experiment achieves a

universal result, which depends on the parameters chosen and measured in the experiment itself. The choice of the experimental factors and the possibility to control them, by isolating them from all the other factors that are considered not relevant, are central for any successful experiment.

Despite these general characterizations, a precise definition of experiment is not straightforward. Rather than a definition, a non-exhaustive list of properties can be useful to better shape this concept. Experiments make use of *precise measurements* in order to quantitatively describe the phenomenon under investigation; they must be *repeatable* at different times and in different places to check the validity and universality of their results; they must be *reproducible* by other scientists to confirm that their results are independent of the precise details of the experiments themselves; they must be *comparable* in order to compare results of different experiments; they must be described with a *precise language* to give rigor and precision to experimental data; they should use *measurement instruments*, when possible, to enhance human capabilities.

These very general features of experiments are in accordance with the view of science originated at the beginning of Scientific Revolution: science is physics, considered as a unique corpus without any further differentiation within it. Accordingly, the philosophical and epistemological issues that arise in experimental practice reflect this view and are devoted to general problems, such as to investigate the general reasons trusting experimental results. Today, both the characterization of experiments and the analysis of the correlated philosophical problems reflect the high specialization of science. Since the XVII century, science has progressively become more specialized and, today, it is almost impossible to reflect on science in its generality. Current science is composed of a large number of specific disciplines that range from physics to economics and psychology, that were not even considered as scientific few decades ago. Within each one of such disciplines the general features of experiments need to be concretely translated and the corresponding philosophy of experimentation has to deal with specific experimental problems that can largely differ across disciplines [46].

2.2 Experimental Principles

Looking at the traditional experimental method, it is possible to single out some principles that constitute its core. They are *comparison*, *reproducibility* and *repeatability*, *justification* and *explanation*, which, although not exhausting the complexity of experimental method, represent nevertheless some defining characteristics of experiments. They constitute the very core of the modern conception of experiments, in which they are so deeply rooted that they are not even referred to in an explicit way in well-developed experimental disciplines. Developed in the context of the Modern Scientific Revolution, these principles helped develop a view of science as a collective activity. Science has become the activity of strongly connected scholars, widespread all over Europe, and highly interested in exchanging their results and achievements. The ancient

conception of a single scientist (or, better, of a philosopher of nature, as scientists were called until the XIX century), capable of carrying out his/her whole research in isolation, slowly faded off, due both to the exceptional growth of scientific knowledge, almost impossible to be managed by a single person, and to the acknowledgment of science as a critical activity [57]. This latter point is central in the development of the conception of experiment considered as more than just a collection of observations. Better achievements can be gained if knowledge is at disposal and can be exchanged among scholars. Clearly, the possibility to compare different results and to make reproducible insightful experiments is at the basis of this new attitude.

Let us consider the principles in more detail.

Comparison

The meaning of 'comparison' is two-fold. On the one hand, it means to know what has been already done in the past within the same field of research, both for avoiding to repeat uninteresting experiments and for getting suggestions on what the interesting questions could be. On the other hand, comparison refers to the possibility for future researchers to accurately compare their new results with the old ones. If these features are easily given for granted in principle, serious difficulties can arise in actual comparisons. In particular, a direct comparison may be problematic. To be comparable with others, an experiment must be thoroughly documented. Moreover, comparison should be accomplished on the basis of a 'sincerity' principle that often can be easily disregarded. One should report any strange or unexpected phenomenon encountered during experimentation. This is not just for reasons of intellectual honesty, but also because anomalies can reveal something important and bring to new discoveries.

The discovery of the planet Neptune in 1846 is a well-known example of how an anomaly can turn out to be decisive for a new discovery. According to the Newtonian theory, astronomers were able to calculate the hypothetical orbit of Uranus, at that time considered the most distant planet of the Solar system. However, this hypothetical orbit was not in accordance with the orbit actually reported in well-documented series of observations. This incongruence was not considered sufficient to reject the whole Newtonian theory, but gave some hints on the fact that something in the theory itself needed a revision. Two astronomers, Adams and Le Verrier, tried to explain this anomaly by postulating the existence of a further planet not yet observed. They first assumed the existence of a new planet (Neptune), farther than Uranus; they calculated its mass and position; and, eventually, they observed it in a position almost identical to the calculated one [26].

