'A Pretence of What is Not?'

A Study of Simulation(s) from the ENIAC Perspective

Liesbeth De Mol

What is the significance of high-speed computation for the sciences? How far does it result in a practice of simulation which affects the sciences on a very basic level? To offer more historical context to these recurring questions, this paper revisits the roots of computer simulation in the development of the ENIAC computer and the Monte Carlo method.

With the aim of identifying more clearly what really changed (or not) in the history of science in the 1940s and 1950s due to the computer, I will emphasize the continuities with older practices and develop a two-fold argument. Firstly, one can find a diversity of practices around ENIAC which tends to be ignored if one focuses only on the ENIAC itself as the originator of Monte Carlo simulation. Following from this, I claim, secondly, that there was no simulation around ENIAC. Not only is the term 'simulation' not used within that context, but the analysis also shows how 'simulation' is an effect of three interrelated sets of different practices around the machine: (1) the mathematics which the ENIAC users employed and developed, (2) the programs, (3) the physicality of the machine. I conclude that, in the context discussed, the most important shifts in practice are about rethinking existing computational methods. This was done in view of adapting them to the high-speed and programmability of the new machine. Simulation then is but one facet of this process of adaptation, singled out by posterity to be viewed as its principal aspect.

Keywords: ENIAC, Computational practice, Programming, Simulation

„A pretence of what is not“? Eine Untersuchung von Simulation(en) aus der ENIAC-Perspektive

Wie relevant sind Hochgeschwindigkeitsrechner für die Naturwissenschaften? In wie weit führen sie zu einer Praxis der Simulation, die die Naturwissenschaft auf fundamentale Art und Weise verändert? Mit dem Ziel, den historischen Kontext für die Diskussion dieser oft gestellten Fragen zu erweitern, wird in diesem Beitrag der Ursprung von Computersimulation in der Entwicklung des Electronic Numerical Integrator and Computers (ENIAC) und der Monte-Carlo-Methode wiederaufgegriffen. Um besser identifizieren zu können, was sich in der Wissenschaft der 1940er und 1950er Jahren aufgrund des Computers geändert hat, betont mein Ansatz die Kontinuität zu älteren Praktiken und stellt zwei unterschiedliche, jedoch miteinander verbundene Thesen auf. Erstens ist im Umfeld des ENIAC eine Vielfalt von Praktiken zu finden, die oft übersehen werden, wenn man den ENIAC allein als den Ursprung der Monte-Carlo-Simulation betrachtet. Zweitens werde ich argumentieren, dass es um den ENIAC herum keine Simulation gab. Nicht nur wurde der Begriff „Simulation“ im ENIAC Kontext nicht verwendet, sondern diese Analyse zeigt auch, dass es sich bei der „Simulation“ um eine komplexe Palette von Praktiken handelte, verschiedene Ebenen betreffend: (1) die verwendete und entwickelte Mathematik; (2) den Code; (3) die Materialität der Maschine. Meine zentrale Schlussfolgerung ist, dass sich im untersuchten Kontext die wichtigsten Verschiebungen aus dem Neudenken bereits bestehender Computermethoden ergaben. Dies geschah mit der Absicht, die Methoden der hohen Geschwindigkeit und Programmierbarkeit der Maschine anzupassen. Simulation ist somit nur eine der vielen Facetten in diesem Prozess der Anpassung, die erst im nach hinein herausgehoben wurde, um als sein wichtigster Aspekt zu gelten.

Schlüsselwörter: ENIAC, Rechenpraxis, Programmierung, Simulation

The past went that-a-way. When faced with a *totally* new situation, we tend always to attach ourselves to the objects, to the flavor of the most recent past. We look at the present through a rear-view mirror. We march backward into the future. (Marshall McLuhan 1967)

Introduction. 'Es gibt keine Simulation'

What, if anything, is the impact of technology x on science y ? While examples of questions of this type have long been neglected within more traditional history and philosophy of science, the steady reversal of the primacy of science over technology since the 1980s (Forman 2007) has given these questions new relevance and put them into a methodological framework where the focus is now much more on how certain technological advances affect and shape scientific practice and the knowledge it produces. This focus on technology is, in itself, a historical phenomenon and is partially rooted in the growing dependencies between certain branches of science and increasingly complex, and often expensive, technologies that an expertise which is not necessarily that of the scientist or team of scientists using the technology.

One important driving factor in this development has been the possibility of high-speed computation. Today most technological complexes require an intricate computational infrastructure which is used in a way which was simply not possible before the development of high-speed computing.

It is from this context that one can understand the recent increased interest in so-called computational science and the changing role and place of computer science in the disciplinary spectrum from a discipline struggling for independence to one which has been identified by some as an entirely new scientific domain on a par with the life, physical and social sciences (Tedre 2015). Thus, one important instance of the above question is:

What, if anything, is the effect of high-speed computation on science y ?

The current historical (and philosophical) literature which deals with this question focuses for the most part on one aspect of this question, viz. the impact of computer simulation (Hashagen 2013).¹ One of the standard historical references in this context is Galison's work on Monte Carlo methods and how they required as a simulation method, the constitution of a "trading zone" where different practices (including technological and

scientific practices) are locally coordinated by relying on a so-called pidgin language (Galison 1997; Galison 2011). But while Galison's work is without doubt a fundamental contribution to the field, it is in need of a revision (see also Borrelli 2019). Indeed, since Galison, hardly any work has been done to reconsider this almost mythical beginning of computer simulation in the ENIAC context. In the meantime, in the more philosophical and epistemological literature, it has become clear that 'computer simulation' does not have one stable meaning but covers a number of different understandings resulting in a set of different philosophical assumptions on their supposed epistemological significance and so different answers to the question (Duran 2018; Hartmann 1996; Humphreys 2004):

What, if anything, is the impact of (computer) simulation on science y?

One such topic, for instance, is the question of the relationship between 'traditional' experiments and computer simulations. The focus of these more philosophical works is on how computer simulation challenges (or not) more classic issues within the philosophy of science and so gives precedence to science rather than to technology. Moreover, these works often lack an *in-depth* historical perspective and assume a kind of stability of concepts over time.² One notable example of this is the fact that many philosophical discussions on simulation focus on the philosophical novelty of computer simulation (or lack thereof) while at the same time ignoring the existing historical work which attacks an idea of the history of computing which has one singular starting point and which also emphasizes the continuities with older practices in order to identify more clearly what really changed in the history of science in the 1940s and 1950s due to the computer (and which lay the basis for later work).

The aim of this paper is to revisit the roots of computer simulation in the ENIAC to contribute to the historical perspective. I will do this by developing two related but distinct claims. First I will show how, around the ENIAC, a diversity of practices developed which tend to be ignored if one focuses *only* on the ENIAC as the originator of Monte Carlo simulation. This broadens the outlook on 'simulation' in those pioneering years which thus has different forms not just chronologically but also diachronologically. Second, I develop the claim that around the ENIAC there is no simulation,³ both in an historical as well as in a methodological sense. From the historical perspective, it should be pointed out that the term of simulation was not really used in the ENIAC context.⁴ Secondly, and more important, the idea that there is no simulation is also taken as a methodological approach in this paper: instead of focusing *on what is being simulated* (weather; neutron behavior; bombing behavior; etc.) the approach here is to look at *what lies behind and makes 'simulation' possible* and thus

show how it was structured by the interrelations between three fundamental levels of the practice in which it emerged:

1. the mathematics used and developed (human)
2. the (logical organization of the) programs (human-machine)
3. the physical set-up of the machine (machine)

In order to develop these claims, this paper is structured in two main sections. In the first section, I introduce the ENIAC and engage with some of the reasons for constructing the machine. This allows me to highlight some of the continuities and discontinuities with previous calculatory practices. In the second section then I engage with the work of three different ENIAC “users”: Derrick H. Lehmer, Haskell B. Curry and John von Neumann. In a discussion section I will return to the main claims and conclude with a critical viewpoint on “simulation” from the contemporary perspective.

Introducing the ENIAC—Historical Setting

The ENIAC (Electronic Numerical Integrator and Computer) is probably the most famous and, at the same time, most controversial computer that has ever been built. As well as being presented as (one of) the first electronic computers, it also played a central role in the important *Honeywell, Inc. vs. Sperry Rand Corporation* case which invalidated the ENIAC patent that was filed by Eckert and Mauchly, the two main engineers involved with the design and construction of the machine.

The ENIAC and other contemporaneous machines can be contextualized in a broader history of mechanization of human calculation in applied mathematics and science in general (Grier 2005). More particularly, it fits into a history of calculatory practices for making and using numerical tables (for instance, the use of difference engines to mechanize the method of finite differences) (Campbell-Kelly et al. 2003; De Mol & Durand-Richard n.d.).

Within that context, it is important to remember that human computation was laborious, time-consuming and error-prone.⁵ For instance, Charles Babbage, whose work on and design of the difference and analytical engines is often seen as a precursor to the modern computer, supposedly and famously exclaimed: “I wish to God these calculations had been executed by steam” when he and Herschel were checking a manuscript of calculations for astronomical tables and realized the rather large number of errors (Swade 2003:158). It was the increased need for such tables com-

bined with the growing realization of the inherent problems surrounding them that resulted in the development of more 'efficient' methods of calculation. One development concerns "deskilling" methods (Swade 2003: 150) which were introduced to simplify the calculatory process in such a way that the calculations could be done by people of a lower level of education. The other is the development of mechanical aids which included digital machines (for instance, difference engines or Aiken's Mark I) and so-called 'analog' devices (for instance, the differential analyzer). The latter machines were used to solve differential equations and remained a preferred tool in engineering for many years after the introduction of the high-speed digital computer (Ramunni 1989).

It was then mostly the issues of speed and error which resulted in the U.S. Army accepting Mauchly's proposal to construct a high-speed electronic machine. In the 1930s there was a growing need for ballistic tables used to aim fire at an enemy target. These tables involved complex calculations because of the large number of factors that affect the trajectory of a missile like, for instance, wind velocity, and the shell's weight, diameter and shape. The Ballistics Research Lab (BRL) at the Aberdeen Proving Ground, Maryland had, as one of its main tasks, the computation of such ballistic tables for the U.S. army. The Aberdeen Proving Ground was the first U.S. Proving Ground and was constituted just a few months after the U.S. entered World War I. It was here that weapons were designed and tested and so it played a major role in bringing mathematics and military applications closer.

