Los Alamos Bets on ENIAC: Nuclear Monte Carlo Simulations, 1947–1948

Thomas Haigh
University of Wisconsin–Milwaukee

Mark Priestley
Crispin Rope

From rich archival sources, the authors reconstruct the evolution of a program first run on ENIAC in April 1948 by a team including John and Klara von Neumann and Nick Metropolis. This was not only the first computerized Monte Carlo simulation, but also the first code written in the modern paradigm, usually associated with the "stored program concept," ever executed.

By reconstructing the Monte Carlo calculations carried out on ENIAC in 1948 on behalf of Los Alamos Scientific Laboratory, this article examines programming practice and scientific computation at the dawn of modern computing. This is the final article in a three-part series published in *Annals* exploring modifications made to ENIAC that made it the first computer to support what we have elsewhere defined as the "modern code paradigm," one of a cluster of related innovations propagated by John von Neumann in his 1945 "First Draft of a Report on the EDVAC."[1] (The first article in this series, "Reconsidering the Stored Program Concept," examined the history of the "stored program concept" and proposed a set of more specific alternatives.[2] The second, "Engineering 'The Miracle of the ENIAC': Implementing the Modern Code Paradigm," explored the conversion of ENIAC to the new programming method and put its capabilities into context against other computers of the late 1940s.[3]) The term "modern code paradigm" describes the control mechanism adopted by modern computers, including the automatic execution of programs stored in an addressable memory and expressed as a series of operation codes followed by arguments. Programs written in this new style relied on conditional or calculated jumps to change the course of their execution and modified the addresses acted on by instructions to iterate through data structures.[2]

The ENIAC Monte Carlo program run in April and May 1948 was both the first computerized Monte Carlo simulation and the first program written in the new paradigm to be executed on any computer. The evolution we document here from computing plan through a series of flow diagrams and planning documents to a revision of the code after initial tests provides a window through which we can observe the first full revolution of what would later be thought of as the software system development lifecycle. Although scholarly historians of computing have good reason to be leery of the hunt for "firsts" that dominated our field in its infancy, this does give the code an undeniable historical interest.

More than a decade ago, historians of computing identified software history as a vital and under-researched area and have since gone a long way toward filling this gap.[4] After an early focus on programming language design, more recent investigations have focused on the history of particular software companies and their founders, economic analysis of different sectors of the software industry, the software engineering movement and its relationship to the identity of programming, and the development of packaged software genres such as spreadsheets, word processors, and database management systems.[5]

Studies of what programmers and other kinds of system developers actually do, of

 Published by the IEEE Computer Society 1058-6180/14/$31.00

software as a technological artifact, or of the coevolution of hardware and software have remained conspicuous by their absence. The importance of these topics has been recognized in related fields, including emerging communities focused on software studies, critical code studies, or platform studies. Within broader historical communities, historians of technology have placed an increasing emphasis on the importance of understanding technology in use, exploring the social meanings and technical cultures in which technologies are enveloped.[6]

Historians of science have placed a corresponding emphasis on studies of scientific practice, examining what scientists actually do inside and outside the laboratory. The study of instrumentation, technologies used to observe and measure aspects of nature, has been a particularly vibrant field. This article engages directly with the computations discussed in Peter Galison's classic "Computer Simulations and the Trading Zone." That chapter depicted early computer simulation as the product of a new heterogeneous community with "a new cluster of skills in common, a new mode of producing scientific knowledge" constituted by "common activity centered around the computer."[7] Different kinds of expertise were "traded" around this common object. Galison suggested that ENIAC's Monte Carlo calculations "ushered physics into a place paradoxically dislocated from the traditional reality that borrowed from both experimental and theoretical domains" by building "an artificial world in which 'experiments' (their term) could take place" within computers.[8] The status of simulation as a new kind of scientific experimentation has since been a major concern for philosophers of science and for historians of computing, such as Michael S. Mahoney and Ulf Hashagen.[9]

Galison's chapter is revered more for its concepts and argument than for its detailed analysis of the specifics of early nuclear simulations. He first writes in some detail on John von Neumann's 1944 work on numerical methods for the treatment of the hydrodynamic shocks produced within exploding nuclear weapons. Although a greatly simplified model of the ignition of a fusion weapon provided ENIAC with its first actual problem, run in late 1945 and very early 1946, this was not a Monte Carlo simulation. Turning next to Monte Carlo, Galison uses archival correspondence to explore the techniques used to produce pseudorandom numbers and sketches in von Neumann's published 1947 plan for the simulation of neutron diffusion in a fission reaction. However, Galison does not follow, or even mention, the main topic of this article: the development of this initial sketch into an evolving set of ENIAC programs used for at least four distinct batches of Monte Carlo fission simulations during 1948 and 1949. Instead, his narrative jumps from von Neumann's 1947 enthusiasm for fission Monte Carlo to 1949 plans to simulate an entirely separate physical system, Edward Teller's design for a "Super" fusion bomb.[10] Anne Fitzpatrick filled some of these gaps from the Los Alamos perspective, but our current article provides the first detailed examination of the 1948 Monte Carlo simulations.[11]

Monte Carlo methods proved to be of great importance to scientific computing and operations research after their computerized debut on ENIAC. The original code's direct descendants, run on computers at Los Alamos and Livermore laboratories, were a vital aspect of weapons design. They drove the needs of two of the world's most important purchasers of high-performance computer systems and so, according to Donald MacKenzie, exerted a direct influence on the development of supercomputer architecture.[12] Monte Carlo methods were one of the most important and widely adopted techniques in the transformation of scientific practices around computer simulation.

Beyond their established importance to the history of science, the Monte Carlo programs run on the ENIAC in 1948 are also of considerable importance to the history of software. We believe them to be the best documented application programs run on any computer during the 1940s, allowing us to assemble a detailed reconstruction of the programs as run. We located several original flow diagrams including the final version for the spring 1948 calculations, the second major version of the program code in its entirety, and a detailed document describing changes made between the first and second versions of the program. We also consulted the ENIAC operations log book, which documents each day of machine activity during the period and the process by which ENIAC was converted into a machine able to run code written in the modern paradigm.

The calculations also shed light on an underexplored aspect of the work of John von Neumann and his Princeton-based collaborators, who were then the most influential group of computing researchers in the

United States and had been intimately involved in creating and disseminating crucial ideas on the design and programming of electronic computers such as the von Neumann architecture, the modern code paradigm, subroutine libraries, and flow diagrams.[13] Their work on ENIAC's new instruction set and the Monte Carlo code took place as they were moving from the design of the Institute for Advanced Studies computer, which provided the template for most of the electronic computers constructed in the United States in the early 1950s, to its construction, issuing in the process an influential series of reports on programming and diagramming methods.[14] The material we have uncovered captures changes in the team's thinking about the structure of the computation as they absorbed the implications of the modern code paradigm, in particular the flexibility its control structures of branches and loops offered in comparison with earlier control methods embodying fixed ideas about computational structures.

Monte Carlo Origins

There is no single Monte Carlo method. Rather, the term describes a broad approach encompassing many specific techniques. As its name lightheartedly suggests, the defining element is the application of the laws of chance. Physicists had traditionally sought to create elegant equations to describe the outcome of processes involving the interactions of huge numbers of particles. For example, Einstein's equations for Brownian motion could be used to describe the expected diffusion of a gas cloud over time, without needing to simulate the random progression of its individual molecules. There remained many situations in which tractable equations predicting the behavior of the overall system were elusive even though the factors influencing the progress of an individual particle over time could be described with tolerable accuracy.