Reproducibility and Repeatability

These features are often confused but, although tightly connected, they refer to different desiderata. They are also related to comparison and to the very

general idea that scientific results should undergo to the most severe criticisms in order to be strongly confirmed.

Reproducibility is the possibility to verify, in an independent way, the results of a given experiment. It refers to the fact that other experimenters, different from the one claiming for the validity of some results, are able to achieve the same results, by starting from the same initial conditions, using the same type of instruments, and adopting the same experimental techniques. As in the case of comparison, to be reproducible an experiment must be fully documented.

Repeatability concerns the fact that a single result is not sufficient to ensure the success of an experiment. A successful experiment must be the outcome of a number of trials, performed at different times and in different places. These requirements guarantee that the result has not been achieved by chance, but is systematic. Repeatability should be intended here in a wide meaning as, according to [29], real repeatability is never realized in practice. Typically, repetitions of an experiment are attempts to do the same thing better, namely to produce a more stable, less noisy version of the phenomenon created during the experiment itself. Even when the goal is to try to make precise measurements, what is called for is a better experiment, that increases the precision of measurements so that systematic errors can be eliminated. The only cases of literal repetitions of experiments are those in which people do not believe experimental results, and repetition is made to overcome this skepticism.

A particularly clear example to illustrate both reproducibility and repeatability is given by the controversy about the supposed discovery of the cold fusion claimed in 1989 by Stanley Pons and Martin Fleischmann, two chemists at the University of Utah and at the Southampton University, respectively [10]. In a news press, they claimed to have produced nuclear fusion in a tabletop experiment, reporting anomalous heat production of a magnitude that (they asserted) would defy any explanation, except in terms of nuclear processes. They interpreted the absence of neutrons in their experiment as the proof for a new type of nuclear reaction. The paper describing this supposedly revolutionary result (rejected by *Nature*, but accepted for publication in the *Journal of Electroanalytical Chemistry*) contained just five references; three of them were to their precedent works, without any reference to the vast literature on the problem in the nuclear physics field, thus revealing a complete ignorance of the previous work. Moreover, they did not give sufficient details to reproduce their experiment nor adequate proofs that their results were the effects of systematic trials. After some time, it turned out that they forgot some basic procedures in their experimentation and that the extraordinary results they claimed to have achieved were not repeatable. In the news press, which unusually took place before the publication of their paper, they declared to have worked on cold fusion in secret during the preceding five years. Also this particular shows that their behavior was not experimentally nor scientifically sound. Science as experimental activity cannot be conducted in isolation. In the achievement of experimental results, other scientists' comments and critiques have a fundamental role to revise work and to check its validity.

Justification and Explanation

This principle deals with the drawing of well-justified conclusions on the basis of all the information collected during an experiment. In an experimental procedure, it is not sufficient to collect as much precise data as possible, but it is necessary to look for an explanation of these data. Therefore, not only the drawn conclusions must be strongly supported, but also all the data from an experiment should be interpreted in order to derive the correct implications.

Usually experiments in physics are considered 'good' when they are grounded in existing theories and use an apparatus that measures the quantities of interest with sufficient accuracy and precision [21]. In this case, experiments are theory laden in that the terms and the used apparatus are dependent on existing theory. One of the important roles of experiments in physics is to test theories [22]: as already discussed, an experiment can confirm an existing theory or can show that a new theory is required, either by showing that an accepted theory is incorrect or by exhibiting a new phenomenon that calls for an explanation. Therefore, the well-founded and comprehensive theories of physics provide the conceptual framework in which experiments are designed and realized. At the same time, theories are tested by experiments that, in some cases, can show that they are inadequate or incomplete. The difficult part concerns the fact that experiments may not always give clear-cut results: the so called *crucial experiments* [18], those that quickly decide in a definitive way between two or more competing theories, are exceptions. In all other cases, experiments are much more difficult to interpret and, as a consequence, complex to explain.