The computation of firing tables relied on both hand calculations aided by desk calculators and computations from the differential analyzers at the BRL and the Moore School of Electrical Engineering at the University of Pennsylvania, depending on the type of trajectory to be computed. However, each of these methods had its shortcomings. With the hand calculation method, it took about two 8-hour days to calculate one trajectory; the differential analyzer was much faster in that respect, taking about 15–30 minutes for one trajectory (Polachek 1997), but the solutions were not accurate and needed to be 'smoothed' by additional hand calculations. Moreover, it took on average one day to set up the differential analyzer to change from one type of trajectory to another (Polachek 1997; Grier 2005).

Given these circumstances, the production speed of the ballistic tables could not keep up with the need for new tables which increased dramatically when the USA entered the Second World War and so Mauchly's proposal to the Army to construct a high-speed computer to compute firing tables was perfectly timed. And so, in 1943, the construction of the ENIAC was started at the Moore School, Penn University. It was to take until 1946, after the war, before the machine was completed. Even though

it was too late to serve its initial purpose, it was realized that the ENIAC could be used for a host of other applications, including the famous Monte Carlo calculations used in the design of nuclear weapons. According to the list made by Barkley Fritz, the ENIAC was used for over one hundred different problems (Fritz 1994), from the production of number-theory tables to weather prediction.

The machine had, in a sense two “lives” (Neukom 2006). In a first stage, it was modular and ran computations in parallel. In order to ‘program’ it, it was necessary to rewire the machine by reconnecting the different units and setting-up a large number of switches, see (Bullynck & De Mol 2010) for more details. Preparing and setting-up a program on the original ENIAC was a very time-consuming job. Moreover, the ‘length’ and complexity of a program, was very much determined by certain physical constraints like the number of units (mostly accumulators and stepper counters) or program cables available. Because of these issues, it was decided that the machine should be permanently rewired once it was moved from the Moore school to the BRL, so that it would become possible to ‘code’ the program by using symbolic instructions rather than to ‘wire’ it. In that set-up the machine became a kind of ‘stored-program’ machine applying concepts from von Neumann’s First Draft of a Report on the ED-VAC (Electronic Discrete Variable Automatic Computer), viz. the design which is considered by many to be the blueprint of the modern computer (Haigh et al. 2016).⁶

So why was this machine so special when compared to other calculatory devices that were being used at the time? The ENIAC has four properties which are considered to be fundamental and where each feature very much depends on the previous one:

1. It was a discrete machine
2. It was capable of so-called conditional branching which is a key feature for the construction of a general-purpose machine, in addition to the four basic arithmetic operations
3. Possibility of coding the machine—this is only true of the rewired ENIAC.
4. It had electronic high speed.

While properties 1–3 could already be found in other machines, most notably Aiken’s Mark 1 and the Bell Model V machine, it is the combination of 1–3 with 4 which was new and so, from this perspective, the most fundamental innovation of the ENIAC was its speed. In fact, it can be argued that it is the addition of this last feature to the other three which requires, permits and gives rise to a rethinking of computational methods including those for ‘controlling’, viz. programming.

Three People—Three Practices

This section considers the work and viewpoints of three people who were involved with the ENIAC: Derrick H. Lehmer, Haskell B. Curry and John von Neumann. Curry and Lehmer were selected here as they were two of the three members of the so-called Computations Committee that was assembled by the U.S. Army in order to test, among other things, the newly built ENIAC. Moreover, they provide a small counterbalance to the “legend” of John von Neumann who is the most well-known mathematician involved with the ENIAC.⁷ The parallel discussion of aspects of each of their work and viewpoints permits me to show the diversity of methods that were developed around this machine, thus contextualizing the Monte Carlo ‘simulations.’

Derrick H. Lehmer—Number Theory

Background and Involvement with the ENIAC

Derrick H. Lehmer was first and foremost a number theorist. He was very much influenced by his father, also a number theorist, and was convinced “that mathematics, and especially number theory, is an experimental science” (Lehmer 1969: 3). Lehmer was very aware of the possibilities of mechanical aids to this kind of number theory as becomes clear from his involvement in the journal *Mathematical Tables and other Aids to Computation* and his construction of several ‘prime sieves’ in the 1920s and 1930s, including a photoelectronic sieve and one with bicycle chains.

During World War II, Lehmer became involved with the war effort by contributing to the research carried out by the Applied Mathematics Panel (AMP). The AMP was established at the end of 1942, when the National Defense Research Committee (founded in 1940) was reorganized from five to nineteen divisions. It was directed by Warren Weaver. Its purpose was “to bring mathematicians as a group more effectively into the work being carried on by scientists in support of the nation’s war effort” (Bush et al. 1946b: vii). The AMP had contracts with different universities who worked on specific classes of problems. The University of California where Lehmer was working at the time, was also involved.

One set of problems that was studied by the California group was related to “pattern bombing”, which concerned “the almost simultaneous release of all the bombs carried by a formation of aircraft, thus giving rise to a pattern of bombs affected, as a unit, by an aiming error” (Bush et al. 1946b: 46) and it was with this problem that Lehmer became involved (Lehmer 1945) describing a photo-electric instrument for mechanizing a specific method “to estimate the probability of at least one hit or, alternatively, the expected proportion of hits, in formation attacks on irregular target areas.” Interest-

ingly, and as will be argued in more detail elsewhere,⁸ this method, which is called a “model experiment” (Bush et al. 1946b: 56), can now be understood as a variant of the Monte Carlo method even before the Monte Carlo method had been developed and used in the ENIAC context and is very close to a notion of simulation as a means to solve problems in a non-analytical manner. More particularly, a model experiment is “used as a means to solve certain bombing problems which would proceed tediously if approached by numerical integration” (Bush et al. 1946b: 56). An example of such a problem is to determine the number of attacks “needed to give a probability of at least P that at least the proportion F of the target would be covered at least n times” (Bush et al. 1946b: 48). The method for tackling this problem was a model experiment “in which a series of synthetic random-bombing operations were performed, with enough replications to permit the estimation of probability levels from order statistics. The data[...] was then used as the basis for an empirical function” (Bush et al. 1946b: 48). Interestingly, a notion of ‘simulation’ is used in this context:

Most of the work in the second and third studies was done by experimental statistical methods in which model experiments simulating the conditions of the problem were repeated a number of times. The theory [...] can be formulated analytically in terms of appropriate mathematical formulas but the computation that would have been involved in the mathematical approach would have been prohibitive. [...] There are undoubtedly many other statistical problems of this type in military research which can be more effectively handled for practical purposes by experimental methods than by analytical methods. (Bush et al. 1946b: 79)⁹

Moreover, it seems that this notion of ‘simulation’ refers directly to the random events used in the model experiments as is clear here:

A second statistical study [...] was carried out by means of a miniature random number experiment, in which the radius of clearance of a single rocket and the errors involved in delivering the 120 rockets in a barrage were simulated. (Bush et al. 1946b: 79)

Fig. 1 gives a schematic representation of Lehmer’s instrument called the photoelectric analyzer.

Its basic principle was to (repetitively) project a synthetic bomb pattern onto a ground glass screen after passing through a diaphragm-stop cut out in the form of the target (the so-called “target screen” in Fig. 1) so that the ground glass screen would be illuminated only by that part of the bomb

Fig. 1 Graphical representation of Lehmer's photo-electronic bombing analyser. (From Bush et al. 1946b)

pattern which intersected the target. The light from the screen was focused on a photoelectric cell which

was instrumented so as (1) to add the effect of successive images of bomb patterns, or (2) to count the cases which were not blank. A movie projector and a film with 1,500 frames were used. Each frame carried a picture of the bomb pattern with its center displayed to represent a random deviate from a Gaussian distribution. (Bush et al. 1946b: 57)

Apparently, after the war, plans were made to construct a number of these instruments at Wright Field but I could not verify to what extent these devices were actually used and for which applications.

In 1945 then a computations committee was assembled at the BRL in order to prepare for "utilizing the [ENIAC] machine after its completion" (Alt 1972: 693). Lehmer was asked to join that committee.

Developing Mathematics on the ENIAC: A Number-theory Problem

So what did Lehmer do to test the ENIAC? Most importantly from that perspective is the number-theory computation that he prepared and set-up over a labor day weekend in 1946 with Emma Lehmer, a mathematician and number theorist who was also his wife.¹⁰ This work concerned the computation of exponents e of $2 \bmod p$, viz. the smallest value of e such that $2^e \equiv 1 \bmod p$. It was a known fact that Fermat's little theorem can be used as a primality test. If for a given number b , $2^b \equiv 2 \bmod b$ then b is with high probability a prime number. Unfortunately, an infinite set of exceptions to this primality test exists. A table of exponents then can be used to compute such exceptions and the Lehmers' ENIAC computation was used to correct and extend the existing tables of exponents. Given

the military context in which the ENIAC was built and used, this problem could be considered as quite obscure or irrelevant and in a certain sense it was not a “standard” problem if one can use that word in the context of this machine. However, as a test problem it was quite important for two reasons. First of all, and as was pointed out by Alt: “It was the most stringent performance test applied up to that time, and would be an impressive one even today” (Alt 1972: 694).

Secondly, given that it was the first number-theory computation that was ever run on an electronic machine, it was important in that it showed that machines like the ENIAC might also be useful for scientific purposes that were unrelated to the war effort and so it could be used as an example to convince scientists of the usefulness of high-speed computation for their work:

I think what’s particularly interesting about the number theory problem they ran was that this was a difficult enough problem that it attracted the attention of some mathematicians who could say, yes, an electronic computer could actually do an interesting problem in number theory. (Alt 2006: 40)

Clearly, while the problem itself already had a tradition within number theory, the introduction of a high-speed and parallel machine affected the methods for tackling it. First of all, and in connection to Lehmer’s previous work, the machine allowed the implementation of a truly parallel prime sieve. Secondly, one of the main subroutines set-up on the machine and which was called the ‘exponent routine’ was quite different from the human methods that would normally be used. Indeed, and as Lehmer explains, the ENIAC “was instructed to take an ‘idiot’ approach” (Lehmer 1974: 5). To start, the machine needed a table of prime numbers. Of course, in a context where one does not have high-speed, the obvious thing to do is to provide that table to the machine during the computation by means of punched cards. However, since this is a mechanical process, it would significantly slow down the computational process. Hence, it was decided not to use an existing prime table but to let the machine compute its own next value of p as it was needed by using the aforementioned prime sieve. The next step was to calculate the powers of 2 reduced modulo p , with p prime. The “idiot approach” taken by the ENIAC resulted in a routine which required “only one addition, subtraction, and discrimination at a time cost, practically independent of p , of about 2 seconds per prime. This is less time than it takes to copy down the value of p and in those days this was sensational” (Lehmer 1974: 5). Thus, with the introduction of high-speed computation, it was realized that the usual slow, human methods needed to be replaced

by other methods which might be less ingenious and require more brute force but were also more efficient.