One of these situations, of great interest to Los Alamos, was the progress of free neutrons hurtling through a nuclear weapon as it began to explode. As Stanislaw Ulam, a mathematician who joined Los Alamos during the war and later helped to invent the hydrogen bomb, would subsequently note, "Most of the physics at Los Alamos could be reduced to the study of assemblies of particles interacting with each other, hitting each other, scattering, sometimes giving rise to new particles."[15]

Given the speed, direction, and position of a neutron and some physical constants, physicists could fairly easily compute the probability that it would, during the next tiny fraction of a second, crash into the nucleus of an unstable atom with sufficient force to break it up and release more neutrons in a process known as fission. One could also estimate the likelihood that neutron would fly out of the weapon entirely, change direction after a collision, or get stuck. But even in the very short time span of a nuclear explosion, these simple actions could be combined in an almost infinite number of sequences, defying even the brilliant physicists and mathematicians gathered at Los Alamos to simplify the proliferating chains of probabilities sufficiently to reach a traditional analytical solution.

The arrival of electronic computers offered an alternative: simulate the progress over time of a series of virtual neutrons representing members of the population released by the bomb's neutron initiator when a conventional explosive compressed its core to form a critical mass and trigger its detonation. Following these neutrons through thousands of random events would settle the question statistically, yielding a set of neutron histories that closely approximated the actual distribution implied by the parameters chosen. If the number of fissions increased over time, then a self-sustaining chain reaction was underway. The chain reaction would end after an instant as the core blew itself to pieces, so the rapid proliferation of free neutrons, measured by a parameter the weapon designers called "alpha," was crucial to the bomb's effectiveness in converting enriched uranium into destructive power.[16]

The weapon used on Hiroshima is estimated to have fissioned only about 1 percent of its 141 pounds of highly enriched uranium, leaving bomb designers with a great deal of scope for refinement. Using Monte Carlo, the explosive yield of various hypothetical weapon designs could be estimated without using up America's precious stockpiles of weapons-grade uranium and plutonium. This was, in essence, an experimental method within a simulated and much simplified reality.

The origins of the Monte Carlo approach have been explored in a number of histories and memoirs, so we need not attempt an exhaustive account here. Ulam later recalled developing the basic idea with John von Neumann during a long car ride from Los

Alamos in 1946.[17] Over the next few years both men, along with several of their Los Alamos colleagues, would actively promote the new approach within the scientific community. For example, von Neumann was already discussing its possible use in his 13 August contribution to the famous Moore School Lectures of 1946.[18]

Early Planning for Los Alamos Monte Carlo

The earliest surviving planning for what became the ENIAC Monte Carlo simulations comes in a manuscript dispatched by John von Neumann as a letter to Los Alamos physicist Robert Richtmyer on 11 March 1947. It included a detailed plan for simulation of the diffusion of neutrons through the different kinds of material found within an atomic bomb.[19] The physical model proposed by von Neumann in his initial letter was a set of concentric spherical zones, each containing a specified mixture of three types of material: "active" material where fission would take place, "tamper" to reflect neutrons back toward the core, and material intended to slow the neutrons before a collision took place.[20] The spherical model simplified computation because the only information needed to model a neutron's path was its distance from the center, its angle of motion relative to the radius, its velocity, and the elapsed time.[21]

This established the physical model used the following year on ENIAC. In a 1959 lecture, Richtmyer gave a cogent explanation of the approach taken, writing of the calculations that

> [T]hey were about as sophisticated as any ever performed, in that they simulated complete chain reactions in critical and supercritical systems, starting with an assumed neutron distribution, in space and velocity, at an initial instant of time and then following all details of the reaction as it develops subsequently.
>
> To get an impression of the kind of problem treated in that early work, let us consider a critical assembly consisting simply of a small sphere of some fissionable material like U235 surrounded by a concentric shell of scattering material.[22]

Von Neumann wrote of the proposed computing plan that "[i]t is, of course, neither an actual 'computing sheet' for a (human) computer group, nor a set-up for the ENIAC, but I think that it is well suited to serve as a basis for either." However, his preference for ENIAC was already clear from the detailed consideration he gave to its use and his conclusion that "the problem … in its digital form, is well suited for the ENIAC."[23] At this point, he does not yet seem to have thought of changing ENIAC's programming method.

The maximum complexity of ENIAC programs, in its initial programming mode, was determined by a variety of constraints spread around the machine. These limitations were complex and their interplay depended on the particular program.[24] Von Neumann thought it "likely that the instructions given on this 'computing sheet' do not exceed the 'logical' capacity of the ENIAC."[25] He intended to implement the plan as a single ENIAC setup, segregating on the function tables all the numerical data characterizing a particular physical configuration within the basic geometry being simulated.

Von Neumann planned that each punched card would represent the state of a single neutron at a single moment in time. After reading a card, ENIAC would simulate the next step of this neutron as it moved through the bomb and punch a new card representing its updated status. Random numbers were used to determine the distance the neutron would travel before colliding with another particle. If this took the neutron into a zone containing different material, the neutron was said to have "escaped" and a card was punched recording the point at which it moved from one zone to another. Otherwise, a further random choice was made to determine the type of the collision: the neutron could be absorbed by the particle it hit, in which case it ceased to participate in the simulation; it could bounce off the particle, being scattered with a randomly assigned change in direction and velocity, or the collision could trigger a fission, yielding up to four "daughter" neutrons whose directions were randomly determined. Cards were punched describing the outcome of the collision. The output deck would be fed back in for repeated processing to follow the progress of the chain reaction as far as required.

Von Neumann's eagerness to harness external computing resources for Los Alamos was understandable. Until 1952 Los Alamos itself operated nothing more sophisticated than IBM punched-card machines. So great was its appetite for computer power that a team from the lab had taken control of ENIAC for several weeks from the end of 1945, before it had even been declared fully operational. Several

years later, when word spread that the National Bureau of Standards SEAC (Standards Eastern Automatic Computer) was almost working, Metropolis and Richtmyer arrived from Los Alamos to commandeer it.[26] Code for Los Alamos was also run on IBM's showpiece SSEC (Selective Sequence Electronic Calculator) in its New York headquarters.

In a letter sent to Ulam in March 1947, von Neumann reported that the "computational set-up" was "investigated more carefully from the ENIAC point of view by H.H. and A.K. (Mrs.) Goldstine. They will probably have a reasonably complete ENIAC set-up in a few days. It seems that the set-up, as I sent it to you (assuming third-order polynomial approximations for all empirical functions), will exhaust about 80–90 percent of the ENIAC's programming capacity."[27]

It is unclear how close the Goldstines got to creating a conventional ENIAC setup for Monte Carlo before abandoning this approach. Instead, as discussed in our companion paper, "Engineering 'The Miracle of the ENIAC,'"[3] their efforts shifted, by mid-May at the latest, to a major effort to reconfigure ENIAC to support the modern code paradigm. This project also involved Richard Clippinger and other staff from the Ballistics Research Laboratory, von Neumann himself, and a group of contractors led by Jean Bartik. The change would lift most of the arbitrary constraints that ENIAC's original design had imposed on its versatility as a general-purpose computer. This allowed for development of a considerably more ambitious Monte Carlo program than the one originally sketched by von Neumann, at the cost of deferring its execution until ENIAC had been converted to the new control mode. It also meant abandoning ENIAC's original style of programming, and the experience with this technique gained during its 1945–1946 operation at the Moore School, for the new and so far untested approach associated with the EDVAC design and the machine being built at the Institute for Advanced Studies.