Physicists, however, have a reasonable belief in experiments and in their results: how is that possible? This question has been addressed by many philosophers of science. Among those, Hacking [29] proposed an elaborated answer, setting out a number of strategies for believing in experiments. For example, he stresses the crucial role of *intervention* in the experimental practice. He considers in particular the case of microscopes used in experiments [28]. For instance, in looking at a cell through a microscope, one may inject some fluid into the cell, in order to change the cell color when the operation is done. If one actually observes the predicted effect of this intervention, his/her beliefs in the proper operation of the experimental apparatus and in the data collected with it are strengthened. Of course this cannot be the only strategy. *Independent confirmation*, namely that the same data can be collected by different microscopes, is another strategy that may be combined with the first one. But what happens when an experiment can be performed with only one type of apparatus or when the intervention is very difficult, if not impossible? In such cases other strategies for trusting experiments are possible. The experimental apparatus can reproduce known phenomena such that, in case of success, the belief in its proper working is enhanced. Other strategies include: the reproduction of artifacts that are known in advance, the elimination of alternative explanations of the results, the use of the results themselves to argue for their validity, the use of an independently well-corroborated theory

of the phenomenon to explain the results, the use of an apparatus based on a well-corroborated theory, the adoption of statistical arguments. Of course it is very unlikely that these strategies can be adopted all together; usually just some of them are at disposal. They provide good reasons for believing in experimental results, even if they do not guarantee that the results are always correct. In the history of science, there are many cases in which these strategies have been adopted, but experimental results eventually turned out to be incorrect. Experiments are *fallible*: this does not mean that it is not possible to reach well-justified conclusions, but that these conclusions are not guaranteed once and for all.

3 Computing and Experiments

Let us critically review what has been said about experiments in computing. We will see that such analysis sheds light on some open methodological issues in the characterization of the discipline, which we propose to tackle with conceptual instruments provided by the history of science, the philosophy of science, and the philosophy of technology.

3.1 Calling for Experiments in Computing

Probably one of the first and most famous concepts of computer science as an experimental science goes back to the 1976 paper by Newell and Simon published in the occasion of their acceptance of the Turing award [45]. "Computer science is an empirical discipline. We would have called it an experimental science, but like astronomy, economics, and geology, some of its unique forms of observation and experience do not fit a narrow stereotype of the experimental method. None the less, they are experiments. Each new machine that is built is an experiment. Actually constructing the machine poses a question to nature; and we listen for the answer by observing the machine in operation and analyzing it by all analytical and measurement means available. Each new program that is built is an experiment. It poses a question to nature, and its behavior offers clues to an answer. Neither machines nor programs are black boxes; they are artifacts that have been designed, both hardware and software, and we can open them up and look inside. We can relate their structure to their behavior and draw many lessons from a single experiment. We don't have to build 100 copies of, say, a theorem prover, to demonstrate statistically that it has not overcome the combinatorial explosion of search in the way hoped for. Inspection of the program in the light of a few runs reveals the flaw and lets us proceed to the next attempt."

It is easy to recognize in these words some of the characterizing features of the experimental method in traditional scientific disciplines, first of all the idea of experiment as a question to nature. This conception of machines and programs as experiments has been influential for many years, promoting the

idea that the appeal to experience is fundamental in contrast with the view of computer science as a pure mathematical and deductive discipline. However, the rather ingenuous view on experiments, without any more specific reference to some principles of experimentation, may have contributed to spread an oversimplified conception of how the experimental method can be applied to computing.

The quest for experiments in computing began to be treated systematically at the beginning of the 1980s, following a crisis in what was then called *experimental computer science*. In an ACM report published in 1979 [33], experimental research in computer science is strongly related to the measurement and testing of computing algorithms and systems. In the same issue of the journal [42] where the ACM report was published, the call for experimentation is expressed in terms of the recognition of the possibility for major advantages in different fields of computing. At the same time, a 'rejuvenation' of experimental computer science is advocated from very concrete perspectives: for example by promoting experimental facilities for computer systems research. However, experimental computer science is seldom defined in a precise way in this context, and experiments are conceived mainly as *explorations*.

Experimental computer science is to be rejuvenated also according to Peter Denning, who proposes in a short article that the experimental work produced in computer science should be judged by *traditional* standards [15]. Denning advances the idea that to implement experimentally a computer system is not just to build the system and "see what happens". In a way, this approach tries to go beyond the 'construct and test' paradigm of Newell and Simon, by proposing that experimental computer science has to deal with the process of supporting and testing a hypothesis, thus making computing closer to the standards of rigor and the practice of traditional sciences. Unfortunately, experimental science in general is reduced to the process of classifying knowledge derived from observations. Denning concludes with a programmatic proposal: traditional experimental criteria should be applied in evaluating the results of experimental computer science.

Although some efforts along this direction have been put to work over the years [13] [52] [37], the above mentioned guidelines, whether reductive or not, remained mostly unattended.