It was also this approach, which was used in the computation of another problem that was suggested by Lehmer to George Reitwiesner, who worked at the Aberdeen Proving Ground, and which became known as "Slow Moses". This was the computation of the so-called Fermat quotient. The program was special because it was probably the first instance of "the interrupted idle time modus operandi" (Lehmer 1974: 5). Apparently, Reitwiesner used it to prove to Aberdeen Proving Ground engineers that the ENIAC was also able to run for longer periods of time by running the problem every night for a certain period of time when it would otherwise stand idle. Interestingly, in this case, Lehmer blames the "idiot approach" for the slowness of the program (Lehmer 1974: 5).

Another interesting aspect of the exponent computation is that it required not just the full machine but often resulted in local engineering tricks not unlike what we would today call 'hacking'. As recounted by Jean Bartik, one of the six female operators of the original ENIAC: "Lehmer's little problems, they were always too big for it. So consequently, you always had to be changing it or to think of something new and innovative in order to get a problem or ways that you could break the problem down into smaller portions" (Bartik 1973). Thus, one could say that in the case of the exponent problem, the machine 'implements' or 'is' the method and so both the method and machine are reciprocally structured and shaped.¹¹ It is not completely clear what other work Lehmer did in the ENIAC context, though it is quite certain that he assisted in the preparation and actual set-up of a number of other problems that were ran on the ENIAC as is clear, for instance, from the acknowledgments in (Hartree 1946) and (Grubbs 1950). Thus, at least in Lehmer's case, there is no 'radical difference' between practices of number theory; practices of machine-building and problems of applied mathematics. In fact, it was the need for mechanization in both number theory and applied mathematics which made it quite straightforward to make the switch from one to the other practice *drawing on using* Lehmer's experience with machine-building. This, of course, fits into a longer historical tradition where mechanization, computation and (applied) mathematics go hand-in-hand.

A Number-theorist Engaging with the ENIAC—Reflections

But while Lehmer's use of the ENIAC is, from a certain perspective, a continuation of an existing tradition, it is also the culmination of that tradition: Lehmer (like several others) was greatly impressed by the potential of high-speed digital computation for his own field and after his experience with work on the ENIAC machine he would write and talk on several occasions

about the usefulness and potential of high-speed electronic computers in science and, more specifically, mathematics.

Lehmer clearly had specific views on number theory and mathematics at large. Indeed, as explained above, he had a view on number theory as an experimental science which he contrasted with the more traditional and widely-held viewpoint, or, in Lehmer's words "school of thought" (Lehmer 1977: 745).¹² In Lehmer's understanding, 'experiment' cannot simply be equated to 'experiments' as in, say, physics.¹³ For one thing, the 'reality' of a physics experiment is quite different from the 'reality' of the mathematical experiment, especially in number theory and so the idea that the computer has come to "stand [...] for nature itself" (Galison 2011: 157) or constitutes "an alternative reality" can and should be understood differently in the case of number theory. Indeed, here the computer, as a digital device which counts, has come to "stand" for the "universe" of numbers.

In Lehmer's view then, the computer is just another tool which requires, first of all, a rethinking of existing methods and, secondly, provides access to a new range of problems which he calls "discrete-variable problems". Lehmer contrasted the discrete-variable or digital machines with analog machines (Lehmer 1956: 481). The main focus of that paper is to show (engineers) what kind of problems both types of machines can be used for and how methods need to be changed when switching from analog to digital devices. Indeed, as Lehmer points out:

From the point of view of the discrete-variable device, things need to be counted rather than measured; mathematics is not geometry but arithmetic; the universe is quantized and this includes mathematics. Integrals are but sums, and derivatives are but difference quotients; functions are discontinuous everywhere; limits, infinities and infinitesimals do not really exist [...]. Thus [...] we seem to go back to Pythagoras. [...]. The methodological step-by-step reiteration [...] is to be contrasted with the *modus operandi* of the analogue machine. (Lehmer 1956: 485–486)

In other words, if the computer constitutes an alternate reality it is a discrete reality and not a continuous one, which is a problem one needs to deal with as a physicist but not as a number theorist.¹⁴

Interestingly and in the same paper, Lehmer does use the word 'simulation' but it is not used with reference to digital machines but with reference to analog devices. This is not surprising: the very idea behind calling the 'continuous' machines analog machines was exactly because they were understood as 'analogous' to what they were supposed to 'simulate' namely, continuous phenomena. In fact, the current *Oxford English Dictionary* (OED) definition of 'simulation', viz.: "The technique of imitating

the behaviour of some situation or process (whether economic, military, mechanical, etc.) by means of a suitably *analogous* [m.i.] situation or apparatus, esp. for the purpose of study or personnel training" (OED online 2019). seems closer in spirit to a notion of analog machines than to a notion of a digital machine as used by many in the late 1940s and early 1950s. From the physics and engineering perspective, the transition from continuous to discrete-variable machines, at least at the time, required in fact a reduced level of analogy between the model or system being "simulated" as compared to so-called analog or continuous machines which were more directly linked to the continuous world of physics.

Curry—Logic and the Automation of Firing Table Computations

Background and Involvement with the ENIAC

Haskell B. Curry is today mostly known as one of the founders of combinatorial logic and mathematical logic in the U.S.A. Less well known is his contribution to the war effort during World War II, when he worked on ballistics rather than logic problems for the US government. It was this involvement which resulted in him becoming one of the members of the *Computations Committee* for testing the ENIAC machine since computing ballistic problems was one of the main occupations of the ENIAC. Without engaging too much with the details of Curry's biography, it is worth pointing out that he was clearly already a patriot as a student when he decided to become a member of the Student Army Training Corps in October 1918, when the U.S. had entered World War I. Also, his decision to study mathematics was motivated by its potential relevance for artillery. Curry was thus part of a small but growing community of people in the U.S. who realized the potentials of mathematics for the military (Archibald et al. 2014). For more details on Curry's biography, see (Seldin 2005).

In the late 1920s, Curry became more interested in logic. He was fascinated by the substitution operation for propositional logic as described in *Principia Mathematica* by Russell and Whitehead which he considered to be too complicated. His approach then was to analyze the substitution operation down to its simplest possible elements. The result of this was a set of operators which he called combinators. This approach, to reduce something to its most elementary form, would become a characteristic of part of Curry's way of working and, as we will see, would be one of the main methods in his theory of program composition that he developed in the wake of his work with the ENIAC.

However, having switched to logic did not stop Curry from investing in the military. And so when the World War II started he became a member of the joint War Preparedness Committee of the *American Mathematical Society* and the *Mathematical Association of America*. In a paper entitled

“Mathematical Teaching and National Defense,” which resulted from the committee’s deliberations on “all aspects of the relation between mathematics and defense” (Curry 1942: 337), it becomes clear how strong Curry’s convictions actually were, stating that “modern war is largely mathematical in character” (Curry 1942: 337).

When the U.S. entered the war, Curry left university life to work and serve as a mathematician at the Frankford Arsenal. This was one of six U.S. Army ammunition facilities. Each of which worked on one or more aspects of ordinance research and development. The facility in Frankford made “studies on the manufacture of metal components for shells and bombs, and designs and develops methods for production of small arms ammunition” (Hoff 1943: 18).

Once a specific weapon was designed and a prototype produced, it was sent to the Aberdeen Proving Ground where it was tested. It is not completely clear what Curry did during his time at Frankford. According to (Seldin 2012) he worked mostly “on the mathematics of aiming a projectile at a moving target, the so-called fire control problem”, clearly, a ballistic problem. He also published two papers while at the Frankford Arsenal (Curry 1943; Curry 1944). Neither of these contain new methods but are instead descriptions of existing mathematical techniques “with emphasis on its practical aspects” (Curry 1944: 259) in order to show their potential for the kind of problems Curry was confronted with at Frankford. In May 1945, Curry moved to the BRL at the Aberdeen Proving Ground where he stayed until September 1946 and made it to Chief of the Theory section of the Computing Laboratory and then (Acting) Chief of the Computing Laboratory. It was during this time that he became involved with the ENIAC as a member of the computations committee.

Developing Mathematics on the ENIAC: Interpolation Problems

While at Aberdeen, Curry worked mostly on problems related to computing firing tables with the ENIAC. I found four problems Curry worked on while at Aberdeen that relate directly to the ENIAC:

1. a numerical method for “smoothing” drag functions
2. inverse interpolation problems
3. fourth order interpolation
4. the computation of the first 2000 digits of e

I will focus here only on (1) and (2). For a short discussion of (4), see (Seldin 2012). A description of the work on (3) can be found in (Lotkin & Curry 1946).

Ad (1) A Numerical Method for “Smoothing” Drag Functions: Curry’s work on the first problem is in fact a modification of the spline interpolation method which was introduced by Isaac Jacob Schoenberg in the ENIAC context. Schoenberg was a Romanian mathematician, who had moved to the U.S. in the early 1930s, and moreover held a position at the University of Pennsylvania by the time the World War II started. In August 1943 he also joined the BRL at the Aberdeen Proving Grounds. Leo Zippin, another mathematician who had joined the war effort, arranged for Schoenberg to work at the BRL for the duration of the war where he was given a very specific task:

Trajectories of projectiles were until then computed with desk calculators by hand. Into these computations entered tables of the drag-functions of air resistance [...] In performing these computations on the ENIAC, which was very fast, a much simpler integration method of very small step could be used. In these methods, the accumulation of the round-off errors was unacceptable due to the rough drag-function tables; they needed to be smoothed by being approximated by analytic functions. (Schoenberg 1988)

In other words, in 1943 when it had only just been decided that the army would finance the construction of a high-speed digital computer, Schoenberg was asked to develop a numerical method which would be more suited to the particularities of that machine, viz. a method that would exploit the discreteness and high-speed of the machine. The method which was introduced by Schoenberg is known as (an instance of) ‘spline interpolation.’ The basic idea is to approximate a function by piecewise polynomial functions, viz. instead of having one polynomial, the approximation is done by polynomial pieces of a certain degree n which join at certain points known as knots. Schoenberg’s original method used equidistant knots and it was then Curry “who recognized the possibility of defining splines with arbitrary non-equidistant knots” (Schoenberg 1988: 5), and this work was published as (Curry & Schoenberg 1947; Curry & Schoenberg 1966).