In the second half of 1947, work on computerized Monte Carlo simulation for Los Alamos centered on a single office at the Institute for Advanced Study in Princeton. Its inhabitants included Adele Goldstine and Richtmyer, who had been dispatched from Los Alamos to liaise with what was known informally as the laboratory's "Princeton Annex."[28] However, the focus of their work soon shifted from Monte Carlo to Hippo, a different kind of atomic simulation. Goldstine remained engaged through 1947 and into 1948 on ENIAC coding for Hippo, until this target computer was eventually abandoned in favor of IBM's SSEC.

Primary responsibility for diagramming and coding Monte Carlo seems to have shifted to the third inhabitant of that busy office, Klara ("Klari") von Neumann. Klara Dan had met John von Neumann in 1937, during one of his regular visits to his hometown of Budapest. The next year they divorced their spouses (von Neumann's had already left him) and married each other. It was his second marriage and her third. As war began in Europe, the new Mrs. von Neumann was settling into the role of an academic wife in Princeton. While it raged, her husband grew ever busier, ever more famous, and ever more frequently absent. Their marriage was strained.[29]

Klara was 35 years old when she began to contribute officially to ENIAC conversion planning and Monte Carlo programming around June 1947.[30] Her family had been wealthy and well connected, and she grew up in an intellectually stimulating environment, but her formal education in mathematics and science had finished at an English boarding school. She wrote later that "I had some college algebra and some trig, just enough to pass my matriculation exams, but that was some fifteen-odd years ago and, even then, I only passed the test because my math teacher rather appreciated my frank admission that I really did not understand a single word of what I had learned."[31] She loved the easy camaraderie between the Eastern European scientists she encountered at Los Alamos on a visit at Christmas 1945. According to George Dyson, who profiled her in his recent book *Turing's Cathedral,* "sparks between Klari and Johnny were rekindled and they began collaborating" on his computer work.[32] She took to it with remarkable ease, despite her later and characteristically insecure self-denigration as a "mathematical moron" serving John as an "experimental rabbit" in a Pygmalion-like attempt to create a computer coder from unpromising materials. Programming, she found, was "just like a very amusing and rather intricate jigsaw puzzle, with the added incentive that it was doing something useful."[33]

From Computing Plan to Flow Diagram

As the team in Princeton worked to transform the original computing plan into the design of a fully articulated program for the ENIAC,

it was guided by the programming method that John von Neumann had previously developed with Herman Goldstine. Their reports on "Planning and Coding Problems for an Electronic Computing Instrument," issued in 1947 and 1948, put flow diagrams as the heart of a fairly rigorous approach for the translation of mathematical expressions into machine language programs.[34] This technique was far more nuanced and mathematically expressive than the simplified flowcharts used by the introductory computing texts of later decades to represent sequences of operations.

In his letter to Richtmyer, von Neumann had expressed the computation as a single sequence of 81 simple steps, most of which involved the retrieval or calculation of a single value. Predicates describing certain properties of the current situation were evaluated and then used to specify whether certain subsequent steps should be executed or ignored.

Flow diagrams, on the other hand, explicitly showed the splitting and merging of possible paths of control through a computation. This reflected the modern code paradigm, in which execution paths could diverge following conditional transfer instructions. In some situations, for example when deciding whether a neutron was traveling inward or outward in the assembly, the translation from the computing plan to a flow diagram was fairly straightforward. In other cases, significant changes to the structure of the original computation were required.

Two complete Monte Carlo flow diagrams from 1947 have been preserved, along with a number of partial and summary diagrams, and with the aid of these it is possible to trace the evolution of the first run program in considerable detail, culminating in a complete diagram dated 9 December 1947.[35] The development seems to have followed the notation and methodology laid down in the "Planning and Coding" reports fairly closely and, given the success of the effort, to have demonstrated its utility. The diagrams themselves remain relatively easy to follow and squeeze a great deal of information on different aspects of the program onto a single (in the final version) piece of paper.

The earlier diagram is in the handwriting of John von Neumann. Associated with it are plans for the storage of numerical data in ENIAC's third function table and an estimate of the running time of the program.[36] This was obtained by multiplying the "repetition rate" of each box in the flowchart (the number of times it would be executed in a typical computation) by the time taken to execute the code in that box. This involved knowing the duration of each instruction in "add times," the quantum of time measurement in ENIAC programs. The existence of these estimates implies that Monte Carlo coding was first carried out using a version of the "60 order" instruction set being planned for in fall 1947, although we have so far located only a subroutine to generate random numbers and a small fragment of other code.[37] The timing estimates were later refined with the aid of an overview flow diagram representing just the structure of the computation.[38] This diagram was kept up to date as the design evolved and changes were made to the algorithms in various parts of the calculation, as Table 1 summarizes.

This was a complex program by the standards of the day. The December 1947 flow diagram included 79 operation boxes, many involving multiple computational steps (see Figure 1).

The program was carefully structured into largely independent functional regions. Many of them are single-entrance, single-exit blocks. These regions first appeared in von Neumann's diagram, which was divided into 10 spatially distinct subdiagrams linked by connectors. Twelve regions were made explicit and labeled in the subsequent overview diagram, and they are clearly visible in the December 1947 diagram, where many of the boundaries between the regions are marked by annotations in the form of "storage tables" noting the variables calculated in the preceding region and the accumulator assigned to each.[39]

As Table 1 indicates, the changes in the evolution of the program can be tracked by the rather confusing conventions used to assign numbers to the operation boxes in the flow diagram. The basic sequence of boxes 0* to 54* implemented the functionality of the original computing plan along with the modifications suggested by Los Alamos. The switch from ENIAC's original programming method to its new implementation of the modern code paradigm allowed for a significant expansion of the program's scope and complexity. Thus, this original sequence accounted for little more than half of the eventual code. Changes made as development moved from the original plan to the first Monte Carlo code went far beyond elaborating storage mechanics or expressing

Table 1. Structure and evolution of the Monte Carlo programs.*

Region	Description	Von Neumann/ Richtmyer plan	Von Neumann flow diagram	"First Run" program (from Dec. 1947 flow diagram)	"Second Run" program
		Early 1947	Summer 1947	Spring 1948	Autumn 1948
A	Read card and store the neutron's characteristics. If the neutron is a fission product, calculate new values for its direction and velocity.	0	0*–8*	1*–8*	Restructured Virgin cards: 0–6 Output cards: 10–16
	Calculate parameter λ*, the random parameter used in region E to determine the expected distance to a collision.	N/A (numbers produced externally)	9*–17*	New algorithm 1°–4°	40–45
B	Find neutron's velocity interval; this value is used in region D to find relevant cross-section values.		$\overline{1} - \overline{13}$	Simplified $\overline{1} - \overline{7}$	30–36
C	Calculate distance to zone boundary.	1–15	18*– 23*	18*–23*	20–26
D	Calculate cross-section of material in zone.		$\overline{14} - \overline{17.1}$	$\overline{14} - \overline{17.1}$, 24*	46
E	Determine if terminal event on current trajectory is collision or escape.	16–47	24*–27*	25*–27*	47–49
	Determine if a census comes first.		28*–29*	28*–30*	50–54
F	Discriminate between terminal events. Update neutron characteristics as needed.	47	30*–35*	31*–35*	55–57
G	Refresh random number.		Inline code 6*	Subroutine ρ/ω	Subroutine ρ/ω
H	Determine collision type (absorption, scattering, or fission).	48–53	$\overline{18} - \overline{27}$	$\overline{18} - \overline{27}$	65–69
J	Elastic scattering.	54–59, 61–68	51*–52*	51*–52*	74–76
K	Inelastic scattering.		53*–54*	53*–54*	75–78
L	Absorption/fission.	54–59, 65–81	36*–46*	Simplified 36*–39*, 46*	70–73
M	Print card and restart main loop.	No looping 51, 69, 73, 77, 81	47*–50*	37.1*, 47*–50*	58–64
N	Zonal escape.	48–50	Computation loops without printing		New process 79–85

*The descriptions are ours, but the regional structure and letters assigned are from an original flow diagram. The numbers in cells are the box labels used on the various diagrams.

processes in a different notation to alter the structure of the computational process itself. Among other things, these changes reduced the need for card operations, which were thousands of times slower than electronic processing, and enabled ENIAC to automate ever greater portions of the overall Monte Carlo process.