More recently, a trend has once again emerged toward making the experimental scientific method take center stage in computing. These recent efforts in several projects have shown a renewed need for an experimental methodology in this discipline [24] [43]. Experiments are deemed to have an impact on several aspects of computing: their importance is recognized for assessing computing systems' performance and for triggering new developments [13] [24] [43] [52], and experimentation with prototypes is considered essential in use-inspired research and product design [49]. Moreover, the use of the experimental scientific method is advocated to understand computations that are often too complex for mathematical analysis, to prove their correctness, to check consistency with hypotheses, to uncover performance constraints, and to show whether original goals are met [11].

'Experimental computer science' has become a quite common label to which today (at least) three different meanings can be associated [20]. The first one refers to the type of research devoted to the realization of concrete systems; this kind of activity lies in the realm of engineering rather than science, and thus its experimental side is related to the demonstration of the feasibility of these systems, whether software or hardware. The second meaning [16] views experimental computer science as the mathematical modeling of the behavior of complex systems, where the anticipated properties of the systems have to be tested experimentally. This 'experimental feedback' is recognized elsewhere as the underlying force of computing [45]. The third meaning defines the discipline as the evaluation of computer systems by means of the traditional methodologies of natural sciences [52], such as repeatability.

Despite the increasing interest in a more rigorous methodological approach to computing, many lament that the current methodology is inadequate and that, in comparison with other fields (e.g. natural sciences), computer scientists should experiment more [12]. Indeed, several articles describe demonstrations rather than real experiments [20], and their sections on experimental results present just weak examples to show the superiority of the proposed solution over a limited amount of alternatives [58] [59] [1].

The invitation to experiment more and better is not always based on convincing arguments. From the one side, it is generally claimed that "without experiments, computer science is in danger of drying up and becoming an auxiliary discipline" [52] and that experimentation can accelerate progress, but no precise analysis of how this experimentation should be carried out in the various fields of computing is provided. On the other side, the adoption of a scientific (in the sense of experimental) approach in computing is justified by the tremendous success of scientific reasoning. Still, no serious inquiry is carried out about the possibility and the consequences of plainly importing traditional scientific methods into computing [43].

The call for an increase in experimentation has often been made with an emphasis on making computing more scientific. However, this 'scientificity' is taken as a positive goal *per se*, without any discussion on whether computing can really benefit from being more scientific, or even on what it really means 'to be more scientific.'

Despite this lack of strong foundations, many recommendations (see [5] [32] [4] [54] [1] [40]) present common traits: they stem from the acknowledgment of a crisis in computing that is meant to be overcome with a greater maturity of the discipline, in terms of a more rigorous experimental method and a more scientific approach to the search for solutions. The title of one of these works is meaningful under this respect: *Pushing Science into Signal Processing* [3]. Taking inspiration from experimental principles adopted in traditional scientific disciplines has become a leitmotif in many analyses, which recognize, for example, the benefits of replicable results [34] or the importance of negative results [3], two of the cornerstones of experimental scientific method. Still, many issues remain open and require, as we argue in the next section, a shift from general statements to concrete discussions in computing to clarify the

advantages and the limits of taking inspiration from more traditional scientific disciplines.

3.2 Open Issues

No clear indication has been given on what experiments in computing are supposed to be like: as hinted above, many recommendations seem to be problematic from one side or another. Let us illustrate in more detail what we consider the most recurring issues, namely, the *lack* of *conceptual clarity* and *concrete examples*, and *neglecting the engineering component* of computing. We argue that the instruments to tackle them may be provided by the history of science, the philosophy of science, and the philosophy of technology.

Some proposals are weakened by a lack of clarity when it comes to the core concepts of experimental methodology. This lack of conceptual clarity could be improved by keeping trace on how these concepts have been conceived in the history and philosophy of experimentation.

First of all, an accurate view on the scientific method itself is missing in some of the works stressing the importance of 'scientificity' in computing. A clear example can be found in [12], where the scientific method is traced back to Francis Bacon, without any mention to Galileo Galilei, usually acknowledged as the founding father of the scientific experimental method.