Ad (2) Inverse Interpolation Problems:¹⁵ The second problem from the ENIAC context on which Curry worked is the problem of inverse interpolation:

This problem is important in the calculation of firing tables. Suppose the trajectory calculations have given us the coordinates (x, y) of the projectile as functions of t (time) and φ (angle of departure). For the tables we want t and φ as functions of x and y ; indeed we wish to determine φ so as to hit a target whose position (x, y) is known, and t

is needed for the fuze setting or other purposes. (Curry & Wyatt 1946)

In other words, the problem of inverse interpolation concerns the computation of the initial settings of an artillery device such as the fuze settings or the angle of departure. As was the case with the smoothing problem, here again the numerical methods would have to be significantly adapted or shaped by the limitations and possibilities of the ENIAC. To start with, the method used is based on iteration “which is eminently suitable for ENIAC” (Curry & Wyatt 1946: 6). But the choice for an iterative method is also anchored in the possibility of the ENIAC to reuse certain programs “independent of the choice of the [interpolation] formula for $f(u)$ ” (Curry & Wyatt 1946: 6) where $f(u)$ is the interpolatory approximation of the function $x(t)$ mentioned above. Indeed, the report by Curry and Wyatt is intended to provide a general framework for problems of inverse interpolation and so: “A basic scheme of programming is set up in detail in such a way that it can be readily modified to suit circumstances” (Curry & Wyatt 1946: 6).

In other words, the methods of tackling the problem of inverse interpolation are very much determined not just by the high-speed of the computation but its combination with the ENIAC’s ‘programmability’. In fact, we can go a step further and say that while the report by Curry and Wyatt clearly focuses on the inverse interpolation problem, this is studied not for its own sake but “with reference to the programming on the ENIAC as a problem in its own right” and thus it was also a good test problem to reflect on issues of programming the ENIAC.

As explained above, for the original ENIAC machine, setting up a problem was quite laborious and hence many of the choices and reflections made in the Curry-Wyatt report are motivated by the need to simplify the programming process. One clear example of that is the choice for a numerical method which is not the most efficient in terms of convergence—which was less of a problem given the high-speed of the machine—but results in a simpler program set-up, or, to put it in the words of Curry and Wyatt: “For the ENIAC [...] extremely rapid convergence is not necessary. [...] A far more important consideration than speed of convergence is simplicity of programming” (Curry & Wyatt 1946: 14). Perhaps more interesting from the contemporary perspective is the development of a more systematic and structured approach to programming in the report, providing a hierarchical structure to programs which differentiates between:¹⁶

- a) program elements,
- b) program sequences or stages,
- c) processes,
- d) program.

The most central elements here are the 'stages' or 'program sequences' of a process: "The stages can be programmed as independent units, with a uniform notation as to program lines, and then put together" (Curry & Wyatt 1946). Indeed, it is the structurization of processes into stages which allows for the modifications needed to carry out different types of inverse interpolation simply by reordering or reusing the stages as well as to making it possible to use a particular stage at different places in a program. The latter method is today known as a 'closed subroutine' and is usually considered to be a fundamental feature only of stored-program machines.

As is clear, even though Curry was only involved with the ENIAC and the BRL for a short time, he was quite able to combine the needs and limitations of the ENIAC machine with his experience in applied mathematics and his viewpoint as a logician who sought to simplify complicated operations by analyzing them into simpler elements. This bringing together of these three aspects was driven not so much by a desire to solve one specific calculatory problem but instead to develop more general numerical and programming methods that suited the machine to provide a higher degree of efficiency.

Curry would pursue this path by developing a so-called "theory of program composition," which aimed to automate the process of subroutining, and so can be and was understood (at least by some) as an anticipation of the work that was done subsequently on compilers and higher-level programming in the late 1950s.

A Logician Engaging with the ENIAC—Reflections

What is logic and, more specifically, formal logic about? In his first published paper on logic, Curry writes, "the essential purpose of mathematical logic is the construction of an abstract (or strictly formalized) theory, such that when its fundamental notions are properly interpreted, there ensues an analysis of those universal principles in accordance with which valid thinking goes on (Curry 1929: 363). In other words, formal logic is the bringing together of what Curry considers to be the main purpose of logic, viz. the analysis and criticism of thought" (Curry 1928: 363) with the formal methods of mathematics. Or, put differently, it is about modeling (valid) human thought through formalization. Moreover, Curry's specific approach was to perform such analysis through simplification, viz.:

The rules [of any abstract theory] form the port of entry of intelligence; and since nothing can be done without them, they represent the atoms of thought [...] into which the reasoning can be decomposed. It follows that in constructing such a theory [...] it is [...] important to so chose the rules that they involve [...] only the simplest actions of the human mind. (Curry 1929: 367–368)

It is exactly this approach of simplification which Curry then applies to the “highly complex” rule of substitution resulting in his theory of combinators. It serves the purpose of understanding “processes by means of which entities may be combined to get new entities” and so we see that Curry’s ‘model’ for substitution is intended to emulate (rather than simulate) a dynamical process.¹⁷ The model however has the potential of being dynamical if we start to effectively ‘apply’ the combinatory rules to ‘generate’ a system of assertions.

This is confirmed by Curry’s use of his work on combinators in his attack on the problem of program composition and which is a continuation of the work done on the ENIAC but for the Institute for Advanced Study (IAS) machine which, basically, followed von Neumann’s EDVAC design (De Mol et al. 2015). The basic idea is that of automating or, at least, mechanizing, that part of the “coding” process which concerns the tying together and combining of several smaller programs into one and so, among others, allow for the automation of access to and return from (closed) subroutines and the automation of loops. In short, the aim is to develop, what we would today call a compiler for programs and to automate part of the programming. Indeed, as was later explained by George W. Patterson of the Burroughs Corporation in a review of a short paper by Curry: “automatic programming is anticipated by the author” (Patterson 1957: 103).¹⁸

Curry’s attack on the problem also relied on his earlier work on combinators (De Mol et al. 2015). By reconnecting Curry’s earlier work on combinatory logic and his work on program composition, it can be seen how his earlier work on substitution can effectively be reinterpreted as a dynamical model once it is applied effectively to program code which, by its very nature, is intended to structure a (computational) process. That is, the potentially dynamical nature of Curry’s earlier model becomes real when it is executed on a machine. Moreover, since it is the work of the human operator which is being automated, the emulative (rather than the more general simulative) aspect of Curry’s work is not about physical processes but about executing a model of human work.

Curry’s work on the calculation of ballistic trajectories gives yet another reading of ‘simulation’ in the early computing context which is quite related to Lehmer’s earlier war work on bombing patterns. It shows that the

fundamental change in the use of electronic computing machinery does not so much lie in the development and elaboration of just one specific scientific method (the Monte Carlo method) as in some kind of 'tertium quid' (Galison 2011: 137) alongside more theoretical and experimental approaches, but instead in the need to *rethink* existing calculatory methods, which were already quite commonly used in a practice that aimed to develop ways to 'approximate' physical realities to test and develop artillery weapons. From that historical perspective, the idea of the Monte Carlo method "elevated [...] above the lowly status of a mere numerical calculation" (Galison 2011: 119) becomes quite problematic. It was just part of a broader historical development in science and technology whereby, for various reasons, calculation is relied on rather than analytical methods or experimentation.¹⁹

Von Neumann—Formalism and Mathematical Physics

Background and Involvement with the ENIAC

Unlike Curry and Lehmer, von Neumann is much celebrated as a computer pioneer and mathematician who has contributed to (the foundations of) a great variety of fields and subfields including computing, operator theory, economic theory, set theory and quantum theory. By consequence, von Neumann has become a kind of hero for those involved in the disciplines to which he contributed and care is needed when discussing his scientific biography.

It seems fair to say that von Neumann was first and foremost a Göttingen-minded mathematician and greatly influenced by Hilbert, among others, via Erhard Schmidt, (Hashagen 2009). Hilbert is of course well-known for his formalist program which, roughly speaking, aimed to provide a consistent foundation for mathematics by relying on finitary and formalist methods. Von Neumann, apparently, had a special talent for the "art" of formalization and so it is not surprising that Hilbert very much appreciated von Neumann's work in this context. However, after he heard Gödel's talk in 1930 at Königsberg that presented the now (in)famous incompleteness theorem, and which is often considered as a fundamental blow to Hilbert's program,²⁰ von Neumann turned his back on the Hilbertian ideal of mathematical logic. As was the case with Curry, it would be this more formal work that would prove very useful for his work in computing. It was around the same time that von Neumann was offered a position by Veblen at Princeton and so he emigrated to the United States where he would stay for the rest of his (relatively short) life. But having turned his back on mathematical logic, Germany and Hungary certainly did not mean turning his back on a Göttingen tradition of establishing bridges between pure and applied mathematics and, especially, physics (Rowe 2018). By the

time von Neumann moved to the United States in the 1930s he was already well equipped to become part of the small but growing community of U.S. mathematicians like Curry and Lehmer, who would later use their talents in the war effort in the Applied Mathematics Panel (see also Aspray 1990: 34). However, his work in the 1930s was still mostly within “pure” mathematics and oriented towards providing foundations for physics.

It was most probably under the initiative of Veblen (Aspray 1990: 26) that von Neumann was asked, in 1937, to work as a consultant at the BRL at Aberdeen where he would become a regular visitor from that time onward. As he recounted later, it was by Robert Kent, a senior BRL official, that he was introduced to “military science, and it was through military science that I was introduced to applied sciences. Before this, I was [...] essentially a pure mathematician. [...] I have certainly succeeded in losing my purity” (Aspray 1990: 26). At that time, von Neumann shifted his attention to problems of shock wave theory and fluid dynamics and realized that the best approach to these problems might be the use of brute-force computational methods—methods which were already extensively applied at the BRL at the time.