Change 1: Relaxation of Notation

Looking closely at the succession of draft flow diagrams gives some insight into the experience of using this technique to develop an actual program. Von Neumann's early diagram sticks closely to the notation published in the "Planning and Coding" reports: symbolic names are used for storage locations, and substitution and operation boxes are used in a systematic manner to keep mathematical notation separate from the handling of storage. He extended the notation slightly by including operations in the alternative boxes and by using ad hoc notes within operation and storage boxes to document the effect of input and output operations.

The team seems to have found the complete methodology defined in the IAS reports to be excessively cumbersome, and by December, their use of the flow diagram notation had visibly evolved (see Figure 2). For example, the symbolic labels for storage locations were replaced by explicit references to ENIAC's accumulators. Decisions about data storage had been taken, and there was evidently no perceived benefit in deferring their documentation to a later stage in the process. The careful distinctions made in the reports between the different types of boxes and their contents are becoming increasingly blurred: the substitution boxes that are meant to control loops have largely been replaced by operation boxes, and storage updates that officially should be located in separate operation boxes are appearing with increasing frequency in alternative boxes.

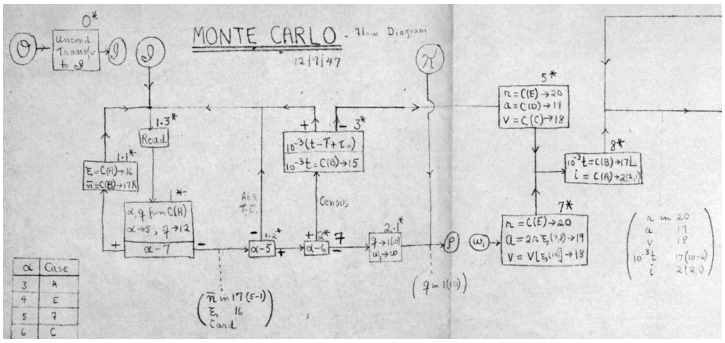

Figure 1. Detail from the December 1947 flow diagram. These largely independent functional regions first appeared in John von Neumann's diagram, which was divided into 10 spatially distinct subdiagrams linked by connectors. (Reproduced from the Library of Congress with permission of Marina von Neumann Whitman. The full diagram, together with other project materials, is available from www.EniacInAction.com.)

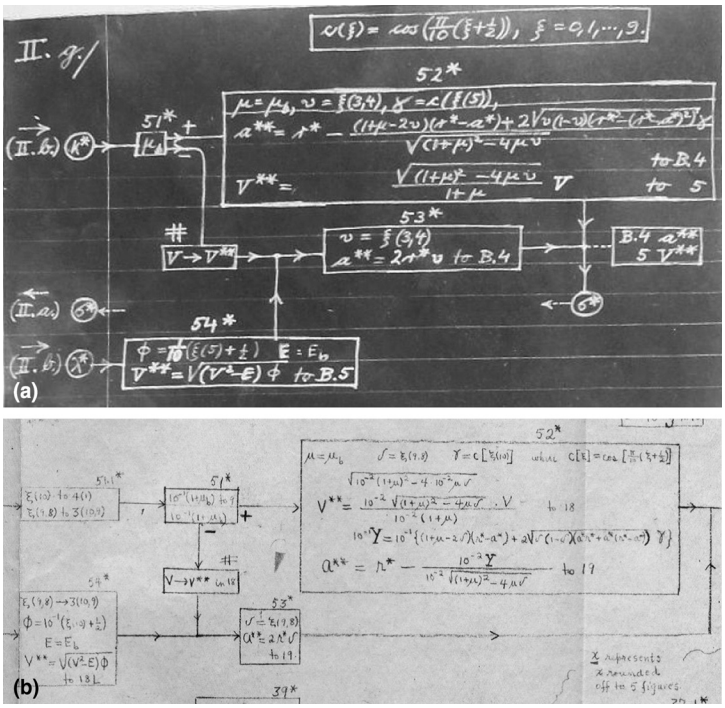

Figure 2. The structure and mathematics of this part of the calculation were almost unchanged between drafts, but in John von Neumann's early draft (white on black) the computation relies on symbolic storage locations (such as B.4). The December 1947 version uses accumulator numbers directly for storage (for example, 19). Note also the complex expressions accommodated within individual flow diagram boxes, such as 52*. (Reproduced from the Library of Congress with permission of Marina von Neumann Whitman.)

Change 2: A Subroutine to Handle Random Numbers

The heart of a Monte Carlo simulation is choosing between different possible outcomes on the basis of their probability. These choices were driven by random numbers, now required in unprecedented numbers. In von Neumann's original plan for the computation, each card fed into ENIAC included the random numbers used to determine a neutron's fate. Getting these numbers onto the cards would have involved an additional process to produce cards holding random numbers, followed by a merge process using conventional punched card machines to create a new card holding both the existing data on a particular neutron (read from one card) and the random numbers (read from another).

As planning continued, von Neumann realized that the ENIAC could itself produce quasi-random numbers. In various letters, he described a technique that involved squaring an eight- or 10-digit number, and then extracting the middle digits to serve as the next value.[40] Thanks to the new programming

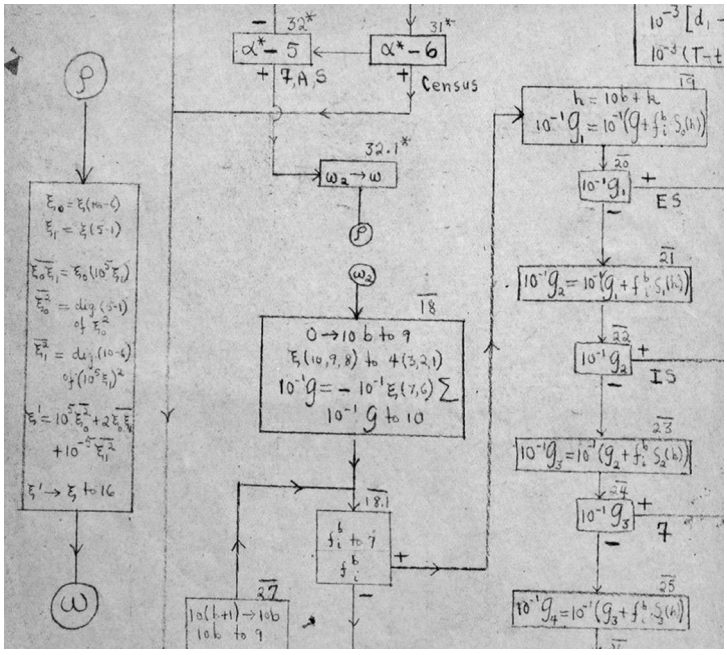

Figure 3. This detail from the December 1947 flow diagram shows both the subroutine (far left) and one of the points from which it is called (the connection ρ following box 32.1*). This box sets the value of the variable remote connection ω to ω_2 so that control will return to box $\overline{18}$ on completion. (Reproduced from the Library of Congress with permission of Marina von Neumann Whitman.)

mode, the logical complexity of a program was limited only by the space available for its instructions on the function table. Thus, these numbers could be produced within the Monte Carlo program itself much more rapidly than they could be read from punched cards.