Moreover, empirical methods and experimental methods are often confused. In some works (for instance in [43]) experiments are identified with empirical methods, which are instead traditionally defined as based on the aggregation of naturally occurring data, without the strict rules that are supposed to guide experiments. Even when a distinction between empirical and experimental is made, the notion of experiment is sometimes naively intended and poorly related to the traditional view of empirical sciences, like in the already quoted passage by Newell and Simon [45] (see Subsection 3.1). The authors, by writing "Computer science is an empirical discipline. We would have called it an experimental science, but" show to believe that 'empirical' and 'experimental' are two distinct concepts, but the discourse gets less clear when they claim that computer science, no matter how artificial or synthetic its subject can be, can be done through empirical enquiry as any other science, thus implying that they identify 'empirical' with 'experimental'.

Even works that are considered a milestone in the development of experimental methods in software engineering like [35], while discussing in detail how experiments should be intended in this field, present some ingenuous and simplistic views on experimentation. Examples that hint at a conceptual framework still at the very early stages of development, with an oversimplified view of the complex relationship between theory, experiments and reality, are the following: "the purpose of experimentation is to match ideas with reality" (p.12), "a particular item of knowledge is considered valid if it has been checked against reality. Scientific progress is founded on the study and settlement of discrepancies between knowledge and reality.[...] Scientific investigations are

objective studies, based on observations of or experimentation with the real world and its measurable changes" (p.24).

There are of course cases in which experiments are defined within a more solid conceptual framework including observation, measurement, and analysis of results [24]; however, in computing, these processes are seldom illustrated in terms of *concrete examples* similar to what natural scientists deal with in their practice.

The acknowledgment of the relevance of scientific experimentation goes hand in hand with the awareness of the limits of analytical methods. The so called 'scientific paradigm' of computer science [19] is characterized by the general claim that the methods of computer science are the methods of the natural sciences, thus pairing deduction with empirical validation. However, when mentioning experiments and scientific experimentation in computing, several works fail to address how these can be applied in this field. "The methods of computer science must combine deductive reasoning with scientific experimentation" [19], but what does this mean in concrete?

As computing has different goals and aims than experimental physics or biology, do experimental principles remain the same? Does it make sense to 'import' traditional experimental principles (comparison, reproducibility and repeatability, justification and explanation) from other scientific disciplines into computing? On the one side, the application of these principles allows for a more rigorous approach to experiments within the field; on the other side, these principles are valid for disciplines (such as physics and biology) that aim at understanding and explaining natural phenomena, whereas computing is focused on the abstract concept of computation and the creation of technological artifacts that implement such concepts.

There are of course various limits in taking inspiration from some general experimental principles holding in scientific disciplines that have very different goals than computing. One drawback is given by the fact that these principles are not applicable in general, but they need to be specifically declined in every situation in the various subfields of computing. Unfortunately few works seem to be interested in investigating what this means from a practical point of view.

Exceptions are represented by some works [4] [54] discussing how reproducibility can be put into the practice of signal processing. In particular, [54] distinguishes among three levels of reproducibility, namely reproducibility of the *algorithm*, of the *code*, and of the *data* and concludes that algorithms are generally well described and compared with each other, whereas implementation details and data are provided only in a very small number of cases. Moreover, this concrete analysis of rigorously reproducing experimental results is paired with more general considerations on how the relevant research community should be reframed to promote these issues, for example by publishing also negative results, or by promoting special sessions at conferences and/or journals for all-experimental, no-novelty works.

Another interesting example of how experimental principles could be concretely translated in computing is represented by some recent trends in au-

tonomous mobile robotics [1]. Here, a concrete way to promote comparison can be seen in the increased use of publicly available data sets [27] that provide a common ground for comparing with more rigor the performances of different systems. However, these comparisons are still just qualitative in most cases (they often boil down to showing simply that a system is performing better than others) and with poor attempts to explain and justify these results [2].

The proposals that are endowed with a discussion on concrete examples show that the application of general experimental principles is not straightforward and immediate, and reveal that differences with traditional sciences are not due only to the relatively young age of computing, but also to its intrinsically different disciplinary nature, scientific and, at the same time, engineering.

Neglecting the engineering component of computing is fairly common in the current literature claiming for more and better experimentation in it (see for example [52]). Computing is treated along the same line as more traditional scientific disciplines, such as physics or biology, and the current approach is to simply import experimental principles from one field to another.