In the 1940s he also became a consultant and later a member of Division 8-Explosives at the National Defense Research Committee (NRDC) where he worked on problems of detonation. It is perhaps no coincidence that Richard Courant (1888–1972), who had emigrated to the U.S. after he had fled from Germany in 1933 because he was Jewish, was also working on these problems. He also had been a consultant for Division 8 and then became the representative at the AMP for the OEMsr-944, “Investigation in shock wave theory”, and OEMsr-945 “Research in problems of the dynamics of compressible gases, hydrodynamics, thermodynamics, acoustics, and related problems” contracts (Bush et al. 1946a). Von Neumann on the other hand was the technical representative for the AMP OEMsr-1111 contract with the Princeton Institute for Advanced Studies which had as its topic: “Studies of the potentialities of general-purpose computing equipment, and research in shock wave theory, with emphasis upon the use of machine computation.” In other words, he became the Army’s specialist for the use of computing machinery and its potential for studying problems in shock wave theory. As explained above, von Neumann had become convinced (or, better, had been convinced) that such problems could not be handled by the usual analytical methods and so he proposed the use of brute-force computation instead. Indeed, in one of the AMP summary reports entitled “Mathematical studies relating to military physical research,” a numerical method proposed by von Neumann (von Neumann 1944) for the handling of problems of shock waves (Galison 2011) is described as follows:

The hypotheses of isentropy and that of all shocks being “straight” are generally not fulfilled. When they are abandoned, however, any exact mathematical analysis becomes quite interactable [sic] [...]. Consequently, considerable importance attaches to a computational treatment [...] which ignores shocks but which appears to produce arbitrarily good approximations to a rigorous theory allowing for shocks. The treatment depends on a much simplified quasimolecular model in place of the continuous theory. (Bush et al. 1946a: 38)

Thus, in the early 1940s, von Neumann was already quite familiar not just with the idea of using brute-force to attack certain problems of (applied) physics but also with the idea of developing another model so that a numerical approach could be used. Moreover, and as was the case for Curry and Lehmer, this kind of work is part of a broader and organized push by the U.S. military and some leading mathematicians to bridge the gap between applied and pure math.

It was also because his “interest in explosives was genuine” (Kistiakowsky 1980: 62) that von Neumann became a Los Alamos consultant, and thus he had become a highly-placed scientist with access to top-secret projects. In this role he also paid regular visits to Aberdeen, and legend has it that it was during one of these visits in 1944 that he accidentally met Herman H. Goldstine at Aberdeen Railway station where Goldstine informed him about the secret ENIAC project. Goldstine arranged the clearance documents and very soon von Neumann became involved with the ENIAC project (with Goldstine as his collaborator).

Developing Mathematics on the ENIAC: A and H Bomb Problems

The story about von Neumann’s work on and with the ENIAC has been told a number of times, hence I keep this section short.

Von Neumann’s most well-known work on the ENIAC, in collaboration with several others, concerns:

1. the rewiring of the ENIAC
2. the ‘first’ Monte Carlo computations on the ENIAC

ENIAC Rewiring: The conversion of the ENIAC into an EDVAC-like machine was very much driven by von Neumann and the ideas he elaborated in his 1945 EDVAC report, though he was certainly not the only one to have contributed to it. The main idea behind this conversion was to ‘program’ the machine by instructions rather than by wiring which meant that the various instructions that could be used were wired once and for all inside the machine and so could be referred to by a code. There were

at least two reasons for this rewiring of the ENIAC into a kind of stored-program, serial machine. First of all, ‘coding’ a problem on the ENIAC required less time than changing and reconnecting the units and re-setting the switches. Secondly, in its original modus, the problems that could be set-up on the machine were limited by the number of available units (mostly accumulators and stepper counters) and so there was a serious restriction on the “size” of the problems that could be run on it. By switching to coded instructions, there was much less restriction on the length of a program. It was especially the size problem that had to be resolved to permit the planned Monte Carlo computations which were prepared concurrently with the rewiring and largely motivated it (Haigh et al. 2016). In fact, it seems fair to conclude on the basis of the careful analyses given in (Neukom 2006; Haigh et al. 2016) that the Monte Carlo method and the ENIAC conversion were co-developed.

The ‘first’ Monte Carlo Computations on the ENIAC: These are the well-known computations which introduced the Monte Carlo method for the first time in the context of studying the behavior of neutron chain reactions for fission devices. Here is von Neumann’s often quoted description of the general idea behind the computation, in a letter to Robert D. Richtmyer dated March 11, 1947 and which also includes a more detailed “computation sheet” for the computation:

Consider a spherical core of fissionable material surrounded by a shell of tamper material. Assume some initial distribution of neutrons in space and in velocity but ignore radiative and hydrodynamic effects. The idea is to now follow the development of a large number of individual neutron chains as a consequence of scattering, absorption, fission and escape. [...] [A] genealogical history of an individual neutron is developed. The process is repeated for other neutrons until a statistically valid picture is generated. [...] How are the various decisions made? To start with, the computer must have a source of uniformly distributed pseudo-random numbers. (quoted from: Metropolis 1987: 127)

Whereas the use of random decisions for studying (models of) physical processes in a computational or other setting, was certainly not new (see the Lehmer example above) it was only with the high-speed, converted ENIAC that such methods could be more fully explored, especially for larger problems which were computation-intensive and required a relatively complex and lengthy program. The method for handling the logical complexity on the programming level was the use of a (large) flowdiagram as developed in (Goldstine & von Neumann 1947–1948).²¹ Thus, we

see that the computational challenges posed by the Monte Carlo computation were practically handled by the (converted) ENIAC and more systematically by von Neumann's report on the EDVAC and the sequence of reports, written with Goldstine, on flowdiagrams (Goldstine & von Neumann 1947–1948). Thus, it can be concluded that the Monte Carlo method in its first application in the ENIAC context, not only resulted from but also very much drove the development of the new technology.

The rewiring of the ENIAC and the planning and programming of the Monte Carlo computations were certainly not the only achievements to which von Neumann contributed in the ENIAC context. One interesting contribution is related to the need for (pseudo-)random numbers for the Monte Carlo computation. Given the speed of the computations, providing random numbers externally would slow down the process and so von Neumann came up with the idea of having the machine compute its own random numbers and developed a numerical method for computing (pseudo)random numbers. This is the middle square method. Von Neumann's interest in pseudo-random numbers most probably led to his interest in computing the first 2000 digits of π and e .²²

As recounted by Reitwiesner "early in June, 1949, Professor John von Neumann expressed an interest in the possibility that the ENIAC might sometime be employed to determine the value of π and e to many decimal places with a view toward obtaining a statistical measure of the randomness of distribution of the digits" (Reitwiesner 1950: 11).

Indeed, the purpose of those computations was to know more of the probability distribution of the digits of both numbers. Thus, the ENIAC not only made it possible to carry out the Monte Carlo study on a scale that was not possible before but it also resulted in an interest in a new class of problems, viz. that of pseudo-random numbers.

Besides, I briefly mention von Neumann's contributions to numerical analysis and, more specifically, the study of error propagation, which is closely related to von Neumann's interest in pseudo-random numbers.²³ This is elaborated in the paper (Goldstine & von Neumann 1947–1948) which provides a "rigorous discussion" of the problem of deriving "rigorous estimates in connection with the inversion of matrices of higher order" (von Neumann & Goldstine 1947: 1022). Like Schoenberg's splines method, it is very much rooted in the problem of round-off errors: given the discrete nature of machines like the ENIAC, one can "achieve any desired precision" (von Neumann 1958: 25). However, because of the high-speed of the operations on numbers, errors occurring in each operation are superposed. Hence, it is important to have an estimate of the precision needed in order to avoid such round-off errors.

A Göttingen Mathematician Engaging with the ENIAC—Reflections

The shift in von Neumann's work around 1940 from developing pure mathematics for theoretical physics to the use of brute-force calculations might seem curious at first. Indeed, even today “[p]roceeding by ‘brute force’ is considered by some to be more lowbrow” (Ulam & von Neumann 1980: 94). However, von Neumann came from a tradition where the building of bridges between the pure and applied was very much promoted and was himself an active supporter of this viewpoint:

I think that it is a relatively good approximation to truth [...] that mathematical ideas originate in empirics [...] But there is a grave danger that the subject [...] so far from its source, will separate into a multitude of insignificant branches, and that the discipline will become a disorganized mass of details and complexities [m.i.] [...]. [W]henver this stage is reached, the only remedy seems to me to be the rejuvenating return to the source: the re-injection of more or less directly empirical ideas. (von Neumann 1947)

So, when von Neumann was working at the BRL and saw the potential of calculation for problems of applied mathematics at work, the move from the pure to the applied side is perhaps less surprising. The computer then, both as a tool for studying problems from mathematical physics and as a device which demands its own mathematical theory, was the ideal in-between for this goal of re-injecting empirical ideas into mathematics. And an in-between it certainly was to von Neumann. In one of his lectures in the series entitled “Theory and Organization of Complicated Automata” delivered at the University of Illinois in December 1949, he frames computing machines as tools in-between mathematical and experimental methods for studying certain problems

there are large areas in pure mathematics where we are blocked by a peculiar inter-relation of rigor and intuitive insight, each of which is needed for the other, and where the unmathematical process of experimentation with physical problems has produced almost the only progress which has been made. Computing, which is not too mathematical either in the traditional sense but is still closer to the central area of mathematics than this sort of experimentation is, might be a more flexible and more adequate tool in these areas than experimentation. (von Neumann 1966: 33–35)

Thus, for von Neumann, the computer was the ideal tool for studying those problems for which the more traditional methods of mathematics did not allow an intuitive understanding of the problem set. The computer, which was still more mathematical than pure experimentation, was the

next best thing available. We see here von Neumann's version of Galison's "tertium quid" (Galison 2011: 137): the computer as a tool which connects the two traditions of experimental and mathematical physics. However, as was also the case with Curry and Lehmer, this 'tertium quid' was very much prepared by a tradition in which disciplinary boundaries were already being crossed.