In the earlier draft flow diagram, a cluster of three boxes representing this process (without providing much detail on the algorithm to be used) was simply duplicated inline at the four points in the computation where a new random number was needed. The December 1947 version placed a detailed treatment of the process in a special box, entered from two separate places in the flow diagram.[41] The random digits it computed were placed in one of the ENIAC's accumulators. This exploited the newly developed concept of a subroutine and demonstrated, apparently for the first time, the incorporation of a subroutine call into the Goldstine and von Neumann flow diagram notation.[42] Their "Planning and Coding" series had not so far dealt with subroutines (these would be covered in the final installment issued in 1948). However, the April 1947 publication did introduce a "variable remote connection" notation that diagrammed code in which the destination of a jump was set dynamically. In the December 1947 flow diagram, a variable remote connection was used to return to the main sequence at the end of the subroutine box (see Figure 3).

The invention of the "closed subroutine," defined by Martin Campbell-Kelly as one that "appeared only once in a program and was called by a special transfer-of-control sequence at each point in the programme where it was needed," is conventionally attributed to David Wheeler's work with the EDSAC, which began operation in 1949.[43] This is distinguished from the "open subroutine," in which a block of code is duplicated as needed, a technique used on the Harvard Mark I some years earlier. We believe that this ENIAC Monte Carlo program was the first code with a closed subroutine to be executed.[44]

Change 3: Lookup of Cross-section Data

The odds that a moving object crashes into an obstacle during a particular time period will change with its velocity, as will the chances of a destructive outcome. In nuclear physics, the probability that a neutron will interact with a nucleus to produce a particular result, such as absorption, fission, or scattering, is referred to as a *collision cross-section*. This depends on both the velocity of the neutron and the nature of the material it is travelling through. Von Neumann's original plan represented the cross-sections as functions of velocity, and he noted that these functions could be tabulated, interpolated, or approximated using polynomials. By the time the earlier flow diagram was produced, he had decided to use lookup tables. The range of possible neutron velocities was divided into 10 intervals, giving 160 possible combinations of collision type, velocity interval, and material. A representative cross-section function value for each combination was stored in a matrix on the numeric function table, and his flow diagram incorporated a new sequence of boxes $\overline{1} - \overline{27}$ to handle the lookup procedure.[45]

A neutron's velocity interval was found by searching through a table of interval boundaries. This search was coded as a loop, providing an early example of an iterative procedure whose purpose was not simple calculation. Once the velocity interval had been located, the appropriate cross-section value could easily be retrieved from the function table by calculating the address corresponding to the current combination of parameters.

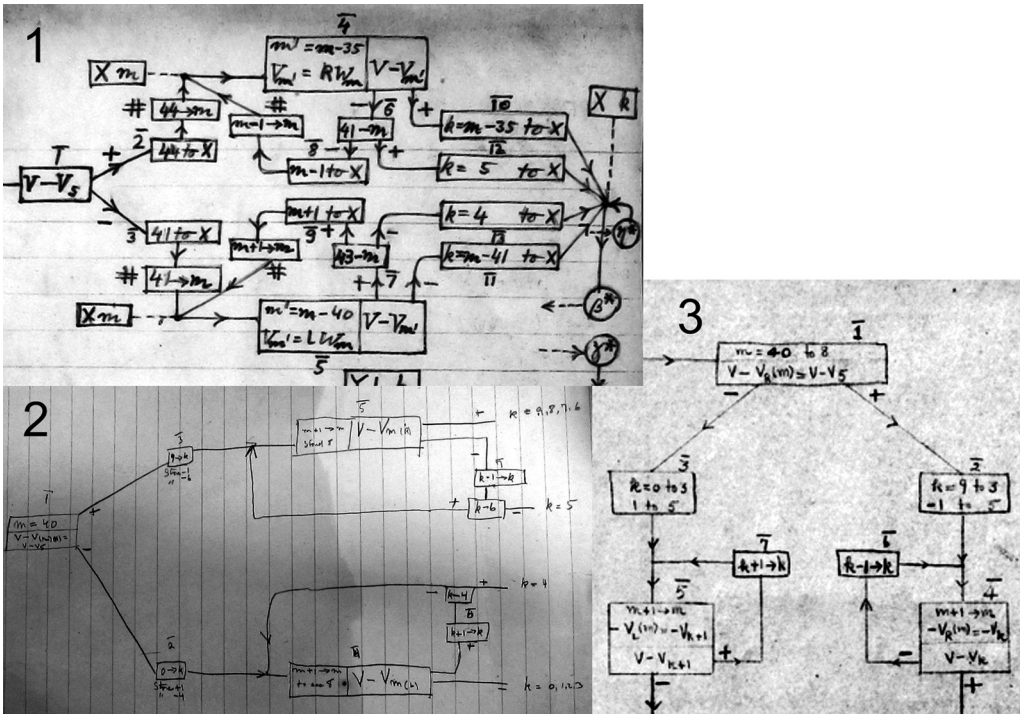

Figure 4. Three progressively more optimized versions of region B of the flow diagram, which finds a neutron's velocity interval. Image 1 is from von Neumann's original flow diagram, and image 3 is from the December 1947 diagram. Image 2 is an intermediate sketch.[46] (Reproduced from the Library of Congress with permission of Marina von Neumann Whitman.)

The design of the search went through a number of revisions. The correct interval for a neutron was found by comparing its velocity with the middle value in the table and then performing a linear search through the top or bottom half of the table, as appropriate. This strategy can be seen in the initial branching in the diagrams shown in Figure 4, which in each case is followed by two similarly structured loops. Originally, the address *m* of the current position in the table was used to control the loop, and the number of the interval, *k*, was then calculated from this in different ways, depending on how the loop had terminated (boxes $\overline{10} - \overline{13}$). This was soon changed, however, so that the interval number itself was used to control the loop, leading to a significant simplification in the expression of the termination conditions. These changes give a vivid impression of the team gradually acquiring a feel for idiomatic techniques of efficient programming in the modern code paradigm.

The introduction of velocity intervals also made it possible to simulate fission more realistically. In the initial plan the "daughter" neutrons produced by fission all had the same velocity. After velocity intervals were introduced, a representative value known as the "center of gravity" was stored for each interval. This allowed different velocities to be easily generated for daughter neutrons by using a digit from the current random number to select a velocity interval.

Change 4: Census Times

The biggest change in scope from the initial computing plan to the program initially executed was the transition from following an individual neutron until it experienced its next "event" to managing a population of neutrons over time. Translating the plan into an ENIAC program made explicit, and partially automated, the work needed to manage multiple neutrons over multiple cycles of simulation. Code implementing the steps defined in the original plan to process one neutron for one event was wrapped in several levels of loops involving both automatic and manual processing steps.