We claim that the engineering component of computing should be considered when reflecting on the nature and role of experiments in computing and that the notion of *technical artifact*, as conceived in the 'analytic' philosophy of technology [23], could be a useful tool. According to [55], engineering is an activity aimed at producing technology, and technology is a practice focused on the creation of artifacts and artifact-based services. Let us consider for instance the purposes of experimentation in autonomous mobile robotics [39]. Robotic systems are human-made artifacts with a technical function, which seems to bring the discipline closer to engineering. Accordingly, experiments in autonomous mobile robotics have the goal of demonstrating that a given artifact is working in some way or that it is better than another. However, the most advanced autonomous robotic systems are so complex that their behavior is hardly predictable, even by their own designers, especially when considering their interaction with the physical world. In this perspective, experiments in autonomous mobile robotics are somehow similar to experiments in natural sciences since, broadly speaking, both have the goal to understand how complex systems work.

Similarly experiments in software engineering present this double nature. They are aimed at establishing criteria that help make the best choices regarding all the factors involved in the creation of software, such as the requirement specification language, the team of programmers, the programming paradigm and language, and so on. All the relevant activities are heavily human-centered, and following a rigorous method to do research on this kind of practice becomes particularly critical. In [7], for example, the performance of human subjects is measured with chronometers and eye-tracking devices to verify which of four competing graphic notation standards provides the best support to software developers in the task of recognizing design patterns in the architecture of a program. An experiment like this can be viewed as having a two-fold purpose: an investigation on the effects of certain information visualization techniques

on humans, and an analysis in search for the best practice in one aspect of the complex software development process.

These examples reflect the peculiar position of computing at the intersection between engineering and science. There is a deep difference between making research on naturally occurring phenomena and artificial ones and this surely has an impact on computing, which deals with both kinds. Experiments about artifacts in computing tell us more about the people that have done the job, than the way the world is [51]. At the same time, these artifacts need to be tested keeping in mind the technical function and use plan they have been designed for. One could ask why we need to test artifacts that we have created: this is due to a sort of unpredictability arising both at the level of the artifact and at the level of its interaction with the physical environment (including the persons) surrounding it.

With these considerations in mind, we hope to have made clear that calling for experiments in computing does not equate with making computing more similar, at least from a methodological point of view, to traditional scientific disciplines. The debate on the nature of this discipline must widen its scope to take into account the apparent features that computing acquires from its aim at creating technical artifacts. The coexistence of scientific observation and engineering pragmatism and their mutual influence call for a rethinking that goes outside the scope of computing alone and involves questions that have long been discussed in the context of the philosophy of science and may be renewed in the philosophy of technology.

4 Conclusions

In this paper we have added some steps to the debate on the disciplinary status of computing by investigating such question with an analysis of the nature of its experiments. As computing is characterized by different areas, the experimental perspective plays a role only in those ones which are more similar to empirical scientific disciplines, devoted to the discovery of the unknown.

Firstly, we have reiterated that the discipline of computing has a peculiar character. Although this may seem like old news, we have shown that this peculiarity emerges also from a methodological point of view. As a consequence, a plain application of the classical experimental protocol seems not possible and new experimental protocols are to be developed. If in the natural sciences it is prescribed that the experimenter should be an outsider of the phenomenon to be explained, it is not clear how a person working in computing, which is aimed at producing computation-based artifacts, could be an outsider with respect to a phenomenon (i.e. an artifact) that he or she has created [51].

Nevertheless, traditional experimental principles can provide some useful inspiration for computing: the rigor they add is not a desirable property *per se*, but it can help in the progress of a discipline and in the construction of its status (even a very peculiar one as that of computing). This does not mean

that computing must be considered a science, but that researchers should recognize that rigorous methods are better than do-it-yourself approaches.

The importance of developing unique and specific experimental protocols in computing (starting from science but moving beyond it) is further emphasized when considering the position of this discipline at the intersection between science and engineering with a particular focus on its experimental activity. Experiments are performed to test artifacts and therefore the use function of these artifacts play a major role in experimentation; at the same time, experiments are made to understand better how these complex artifacts behave and interact with the environment, or may increase the experimenter's knowledge about phenomena occurring in the environment, like in traditional scientific disciplines, although the human factor intervenes in a significant way.

Secondly, in this paper we have presented a survey of computing and experiments and we have singled out some open issues. We have argued that to deal successfully with them, which are mainly due to a deep conceptual uncertainty, the history of science, the philosophy of science, and the philosophy of technology should be taken into account.

Philosophers have traditionally refrained from getting their hands dirty with technology. Philosophers of science have developed a paradigm of science that has been strongly influenced by their championing of physics. Things do not seem to be much different nowadays, but some counterexamples like the analytic philosophy of technology are on the rise. We consider it a very promising path, because in the light of what we have said, it is rather clear that the debate on the disciplinary nature of computing goes well beyond its boundaries, and given its undeniable engineering component, the discussion appears to be a topic well worth investigating in the context of the philosophy of technology.