Von Neumann's reflections on computing machines *as a user* went hand-in-hand with reflections from the more theoretical side of the bridge between the pure and applied: there is of course his later work on automata, which provide a basic model used in current simulation contexts viz. cellular automata, but before that he also described a more theoretical model for a computing machine known as the EDVAC model and which is very much rooted in some of the issues with the original ENIAC. As has been discussed elsewhere (Aspray 1990; Haigh et al. 2016), that model abstracts from engineering details and instead uses the formal model of neuron nets developed by McCulloch and Pitts (McCulloch & Pitts 1943). The latter paper also abstracts away from "the physiological and chemical complexities of what a neuron really is" (von Neumann 1966: 43) and instead uses a framework of formal logic. Among others, it refers to Turing's work on abstract computing machines, viz. formal models for defining the notion of (human) computability and which *retroactively* have become an important model for the modern computer.²⁴ The McCulloch and Pitts paper thus had an important effect on von Neumann's work in computing not just by the application of its methods and ideas in the EDVAC context, but also as a basic reference in his reflections on natural and artificial automata.²⁵ This is not surprising: the idea of connecting the field he had turned his back on, viz. mathematical logic, with 'empirical' processes, whether engineered or not, must have been very appealing to him in the light of his later viewpoints on mathematics: it offered him yet another opportunity to re-inject the empirical back into that part of mathematics which had been so devoid of any empirical content and, perhaps because of that, had failed (at least in von Neumann's view).²⁶

The possibility of a new role for formal logic within the field of computing was further elaborated by von Neumann in his work on programming (together with Hermann H. Goldstine) and developed in the three volume reports (Goldstine & von Neumann 1947–1948).²⁷ These are often considered as one of the first historical sources on programming;²⁸ the flowdiagram idea they developed, was for a long time a much used method within programming (Ensmenger 2016). In fact, these flowdiagrams were very much part of the practice developing around the Monte Carlo program and calculations and were used to develop them for the (rewired) ENIAC machine. And it was in using the flowdiagram notation in actual practice

that certain alterations were made to it (Haigh et al. 2016). In other words, as with our previous cases, here also we see that the actual methods are part of a complex practice which involves use of the machine, mathematics and the development of programming methods and notations.

According to von Neumann, the need for a more logical approach to programming, which he called the planning and coding of problems,²⁹ is rooted in the high speed of the computer

[C]ontemplate the prospect of locking twenty people for two years during which they would be steadily performing computations. And you must give them such explicit instructions at the time of incarceration that at the end of two years you could return and obtain the correct result for your lengthy problem! This dramatizes the necessity for high planning, foresight, and consideration of the logical nature of computation. *This integration of logic in the problem is a consequence of the high speed.* (von Neumann 1948: 2, my emphasis)

More specifically, logic can be used to capture the so-called ‘dynamical’ aspect, the flow, of a computation—that is, the structuring of how orders have to be executed. It is introduced mostly because the use of loops, sub-routines and conditionals in high-speed computing mean that the actual “flow” of a program is not reflected by the mere static sequence of coded instructions, viz. the control C does not “scan the coded sequence of instructions linearly. It may jump occasionally forward or backward omitting [...] some parts of the sequence, and going repeatedly through others. It may modify some parts of the sequence while obeying the instructions in another part of the sequence” (Goldstine & von Neumann 1947–1948: 1). In other words, the behavior of the control C is dynamical and so requires dynamical notation.

This dynamical stage, which is the first stage of the coding of problems, was quite strictly separated from the so-called static or microscopic stage of coding, viz. the individual coding of every single operation indicated in the flowdiagram. Indeed, the flowdiagram (Fig. 2) is the notation to be used for the dynamical stage whereas the static stage (Fig. 3) is the explicit sequence of coded instructions resulting from the flowdiagram.

This separation between the static and the dynamical stage is quite strict and motivated by what could be called a division of labor. Indeed, the dynamical stage requires a “mathematician or [...] [a] moderately mathematically trained person and the static stage someone with a moderate amount of experience” (Goldstine & von Neumann 1947– 1948: 20).

This is at least partially rooted in the fact that von Neumann’s flowdiagram is a model of the dynamical behavior of the control C whereas in

Fig. 2 Flowdiagram of the first problem discussed in von Neumann & Goldstine (1947) which requires a flowdiagram

111,1	C		Ac	$(m + 2i - 2)_0$
2	111,3	Sp	111,3	$(m + 2i - 2)_0$
3	—		Ac	u_i
[	$m + 2i - 2$		s.1	u_i
4	s.1	S	Ac	$(m + 2i - 2)_0$
5	C		Ac	$(m + 2i - 1)_0$
6	B.8	h	111,22	$m + 2i - 1$ S
7	111,22	Sp	Ac	$(m + 2i)_0$
8	B.8	h	C	$(m + 2i)_0$
9	C	S	R	d
10	B.4	R	Ac	du_i
11	s.1	x	Ac	$du_i + e$
12	B.5	h	s.2	$du_i + e$
13	s.2	S	R	a
14	B.1	R	Ac	au_i
15	s.1	x	Ac	$au_i + b$
16	B.2	h	s.3	$au_i + b$
17	s.3	S	R	$au_i + b$
18	s.3	R	Ac	$au_i^2 + bu_i$
19	s.1	x	Ac	$au_i^2 + bu_i + c$
20	B.3	h	R	$v_i = \frac{au_i^2 + bu_i + c}{du_i + e}$
21	s.2	÷		

Fig. 3 Part of the static coding for the problem from Fig. 2. (von Neumann & Goldstine 1947)

Curry's case it is much more about automating the coder. To quote Lehmer, who apparently was not aware of Curry's work:

Much has been said, but little written, about the logic or even the topology of programming. Logicians and topologists are not coming to the rescue of the desperate programmer. [...] This is the combinatorial complexity to which I have referred. Flow diagrams showing the routines, subroutines, and other wheels within wheels are hardly distinguishable from the block diagrams of the machine itself; the latter, however, are made once and for all. This then is the white mans burden of large-scale computing. (Lehmer 1951)

The result is an approach which cannot be automated: the flowdiagram which models the dynamics of the control cannot result in an 'emulation' of the programmer/coder.

Discussion "A Pretence of What is Not"

What, if anything, is the impact of high-speed computing on science? This is the driving question behind current discussions on computer simulation. The aim of this paper was *not* to contribute *directly* to this discussion but to provide instead *a broader and deepened historical perspective* by revisiting the roots of computer simulation, viz. the context from which von Neumann and Ulam developed their Monte Carlo methods, and studying part of the wider historical context in which they arose.

As is shown here the Monte Carlo method and computations belong to a broader institutional, scientific and technological tradition (or set of traditions). Thus, the apparent "chaotic assemblage of disciplines" which "have no single history that can be narrated smoothly across time" but which shared a practice that was "sufficiently congruent in the years just after World War II" (Galison 2011: 119) becomes less chaotic in the light of increased investments in and organization of military science and the involvement of mathematicians. This goes back, for the U.S. at least, to the end of the World War I, thus when the U.S. entered World War II, several mathematicians like Lehmer and Curry brought their mathematical knowledge to the war effort. Furthermore, there was von Neumann (and with him several other émigré mathematicians) who came from the Göttingen tradition which promoted a view of a close connection between pure and applied mathematics. Finally, there were also the existing calculational practices used in making tables for scientific or other purposes and which involved mathematicians, engineers and human computers work-

ing together, as well as the increased use of calculatory instruments. It is against this background that one can understand that when the AMP was created in late 1942, there was a “complete lack of astonishment which greeted [the] contributions [of the AMP members]” (Bush et al. 1946a: v). Or, put differently, the idea of “radically different activities” which could now be coordinated locally in the trading zone that was supposedly constituted by the Monte Carlo method and calculations (Galison 2011: 119), needs perhaps to be seen in the inverse: it was the increasing level of co-operation and the crossing of disciplinary boundaries together with the increasing level of mechanization and organization of computation which made possible and required the development of new methods and technologies. These new methods and technologies in their turn, increased the need for this kind of crossing of (perhaps retro-actively so-construed) disciplinary boundaries. They did not, however, make it possible.

But the historical perspective on the Monte Carlo method not only needs to be broadened by connecting it to what came before, and thus as the result and continuation of earlier work, but also in relation to what was happening around the same time locally. First of all, the Monte Carlo computations *were clearly not* the first or only instances in which it was decided to rely on calculation rather than on ‘pure’ mathematical methods and/or ‘real’ experiments at that time. Perhaps the most explicit example of this discussed in this paper was the use of so-called model experiments which are very close to the idea behind the Monte Carlo computations.³⁰ Moreover, and especially if we consider the ENIAC context, it is understood that the main concern was the development of efficient mathematical methods and programming techniques which were adapted to the particularities of the machine (its high-speed and general programmability) in order to tackle both older and newer problems. One example is the Curry inverse interpolation problem which is framed in a problem context where the aim was to ‘approximate’ physical realities to test and develop artillery weapons. Put differently, it is the machine and the problems to be attacked with it that required fundamental rethinking of the three interrelated practices around the machine, viz.: the mathematics used and developed, see for instance the use of an “idiot” approach; new approximation methods or new methods for generating “random” numbers;

- Programming, see for instance Curry’s theory of program composition or von Neumann and Goldstine’s flowdiagram approach;
- Machine design, see for instance the ENIAC conversion and the EDVAC design.

These structured practices are given context through the particular viewpoints, backgrounds and purposes of the different “users” involved.

Thus, what mattered first of all was not questions of ‘simulation’ but questions of good computational methods for specific problems independent of whether or not they created some kind of *tertium quid*. From this perspective, the Monte Carlo technique is just one of the many methods and so is not more ‘elevated’ than the others (see endnote 21).

Moreover, other work that was done in the wake of the ENIAC context needs to be re-interpreted as *simulative work* in the context of programming (and thus consider the intertwining of mathematical logic and engineering) and the context of number theory (and thus the computer as an “analog” machine for doing number theory) rather than with the usual understanding of ‘simulation’ from the 1940s and 1950s as mimicking some physical process by relying on randomized methods.

Thus, instead of focusing on (philosophical issues of) simulation and on what the relation is between the simulator and what it mimics, viz. simulation *as* simulation, the approach here was to render it historically transparent by showing how it was and is structured by a set of different practices. It is only from such a perspective that one can really appreciate the complexity and constituting role not just of the Monte Carlo method but of a broader range of other problems and methods such as, for instance, Curry and Wyatt’s work on inverse interpolation, for the history of (computational) science and technology.