The program was organized around the notion of "census times." This idea was introduced by Richtmyer in his response to von Neumann's original computing plan, when

he observed that it would generate output decks in which cards held snapshots of neutrons at widely different points in time. Richtmyer suggested as a "remedy for this difficulty" to

> follow the chains for a definite time rather than for a definite number of cycles of operation. After each cycle, all cards having t[ime] greater than some preassigned value would be discarded, and the next cycle of calculation performed with those remaining. This would be repeated until the number of cards in the deck diminishes to zero.[47]

These preassigned values were dubbed "census times." A statistically valid picture of the overall neutron population at these points would then be produced, just as governments make measurements of the characteristics of their national populations at certain periodic dates. The census concept was widely adopted for Monte Carlo simulations.[48]

Change 5: Simulating Multiple Events Per Cycle

According to the original computing plan, each cycle of computation would track the progress of a neutron only as far as its next event (scattering, zonal escape, total escape, absorption, or fission). One or more new cards representing the consequences of the event would then be produced. The additional logical complexity afforded by the new programming mode made it possible for ENIAC to simulate more than one event in a neutron's career before printing a new card for it. If a neutron was scattered or moved into a new zone but had not yet reached the census time, the program branched back to an earlier region to follow its progress further rather than producing an output card immediately. This increased the complexity of the program but reduced the amount of manual card processing required.

Figure 5 gives an overview of how all this worked in practice. The initial stack of neutron cards was read one at a time from the input hopper. After reading each card, ENIAC punched one or more output cards. If a neutron reached the current census time without incident, it was followed no further for the moment and ENIAC output a "census" card with its updated information. If a neutron was absorbed or experienced total escape, then its career as a free neutron within the simulation was over, but a card identifying the nature of the terminal event was nevertheless output for analytical processes. Likewise, when a neutron triggered fission, a terminal card for that neutron was produced specifying that fission had taken place, including a "weight" field to indicate the number of daughter neutrons set free.

ENIAC's operators would then use a suitably configured sorting machine to separate the output deck into three trays. One tray accumulated cards representing neutrons that did not need to be processed again because they had terminated their simulated career after escaping or being absorbed. Another tray accumulated census cards representing neutrons that had reached the current census time without incident.

The third tray held the cards representing simulated fission events. Because these fissions had taken place before the end of the current census period, the cards were carried over to ENIAC's input hopper for further processing.[49] The data on this card was read once, but ENIAC processed it repeatedly to simulate each daughter neutron generated by the fission. The careers of these daughter neutrons were followed as normal, with one card being punched for each when a terminal event was reached. These output cards were sorted again, in case any further fissions had occurred, and the process repeated until ENIAC's output deck included no fission cards, indicating that each neutron had been followed up to the census time.

At this point, the target census time could be incremented and the simulation could move on to the next census period. The pile of census cards, representing neutrons that were still active at the end of the previous period, provided the new input deck. However, it needed additional manual processing before use. The team had decided to start each census period with the same number of neutrons, even though the number of neutrons careening around inside a bomb tends to rise or fall precipitously after its detonator is triggered. A larger neutron sample population increased statistical confidence in the results of the simulation but increased the time, work, and pile of punched cards needed to run it. A smaller neutron sample population could be processed more quickly but its behavior would be less likely to track the larger system being simulated. Thus, allowing the neutron sample size to grow or shrink in proportion to the simulated population would sacrifice either accuracy or practicality. Cards from the deck were therefore manually duplicated or discarded to create a new input deck of 100 cards.

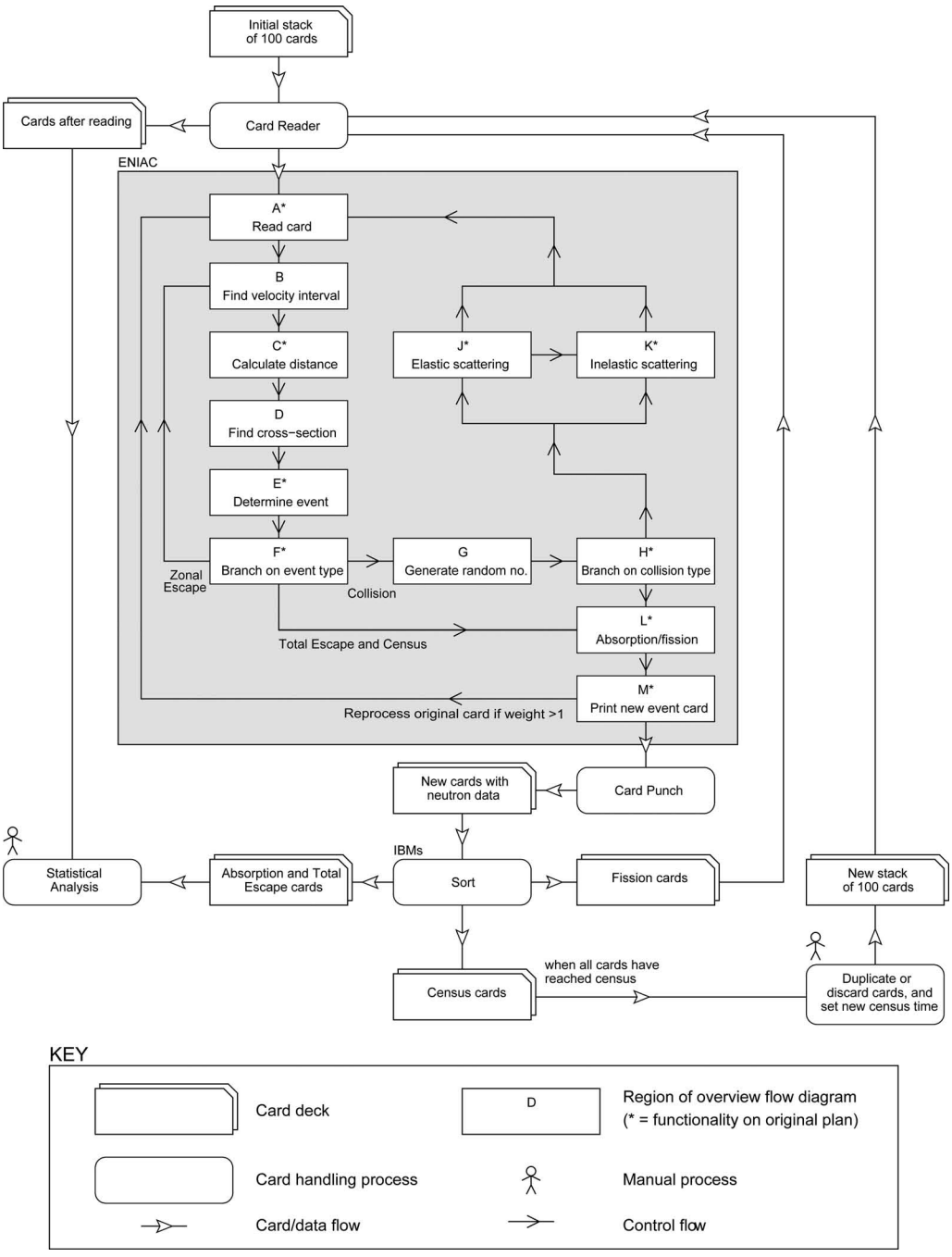

Figure 5. Simulating multiple events per cycle. The shaded region shows the structure of the "First Run" Monte Carlo program, including program regions. The unshaded region shows the card operations required outside ENIAC.

All the cards punched during the computation were retained. These could be analyzed to show the distribution of neutrons and fissions in time and space, something Richtmyer had mentioned as desirable. They would also reveal trends in neutron velocity, the relative frequency of events such as escape or fission at that instant, and the rate at which the population of free neutrons was increasing or decreasing. Tracking the latter would allow Los Alamos to estimate the overall size of the free neutron population at each point in the simulation (something that ENIAC did not itself track).