Acknowledgements This work was partially supported by the EC within the 7th Framework Programme, under grant agreement 257129 (PoSecCo), and by the MIUR in the framework of the PRIN project Gatecom.

References

1. F. Amigoni, M. Reggiani, and V. Schiaffonati. An insightful comparison between experiments in mobile robotics and in science. *Autonomous Robots*, 27(4):313–325, 2009.
2. F. Amigoni, V. Schiaffonati, and M. Verdicchio. An analysis of experimental trends in autonomous robotics papers. In *IEEE International Conference on Robotics and Automation (ICRA2012) Workshop on the Conditions for Replicable Experiments and Performance Comparison in Robotics Research*. 2012.
3. M. Barni and F. Perez-Gonzalez. Pushing science into signal processing. *IEEE Signal Processing Magazine*, 120:119–120, 2005.
4. M. Barni, F. Perez-Gonzalez, P. Comesana, and G. Bartoli. Putting reproducible signal processing into practice: A case study in watermarking. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, 2007*, pages 1261–1264, 2007.
5. V. R. Basili and L. C. Briand, editors. *Empirical Software Engineering*. Springer.
6. F. P. Brooks Jr. The Computer Scientist as Toolsmith II. *Communications of the ACM*, 39(3):61–68, 1996.

7. G. Cepeda Porras and Y. G. Guéhéneuc. An empirical study on the efficiency of different design pattern representations in UML class diagrams. *Empirical Software Engineering*, 15(5):493–522, 2010.
8. T. Colburn. *Philosophy and Computer Science*. M.E. Sharpe, 2000.
9. T. R. Colburn. Methodology of computer science. In L. Floridi, editor, *Philosophy of Computing and Information*. Blackwell Publishing, 2004.
10. A. Cromer. *Uncommon Sense*. Oxford University Press, New York, 1993.
11. P. Denning and P. Freeman. Computing's paradigm. *Communications of the ACM*, 52(12):28–30, 2009.
12. P. J. Denning. Is computer science science? *Communications of the ACM*, 48(4):27–31, 2005.
13. P. J. Denning, D. E. Comer, D. Gries, M. C. Mulder, A. Tucker, A. J. Turner, and P. R. Young. Computing as a discipline. *Communications of the ACM*, 32(1):9–23, 1989.
14. P. J. Denning and P. S. Rosenbloom. Computing: The fourth great domain of science. *Communications of the ACM*, 52(9):27–29, 2009.
15. P.J. Denning. What is experimental computer science? *Communications of the ACM*, 23(10):543–544, 1980.
16. P.J. Denning. ACM's president letter: Performance analysis: Experimental computer science at its best. *Communications of the ACM*, 24(11):725–727, 1981.
17. P.J. Denning. Computing as a natural science. *Communications of the ACM*, 50(7):13–18, 2007.
18. P. M. Duhem. *The Aim and Structure of Physical Theory*. Princeton University Press, 1991.
19. A. H. Eden. Three paradigms of computer science. *Minds and Machines, Special issue on the Philosophy of Computer Science*, 17:135–167, 2007.
20. D. G. Feitelson. Experimental computer science: The need for a cultural change. 2006.
21. A. Franklin. What makes a 'good' experiment? *British Journal for the Philosophy of Science*, 32(4):367–379, 1981.
22. A. Franklin. Experiment in physics. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2009.
23. M. Franssen, G. J. Lokhorst, and I. van de Poel. Philosophy of technology. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2010.
24. P. Freeman. Back to experimentation. *Communications of the ACM*, 51(1):21–22, 2008.
25. P. Godfrey-Smith. *Theory and reality: An introduction to the philosophy of science*. The University of Chicago Press, Chicago and London, 2004.
26. M. Grosser. *The Discovery of Neptune*. Harvard University Press, 1962.
27. J. Guivant, E. Nebot, and S. Baiker. Autonomous navigation and map building using laser range sensors in outdoor applications. *Journal of Robotic Systems*, 17(10):565–583, 2000.
28. I. Hacking. Do we see through a microscope? *Pacific Philosophical Quarterly*, 62:305–322, 1981.
29. I. Hacking. *Representing and Intervening. Introductory Topics in the Philosophy of Natural Science*. Cambridge University Press, 1983.
30. J. Hartmanis. Turing award lecture on computational complexity and the nature of computer science. *Communications of the ACM*, 37(10):37–43, 1994.
31. J. Hartmanis and H. Lin, editors. *Computing the Future: A Broader Agenda for Computer Science and Engineering*. National Academy Press, 1992.
32. IEEE. *Proceedings of First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, Madrid, 2007.
33. J.A.Feldman and W.R. Sutherland. Rejuvenating experimental computer science. *Communications of the ACM*, 22(9):497–502, 1979.
34. N. Juristo and O. Gomez. Replication of software engineering experiments. In B. Mayer and M. Nordio, editors, *Empirical Software Engineering and Verification*, pages 60–88. Springer, 2012.
35. N. Juristo and A.M. Moreno. *Basics of Software Engineering Experimentation*. Kluwer, Dordrecht, Germany, 2001.
36. D.E. Knuth. Computer programming as an art. *Communications of the ACM*, 17(12):667–673, 1974.