Moreover, I have shown that accounts of (computer) simulation which assume a discontinuous history that has one main starting point (the rise of the modern computer and the ‘first’ Monte Carlo computations) give a reductive and singular account which might distort our current views on simulation. As such, a focus on ‘simulation’ as simulation might result in a two-fold masking. First, by focusing on simulation and not on computational methods one masks a plurality and diversity of practices and how they are determined not just by what they are supposed to ‘simulate’ but also by the techniques, programming interfaces and, ultimately (historically) the machine. Secondly, because of the additional connotation of ‘simulation’ as a ‘mimicking’ this masking is strengthened. Viz, it suggests that it suffices to look at what is being simulated and not at what constitutes the simulator. As such, simulation really does become “a pretence of what is not,” connecting it back to its older meanings.³¹ This is not only a problem for philosophers and historians of science but a very real one today when science relies more and more on computation and where scientists alongside mathematicians are confronted more and more with issues of transparency, error, open and closed software, etc. (see e.g. Ince et al. 2012). Uncovering some of the historical complexities of ‘simulation,’ retracing its so-called origins, is one strategy in addressing such issues.

Acknowledgements

This paper is to some extent determined by some of the ideas elaborated in the ANR project PROGRAMme (ANR-17-CE38-0003-01). I am indebted to Maarten Bullynck, Martin Carlé, Juan Duran, Mark Priestley and Franck Varenne for discussing aspects of this paper. Special thanks go to Sibylle Anderl, Arianna Borrelli, Nathalie Bredella, Rudi Seising and Janina Wellmann for their extensive comments on an earlier version of this paper. Part of this work is based on a talk given at the conference "The Plurality of Numerical Methods in Computer Simulations and their Philosophical Analysis," 3–4 November 2011, entitled "ENIAC, matrix of numerical simulation(s)". The talk was prepared by Maarten Bullynck, Martin Carlé and myself, and dedicated to the memory of Friedrich Kittler.

Endnotes

- 1 This is to some extent explained by the fact that the main community of historians of computing identifies itself as historians of technology rather than of science, and hence not so much attention has been given to actual scientific practices in that community. See (Hashagen 2013) for an overview and (De Mol & Bullynck 2019) for a critical reflection on the relation between historians of computing and historians of science.
- 2 A basic exception here is the work by Franck Varenne (Varenne 2007; Varenne 2018), which is anchored in detailed historical case-studies that focus on the use of simulation in the context of plant biology.
- 3 This statement, "Es gibt keine (Computer) Simulation" is very much inspired by Friedrich Kittler's "Es gibt keine Software" (Kittler 1993). The phrase should be understood accordingly, viz. that simulation is contingent on what lies historically and epistemologically at its basis.
- 4 See in this respect (Ramunni 1989) where it is shown that it was only much later, when the digital high-speed computer started to successfully compete with analog machines in industry that the term became more frequently used with its current meaning (i.e. referring to digital computer-aided computation).
- 5 Of course, one can say the same of machine computations but scaled by several factors.
- 6 Note that the converted ENIAC is, from a certain perspective, just another set-up of the original machine.
- 7 It should be noted here that the focus on these three mathematicians is not intended to give a full representation or overview of all the ENIAC-related practices nor is it implied that there were no other basic uses. The aim is rather to show how different practices result in different notions, understandings and methods for doing computer-assisted research. A detailed study of the problems of (Fritz 1994) would of course be very interesting but that would require another ENIAC book or a series of papers.
- 8 I am preparing a paper together with Maarten Bullynck with the provisional title: Monte Carlo on the beach: model experiments and throwdown studies before the ENIAC.
- 9 It is important to recall here that usually, the real novelty of von Neumann and Ulam's methods is that they applied sampling techniques to non-probabilistic problems (see also Borrelli 2019). This case clearly shows that this cannot be true since it was realized several years before that problems which are described analytically can nonetheless be tackled with sampling techniques.

- 10 A detailed study of that work can be found in (Bullyncx & De Mol 2010).
- 11 This is less the case today where one no longer has access to the machine in such direct manner though one could argue that it is now the “program” and so, by extension, the software “implementing” the method which is reciprocally shaped and structured by its applications and uses.
- 12 Lehmer was not alone in having this kind of viewpoint. Ulam, for instance, also had a more empirical view on mathematics and the same can be said of the later von Neumann.
- 13 For a more detailed discussion of Lehmer’s understanding of experiment, see for instance (De Mol 2015).
- 14 Of course, Lehmer was not the only one nor the first to describe the difference between so-called ‘analog’ and digital machines in terms of measuring vs. counting devices. See e.g. (Stibitz 1945; von Neumann 1958: 3 and 6)
- 15 See (De Mol et al. 2015) for a detailed study of Curry’s work on the inverse interpolation problem and how it affected his later work on the composition of programs.
- 16 It seems that this structurization as result of Curry’s work. Indeed, in his notes on the e calculation, which predate the Curry-Wyatt report, he already identifies this kind of hierarchical structure.
- 17 I am indebted to Franck Varenne for having noticed that the original phrasing in terms of capturing (rather than emulating) was less appropriate.
- 18 Note however that Curry’s work as such most probably did not have an impact on the later work, among other reasons, because he lost interest in the problem due to various factors.
- 19 To be clear, I am not making claims here about the significance of the Monte Carlo method as such, but instead argue that this does not necessarily imply it was considered more elevated than other methods studied and developed at the time.
- 20 It proves, very roughly speaking, that no finite axiomatic basis can completely capture mathematics.
- 21 In (Haigh et al. 2016) the different flowdiagrams for the different stages of the Monte Carlo computation are given and discussed in more detail and so we do not repeat that material here. It should be noted that in their interpretation these flowdiagrams illustrates a new way of programming which they call the modern code paradigm and which, according to them, shows a basic difference between programs from the original ENIAC and the rewired ENIAC.
- 22 Though we should not forget here that also Curry had considered the problem of computing digits of e on the early ENIAC.
- 23 In fact it is quite probable that von Neumann’s so-called middle-square method originated in his research on round-off errors. See in this context (von Neumann 1951).
- 24 Note though that the Turing machine model did not serve as a model for the first computers.
- 25 See for instance the second lecture at Illinois University entitled ‘Rigorous Theories of Information and Control’ which deals mostly with the Turing model and the McCulloch-Pitts model.
- 26 Indeed in “The Mathematician” von Neumann’s main argument in favor of reconnecting the pure with the empirical is connected with Gödel’s incompleteness results which showed him that one should be cautious “against taking the immovable rigor of mathematics too much for granted. This happened in our own lifetime, and I know myself how humiliatingly easily my own views regarding the absolute mathematical truth changed during this episode” (von Neumann 1947).
- 27 The following paragraphs are based on two talks I gave in 2014 and 2016. The slides from 2016 (in French) are available here: <https://hal.archives-ouvertes.fr/cel-01345599>.
- 28 But see (De Mol et al. 2015) for a critical discussion.
- 29 It is quite interesting to point out that von Neumann did not really use the programming terminology which, as is known, had its roots in the ENIAC.

- 30 Though, of course, they were not used to study thermonuclear problems nor were the computations done with the help of a high-speed device like the ENIAC.
- 31 See the Oxford English Dictionary and its reference to meanings of simulation prior to 1947 (OED online [2019](#)).

References

- Alt, Franz (1972). Archaeology of Computers. Reminiscences, 1945–1947. *Communications of the Association for Computing Machinery* (15): 693–694.
- Alt, Franz (2006). Dr. Franz Alt. Oral history by Atsushi Akeru, January 23 and February 2, 2006. In: *Proceeding Association for Computing Machinery Oral History interviews*, interview nr. 1, New York: Association for Computing Machinery.
- Archibald, Thomas, Della Dumbaugh and Deborah Kent (2014). U.S. mathematicians in World War I. In: David Aubin and Catherine Goldstein (eds.). *The War of Guns and Mathematics. Mathematical Practices and Communities in France and its Western Allies around World War I*. Providence: American Mathematical Society: 229–271.
- Aspray, William (1990). *John von Neumann and the Origins of Modern Computing*. Boston: MIT Press.
- Bartik, Jean J. (1973). Interview with Jean J. Bartik (1924–) and Frances E. (Betty) Snyder Holberton (1917–2001) by Henri S. Tropp. *Computer Oral History Collection*, 1969–1973, 1977, Archives Center, National Museum of American History.
- Borrelli, Arianna (2019). Program FAKE: Monte Carlo Simulations as Tools of Theory in Early High Energy Physics. *NTM* this special issue. <https://doi.org/10.1007/s00048-019-00223-w>
- Bullyncx, Maarten and Liesbeth De Mol (2010). Setting-up Early Computer Programs: D. H. Lehmer's ENIAC Computation. *Archive for Mathematical Logic* (49): 123–146.
- Bullyncx, Maarten and De Mol, Liesbeth (2019). The History (of the History) of Computing and its Relation to the History (of the History) of Mathematics. In: Maarten Bullyncx, Liesbeth De Mol and Marie-José Durand-Richard (eds.). *Mathematics and Machines*, special issue of *Revue de Synthèse*, 2019.
- Bush, Vannevar, James B. Conant and Warren Weaver (1946a). *Mathematical Studies Relating to Military Physical Research*, National Defense Research Committee, Summary Technical Report of the Applied Mathematics Panel, NDRC, vol. 1, AD221604.
- Bush, Vannevar, James B. Conant and Warren Weaver (1946b). *Probability and Statistical Studies in Warfare Analysis*, National Defense Research Committee, Summary Technical Report of the Applied Mathematics Panel, NDRC, vol. 3.
- Campbell-Kelly, Martin, Mary Croarken, Raymond Flood and Eleanor Robson (2003). *The History of Mathematical Tables. From Sumer to Spreadsheets*. Oxford: Oxford University Press.
- Curry, Haskell B. (1929) An Analysis of Logical Substitution, *American Journal of Mathematics* (51): 363–384.
- Curry, Haskell B. (1942). Mathematical Teaching and National Defense, *The School Review* (50): 337–346.
- Curry, Haskell B. (1943). The Heaviside Operational Calculus. *The American Mathematical Monthly* (50): 365–379.
- Curry, Haskell B. (1944). The Method of Steepest Descent for Non-linear Problems, *Quarterly of Applied Mathematics* (2): 258–261.
- Curry, Haskell B. and Isaac J. Schoenberg (1947). On Spline Distributions and Their Limits: The Polya Distribution Functions. *Bulletin of the American Mathematical Society* (53): 1114.
- Curry, Haskell B. and Isaac J. Schoenberg (1966). On Polya Frequency Functions IV: The Fundamental Spline Functions and Their Limits. *Journal d'Analyse Mathématique* (17): 71–107.