Making the First Run, April–May 1948

The three main sets of Monte Carlo chain reactions carried out for Los Alamos in 1948 and 1949 were referred to by the von Neumanns as the "first run," "second run," and "third run." Within each run a different evolution of the program code was used to investigate a number of "problems," each simulating a different physical configuration. The term "run" may echo the idea, later ubiquitous, of "running" a program or may refer instead to the physical movement of people and materials to Aberdeen and back—in the sense of a "bombing run" or "school run." The term "expedition" was also used by John von Neumann to refer to these trips and to later trips made to Aberdeen to experiment with numerical weather predictions.[50] It is a striking term, capturing a world in which computers were scarce and exotic things by evoking the scientific tradition of mounting long, arduous, and painstakingly planned field expeditions to observe eclipses, uncover buried cities, or explore the polar regions. Explorers returned with knowledge that could never be obtained at home. Using ENIAC was an adventure, a journey to an unfamiliar place, and often something of an ordeal.

Anne Fitzpatrick, who wrote with access to internal Los Alamos progress reports and other classified documents, concluded that all ENIAC Monte Carlo work done for Los Alamos in 1948 and 1949 was focused on fission weapons, rather than relating directly to the fusion-powered hydrogen bomb. The initial set of seven calculations in spring 1948 were "primarily for the purpose of checking techniques, and according to Metropolis, did not attempt to solve any type of weapons problem." It is, however, clear that Los Alamos viewed the work on ENIAC as crucial to its own progress with new weapons designs. Fitzpatrick continues,

> Throughout March and April Carson Mark [director of the lab's Theoretical Division] complained in his monthly reports about the delays encumbered (sic.) by the fission program because of the slow pace of the ENIAC's conversion and "mechanical condition." The whole point of having fission problems run on ENIAC in the first place, Mark noted, was to speed up T Division's work by "mechanization" of calculations.[51]

Even a man as well connected as John von Neumann could not simply show up at the Ballistic Research Laboratory in Aberdeen and ask his friends for the keys to ENIAC. There were bureaucratic niceties to follow and a chain of command to respect. On 6 February 1948, he wrote to Norris Bradbury, director of Los Alamos, asking him to make a formal request for time on the ENIAC. Political as ever, von Neumann reminded Bradbury to "mention in your letter how much you appreciate all the courtesies that have been extended to your staff by the BRL and how extremely important the ENIAC is to your work, etc. This will greatly help Colonel Simon politically and will also be good for our future relations with the BRL."[52]

The terms had already been worked out informally with Simon and his subordinates, with delays imposed by the arrival of ceiling contractors, testing, and the delayed installation of new hardware.[53] Bradbury sent the necessary official request, flattery included, to the Office of Chief of Ordnance. On 13 March 1948, von Neumann wrote to Carson Mark, leader of the Theoretical Division at Los Alamos, that "The Monte Carlo problem is set up and ready to go."[54]

The von Neumanns visited Aberdeen on 8 April 1948, and Klara remained behind for the next month to work with Metropolis, who had by then completed initial reconfiguration of ENIAC to support the modern code paradigm. The first successful "production run" made on Monte Carlo took place on 17 April 1948. However, the machine was still plagued by "troubles" and the calculation did not begin in earnest until 28 April.[55] Progress seems to have been slowed far more by hardware glitches than program bugs, which is interesting given that nobody had previously debugged a program written in the modern code paradigm. This is a tribute to the care with which the program had been planned, the depth of thought the Princeton group had given to the new programming style, and perhaps also to ENIAC's relative friendliness to debugging.

We have not located the program code for the first run. However, triangulating from several sources gives a reliable sense of how this first Monte Carlo worked. First, we have the series of flow diagrams discussed previously. Second, we have the revised program code used for the second run in late 1948. Third, we have a lengthy archival document, "Actual Running of the Monte Carlo Problems on the ENIAC," which describes the programming techniques used in both versions. This explicitly highlights changes made between the two runs. Fourth, the draft

flow diagrams for the second run are in many regions identical to the first run diagram. Finally, archival materials for the first run document the allocation of data to the ENIAC's accumulators, the layout of constant data on the third function table, the card format used, and the associated use of the constant transmitter.[56]

The calculations performed during the first run simulated seven different situations, each represented by changing some of the data stored on the third function table. As Richtmyer wrote,

> Certain experimentally determined nuclear data are obviously needed. One must know the so-called macroscopic cross-sections, that is, the probabilities, per unit distance travelled, of the various processes (absorption, elastic scattering, inelastic scattering) in each medium, as a function of the neutron's velocity. For scattering, one must know the angular distribution, that is, the relative probabilities of various angles of scattering; for inelastic scattering one must also know the energy distribution of the scattered neutrons; and, for fission, one must know the average number and energy distribution of the emitted neutrons.[57]

This data was the most militarily sensitive part of the entire operation. Documents retained in the archives record in triplicate the receipt of classified material by Klara von Neumann on various occasions, sometimes (as with "cross section data" on 16 January 1947) from her husband.[58]

Our companion article "Engineering 'The Miracle of ENIAC'"[3] discusses several draft instruction sets created for ENIAC during the planning process. For the first run program, ENIAC was set up to implement a code providing 79 instructions.[59] This would not have required major changes to draft programs targeting the "60 order code." Beyond updating the numerical codes corresponding to the instruction mnemonics, a simple matter of substitution, the main challenge would have been restructuring shift instructions as these were handled quite differently in the new instruction set.

By 10 May 1948 it was all over. John von Neumann wrote on 11 May to Ulam that "ENIAC ran for 10 days. It was producing 50% of these 10x16 hours, this includes two Sundays and all troubles…. It did 160 cycles ('censuses,' 100 input cards each) on 7 problems. All interesting ones are stationary at the end of this period. The results are very promising and the method is clearly a 100% success."[60] Three days later, he added that "There is now a total output of over 20,000 cards. We have started to analyze them, but … it will take some doing to interpret it."[61]

Klara von Neumann documented the techniques used in a manuscript cryptically headed "III: Actual Technique—The Use of the ENIAC."[62] This began with a discussion of the conversion of ENIAC to support the new code paradigm, documented the data format used on the punched cards, and outlined in reasonable detail the overall structure of the computation and the operations performed at each stage.

Second Run, October–November 1948

Klara von Neumann returned to Aberdeen on 18 October 1948 to perform a second run of Monte Carlo calculations and was joined by Metropolis two days later. Production work began on 22 October. On 4 November, John von Neumann wrote to Ulam that "[t]hings at Aberdeen have gone very well. The present segment of the Monte Carlo program is likely to be completed at the end of this week or early next week."[63] According to Fitzpatrick, this second series of problems "constituted actual weapons calculations" including "an investigation of the alpha for UH3, a 'hydride' core implosion configuration" and "a supercritical configuration known as the Zebra."[64]

Changes between the first and second versions of the Monte Carlo program were described in some detail in the report "Actual Running of the Monte Carlo Problems on the ENIAC." An expanded and updated version of the earlier "Actual Technique" report, this was written by Klara von Neumann and edited in collaboration with Metropolis and John von Neumann during September 1949. It contains a detailed description of the computations, highlighting the changes in the flow diagram, program code, and manual procedures between the two versions.[65]

The physical model used and the calculations performed to follow the paths of individual neutrons were little changed. Modifications were made to representations of the zones of different materials and of the zonal escape of neutrons. Most changes were operational optimizations. For example, some early sections of the program were reordered to marginally increase overall efficiency, and collision cross-section ratios were precomputed and stored on the function table to avoid recalculating them when needed.