37. P. Langley. Machine learning as an experimental science. *Machine Learning*, 3:5–8, 1988.
38. M. Loui. Computer science is a new engineering discipline. *ACM Computing Surveys*, 27(1), 1995.
39. R. Madhavan, C. Scrapper, and A. Kleiner. Special issue on characterizing mobile robot localization and mapping. *Autonomous Robots*, 27:309–481, 2009.
40. B. Mayer and M. Nordio. *Empirical Software Engineering and Verification*. LNCS. Springer, 2010.
41. J. McCarthy. Computer programs for checking mathematical proofs. *Amer. Math. Soc. Proc. of Symposia in Pure Math*, 5, 1962.
42. D.D. McCracken, P.J. Denning, and D.H. Brandin. An acm executive committee position on the crisis in experimental computer science. *Communications of the ACM*, 22(9):503–504, 1979.
43. C. Morrison and R. Snodgrass. Computer science can use more science. *Communications of the ACM*, 54(6):38–43, 2011.
44. A. Newell, A. J. Perlis, and H. Simon. Computer Science. *Science*, 157:1373–1374, 1967.
45. A. Newell and H. Simon. Computer science as empirical inquiry: Symbols and search. *Communications of the ACM*, 19(3):113–126, 1976.
46. H. Radder, editor. *The Philosophy of Scientific Experimentation*. University of Pittsburgh Press, 2003.
47. S.C. Shapiro. Computer science: The study of procedures. http://www.cse.buffalo.edu/ shapiro/Papers/whatiscs.pdf, 2001.
48. H. Simon. *The sciences of the artificial*. MIT Press, 1981. (2nd edition).
49. M. Snir. Computer and information science and engineering: One discipline, many specialties. *Communications of the ACM*, 54(3):38–43, 2011.
50. M. Tedre. Computing as engineering. *Journal of Universal Computer Science*, 15(8):1642–1658, 2009.
51. M. Tedre. Computing as a science: A survey of competing viewpoints. *Minds & Machines*, 21:361–387, 2011.
52. W. Tichy. Should computer scientists experiment more? *IEEE Computer*, 31(5):32–40, 1998.
53. T.S.Kuhn. *The Copernican Revolution. Planetary Astronomy in the Development of Western Thought*. Harvard University Press, 1957.
54. P. Vandewalle, J. Kovacevic, and M. Vetterli. Reproducible research in signal processing. *IEEE Signal Processing Magazine*, 37:37–47, 2009.
55. P. Vermaas, P. Kroes, I. van de Poel, M. Franssen, and W. Houkes. *A Philosophy of Technology. From Technical Artefacts to Sociotechnical Systems*. Morgan and Claypool, 2011.
56. P. Wegner. Research paradigms in computer science. In *ICSE '76: Proceedings of the 2nd International Conference on Software Engineering*, pages 322–330, Los Alamitos, CA, 1976. IEEE Computer Society Press.
57. Richard Westfall. *The Construction of Modern Science. Mechanisms and Mechanics*. John Wiley & Sons Inc., 1971.
58. M. V. Zelkowitz and D. R. Wallace. Experimental validation in software engineering. *Information and Software Technology*, 39(11):735–743, 1997.
59. M. V. Zelkowitz and D. R. Wallace. Experimental models for validating technology. *Computer*, 31(5):23–31, 1998.