- Curry, Haskell B. and Willa A. Wyatt (1946). *A Study of Inverse Interpolation of the ENIAC*. Technical Report 615, Ballistic Research Laboratories, Aberdeen Proving Ground.
- De Mol, Liesbeth (2015). Some Reflections on Mathematics and Its Relation to Computer Science, in: Andy Adamatzky (ed.). *Automata, Universality, Computation. Tribute to Maurice Margenstern*. Berlin: Springer: 75–101.
- De Mol, Liesbeth, Martin Carlé and Maarten Bullynck (2015). Haskell Before Haskell: an Alternative Lesson in Practical Logics of the ENIAC. *Journal of Logic and Computation* (25): 1011–1046.
- De Mol, Liesbeth and Marie-José Durand-Richard (forthcoming). Calculating Machines and Numerical Tables—a Reciprocal History. In: Dominique Tournès (ed.). *The History of Numerical Tables*. Berlin: Springer.
- Duran, Juan (2018). *Computer Simulations in Science and Engineering. Concepts—Practices—Perspectives*. Dordrecht: Springer.
- Ensmenger, Nathan (2016). The Multiple Meanings of a Flowchart. *Information and Culture: A Journal of History* (51): 321–351.
- Forman, Paul (2007). The Primacy of Science in Modernity, of Technology in Postmodernity, and of Ideology in the History of Technology. *History and Technology* (23): 1–152.
- Fritz, Barkley (1994). ENIAC—a Problem Solver. *IEEE Annals of the History of Computing* (16): 25–45.
- Galison, Peter (1997). *Image and Logic. A Material Culture of Microphysics*. Chicago: University of Chicago Press.
- Galison, Peter (2011) [1996]. Computer Simulations and the Trading Zone. In: Gabriele Gramelsberger (ed.). *From Science to Computational Science*. Zurich: Diaphanes: 118–157.
- Goldstine, Hermann H. and John von Neumann (1947–1948). Planning and Coding of Problems for an Electronic Computing Instrument, Vol. 2 of *Report on the Mathematical and Logical Aspects of an Electronic Computing Instrument*, part I, II and III. Report prepared for U. S. Army Ord. Dept. under Contract W-36-034-ORD-7481, 1947–1948.
- Grier, David A. (2005). *When Computers Were Human*. Princeton: Princeton University Press.
- Grubbs, F. E. (1950). Sample Criteria for Testing Outlying Observations. *The Annals of Mathematical Statistics* (21): 27–58.
- Haigh, Thomas, Mark Priestley and Crispin Rope (2016). *ENIAC in Action. Making and Remaking the Modern Computer*. Boston: MIT Press.
- Hartmann, Stephan (1996). The World as a Process. In: Rainer Hegselmann, Ulrich Mueller, and Klaus G. Troitzsch (eds.). *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*. Dordrecht: Springer: 77–100.
- Hartree, David R. (1946). The ENIAC, An Electronic Computing Machine. *Nature* (158): 500–506.
- Hashagen, Ulf (2009). Die Habilitation von John von Neumann an der Friedrich-Wilhelms-Universität in Berlin: Urteile über einen ungarisch-jüdischen Mathematiker in Deutschland im Jahr 1927. *Historia Mathematica* (37): 242–280.
- Hashagen, Ulf (2013). The Computation of Nature, or: Does the Computer Drive Science and Technology? In: P. Bonizzoni P., V. Brattka V. and B. Löwe (eds.). *The Nature of Computation. Logic, Algorithms, Applications. CiE 2013*. Berlin: Springer: 263–270.
- Hoff, Frederick C. (1943). Ordnance—the Backbone of Warfare. *Alumni Review. California Institute of Technology* (6): 18–19.
- Humphreys, Paul W. (2004). *Extending Ourselves: Computational Science, Empiricism, and Scientific Method*. Oxford: Oxford University Press.
- Ince, Darrel C., Leslie Hatton and John Graham-Cumming (2012). The Case for Open Computer Programs, *Nature* (482): 485–488.
- Kistiakowsky, George B. (1980). Reminiscences of Wartime Los Alamos. In: L. Badash, J. O. Hirschfelder and H. P. Broida. *Reminiscences of Los Alamos 1943–1945*. Dordrecht: Reidel Publishing Company.
- Kittler, Friedrich A. (1993). Es gibt keine Software. In: Friedrich Kittler (ed.). *Draculas Vermächtnis. Technische Schriften*. Berlin: Reclam: 225–242.

- Lehmer, Derrick H. (1945). *A Photo-Electric Instrument for Rapid Bomb-Fall Analysis and Damage Prediction*, D. H. Lehmer, OSRD 5641, OEMsr-817, AMP Report 190.1R, University of California, May 1945. AMP-804.4-M.
- Lehmer, Derrick H. (1951). Mathematical Methods in Large Scale Computing Units, *Proceedings of Second Symposium on Large-Scale Digital Calculating Machinery*, 1949, Cambridge, MA: Harvard University Press: 141–146.
- Lehmer, Derrick H. (1956). High-speed Computing Devices and Their Applications. In: Edwin F. Beckenbach (ed.). *Modern Mathematics for the Engineer*. New York: McGraw-Hill: 480–497.
- Lehmer, Derrick H. (1969). Computer Technology Applied to the Theory of Numbers. *Studies in Number Theory* (6): 117–151.
- Lehmer, Derrick H. (1974). The Influence of Computing on Mathematical Research and Education. In: Joseph Pierre LaSalle (ed.). *The Influence of Computing on Mathematical Research and Education*. Providence: American Mathematical Society: 3–12.
- Lehmer, Derrick H. (1977). Mechanized Mathematics. *Bulletin of the American Mathematical Society* (72): 739–750.
- Lotkin, Max and Haskell B. Curry. (1946). A Study of Fourth Order Interpolation on the ENIAC, BRL report 613.
- McCulloch, Warren and Walter Pitts (1943). A Logical Calculus for the Ideas Immanent in Nervous Activity, *Bulletin of Mathematical Biophysics* (5): 115–133.
- McLuhan, Marshall and Quentin Fiore (1967). *The Medium is the Massage: An Inventory of Effects*. New York: Bantam.
- Metropolis, Nicholas (1987). The Beginning of the Monte Carlo Method. *Los Alamos Science* (15) Special Issue: Stanislaw Ulam 1909–1984: 125–130.
- Neukom, Hans (2006). The Second Life of ENIAC. *IEEE Annals of the History of Computing* (28): 4–16.
- von Neumann, John (1944). Proposal and Analysis of a New Numerical Method for the Treatment of Hydrodynamical Shock Problems. AMP Report 108.1R, Institute for Advanced Study, Mar. 20, 1944.
- von Neumann, John (1947). The Mathematician. In: Heywood, R.B. (ed.). *The Works of the Mind*. Chicago: University of Chicago Press: 180–196.
- von Neumann, John (1948). Electronic Methods of Computation. *Bulletin of the American Academy of Arts and Sciences* (1): 2–4.
- von Neumann, John (1951). Various Techniques Used in Connection with Random Digits. Summary written by G. E. Forsythen. *National Bureau of Standards Applied Mathematics Series* (12): 36–38.
- von Neumann, John (1958). *The Computer and the Brain*, New Haven: Yale University Press.
- von Neumann, John (1966). *The General and Logical Theory of Automata*, edited by Arthur Burks, Urbana: University of Illinois Press.
- von Neumann, John and Goldstine Hermann H. (1947). Numerical Inverting of Matrices of High Order. *Bulletin of the American Mathematical Society* (53): 1021–1099.
- OED online (2019). *Simulation*. Oxford: Oxford University Press. <https://www-loed-1com-10066f0ya024d.erf.sbb.spk-berlin.de/view/Entry/180009?redirectedFrom=simulation&> (24.08.2019).
- Patterson, G.W. (1957). Review of “The Logic of Program Composition by H.B. Curry”. *Journal for Symbolic Logic* (22): 102–103.
- Polachek, Harry (1997). Before the ENIAC. *IEEE Annals for the History of Computing* (19): 25–30.
- Ramunni, Girolamo (1989). *La Physique du Calcul. Histoire de l'Ordinateur*. Paris: Hachette.
- Reitwiesner, George W. (1950). An ENIAC Determination of π and e to more than 2000 Decimal Places. *Mathematical Tables and Other Aids to Computation* (4): 11–15.
- Rowe, David (2018). *A Richer Picture of Mathematics. The Göttingen Tradition and Beyond*. Berlin: Springer.
- Schoenberg, Isaac J. (1988). A Brief Account of My Life and Work. In: C. De Boor (ed.). *I.J. Schoenberg. Selected Paper*. Basel: Birkhäuser, vol. 1: 1–10.
- Seldin, Jonathan P. (2005). *Haskell B. Curry (1900–1982)*. Dictionary of Modern American Philosophers: Bristol Thoemmes Press.

- Seldin, Jonathan P. (2012). Curry's Work on Computers in the Early Days of Computing. *Canadian Society for History and Philosophy of Mathematics. Proceedings of the 38th Annual Meeting* (25): 149–153.
- Stibitz, George R. (1945). *A Talk on Relay Computers*. AMP Memo 171.1M, National Defense Research Committee.
- Swade, Doron (2003). The 'Unerring Certainty of Mechanical Agency': Machines and Table Making in the Nineteenth Century. In: M. Campbell-Kelly, M. Croarken, R. Flood and E. Robson (eds.). *The History of Mathematical Tables. From Sumer to Spreadsheets*. Oxford: Oxford University Press: 145–174.
- Tedre, Matti (2015). *The Science of Computing: Shaping a Discipline*. London: Routledge.
- Ulam, Stanislaw and John von Neumann (1980). The Interaction of Mathematics and Computing, 1980. In: Jack Howlett, Nicholas Metropolis and Gian-Carlo Rota (eds.). *A History of Computing in the Twentieth Century*. New York: Academia Press: 93–99.
- Varenne, Franck (2007). *Du Modèle à la Simulation Informatique*. Vrin: Paris.
- Varenne, Franck (2018). *From Models to Simulations*. London: Routledge.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Liesbeth De Mol
 Centre National de la Recherche Scientifique (CNRS), Unité Mixte de
 Recherche (UMR) 8163 Savoirs, Textes, Langage
 Université de Lille
 France
 liesbeth.de-mol@univ-lille.fr