The most important change was to further increase the share of the overall Monte Carlo procedure being automated by ENIAC. During the first run, ENIAC finished processing a neutron when its path had been followed to the next census time. When a neutron reached a census time, its data was punched onto a card, which was then sorted into a separate pile. Only after all the fissions that occurred during that census time had been processed was the card read back in and the next period in the neutron's life simulated. Processing the entire neutron population for one census period before proceeding to the next allowed von Neumann and Metropolis to intervene after each census time to adjust the population size, but it also introduced considerable inefficiency. Sorting work was needed every time a deck of cards was fed through ENIAC, and the deck was examined and modified manually when processing for each census period was completed.

The second run eliminated both kinds of manual processing by including "in the logical sequence and coding of the program, an automatic way of handling the beginning and ending of a time cycle."[66] "No attempt was made to keep the input stack at a fixed number," according to Klara von Neumann's report. Indeed, each surviving neutron at the end of a census period gave rise to two neutrons in the next period to ensure that the sample population expanded even if the simulated population was falling.[67]

As before, reading a fission or census card would further progress the life history of the neutron(s) described on the card. It was no longer necessary, however, to process all the neutrons produced by fission during a census interval before rebalancing the deck and proceeding to the next interval. There was thus no need to separate fission and census cards from each other, or from other card types. Each deck of cards punched by ENIAC could be transferred immediately to its input hopper for further processing.[68]

Total escape, absorption, or fission still ended a free neutron's career. The handling of zonal escape changed to limit the amount of time spent processing neutrons that were scattered in the large outer zone of tamper material found in several second run problems. Each time a neutron escaped from one zone to another in the second run, a card was punched and the computation continued with the next neutron, whereas in the first run these "zonal escapes" had not been logged. To make sorting easier at Los Alamos, each card was punched with the number of the census interval it represented.

The new operating procedures distinguished between subcritical and near- or super-critical reactions. For subcritical systems, the original cards were given start times spread over the course of the simulation to increase the proportion still active at its conclusion. The new flow diagram contained two separate card read sequences, and the appropriate one was selected by modifying a single address on the function table before the computation was started. This technique was also used to control certain aspects of the processing of zonal escape and to include special code sections needed for particular problems.

ENIAC finished processing this series of problems by 7 November 1949. On 18 November, John von Neumann wrote that "The whole second Monte Carlo seems to have been successful. The ENIAC functioned marvelously. About 10^5 cards were produced in three weeks, and while the material hasn't been analyzed as yet, there is reason to hope that it contains a considerable amount of valued and useful information."[69]

We uncovered two flow diagrams describing this run, along with a complete code listing that very closely matches one of them.[70] The two diagrams exhibit similar levels of formality suggesting that a stable usage of the flow diagram methodology was emerging with experience.

The code listing, in Klara von Neumann's handwriting, covers 28 pages (see Figure 6). It is broken up into sequentially numbered six-line sections. In many sections annotations provide a simulated trace of the progress of the program, using typical data values to check the effect of the orders.

The program was faithful to the flow diagrams' novel features, such as calling the subroutine from two places in the code and storing the appropriate return address, and reflected its structure of regions with fairly disciplined entry and exit points. Its main sequence filled rows –2, –1, and 12–99 of the first function table and rows –2 to 96 of the second function table. With four of these rows blank, this accounted for approximately 2,208 digits of program code, representing about 840 instructions.[71] The first run was of similar complexity—some aspects were simplified for the second run, but this was balanced by the addition of some new data fields and automation of some tasks that required manual card sorting in the initial version.

The code for the second run included a number of variant paths and sections and could be configured for a specific problem by manually setting a handful of transfer addresses within instructions on the function tables before execution was started. The most significant of these is a section of code dealing with the elastic scattering of neutrons in "light materials." This reflected interest at Los Alamos in the use of uranium hydride cores. The hydrogen separates from the uranium, acting as a moderator to slow neutrons and reduce the critical mass of uranium needed to build a weapon. Edward Teller believed, wrongly as it turned out, that the inevitable reduction in explosive yield would be more than offset by of the opportunity to build more bombs from scarce weapons-grade uranium.[72]

Klara von Neumann left for Los Alamos around 1 December, presumably to help interpret the second run results and to lay the groundwork for future calculations. A letter from her husband, who arrived some weeks later, expressed "dense confusion" after what seemed to be some kind of argument triggered by a crisis of confidence on her part as she prepared to provide "proof of [her] intellectual independence" via this solo trip.[73] Even someone blessed with ample self-confidence and robust mental health would feel somewhat daunted at the prospect of defending one's mathematical technique to Ulam, Teller, and Enrico Fermi (the latter already a Nobel Prize winner). On 13 December, John wrote to her at Los Alamos to admit himself "scared out of my wits" after finding her "catastrophically depressed" during a phone call and worrying that the stress would leave her ruined "physically and emotionally." Seeking independence within the shadow of John von Neumann only added to the challenges facing Klara, despite his fervent attempts to allay her worries about her loss of youth ("your problems and dispositions are perennial, and age is the least of your troubles"), intelligence ("a bright girl") and flawed character ("and a very nice one").[74]

Her worries could only have been compounded when an error was discovered in the hydride calculations. Ulam wrote to John von Neumann on 7 February that "It seems that our electronic computation is wrong in the Problem No. 4—Nick found out which it was. The problem has to be repeated."[75] Carson Mark complained in his regular Theory Division progress report that it was "evident that the ENIAC has not advanced beyond an experimental stage in doing serious computation for this project."[76] These gripes did not deter further use of ENIAC or slow Monte Carlo's rapid enshrinement as an indispensable tool for nuclear science. ENIAC and Klara von Neumann hosted at least three further Monte Carlo expeditions by the staff of Los Alamos and Argonne labs during 1949 and 1950 before more powerful computers became available for their use.

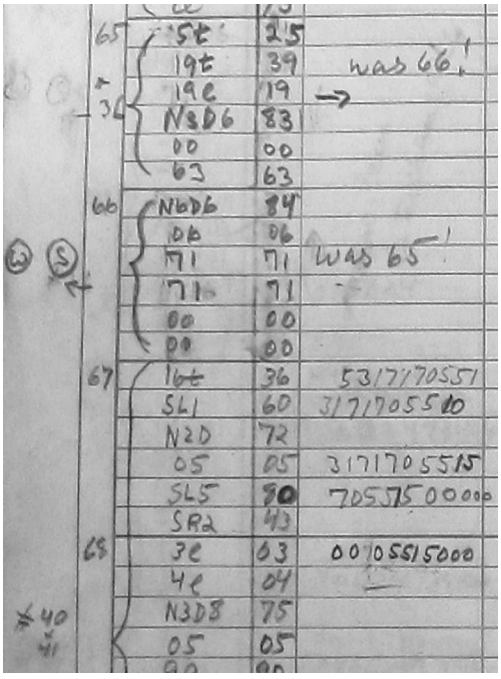

Figure 6. Detail from page 7 of the second run program code, showing 13 of the 840 instructions in the program. Numbers 65–68 show positions on the function tables, and annotations in the left margin refer back to the corresponding boxes on the flow diagram. Each row gives the two decimal digits entered on the function table, and when those numbers code an operation rather than a data field, the corresponding mnemonic such as 3l or N3D8. Some corrections have been made in red pencil, and blocks 65 and 66 of code have been erased and substituted for each other. (Reproduced from the Library of Congress with permission of Marina von Neumann Whitman.)

Conclusions

John von Neumann's contribution to the development of modern computing is well known, and the roles of some of his collaborators such as Herman Goldstine and Arthur Burks are also well documented. Our investigation has shed new light on the importance

of work done by some of his other collaborators, most notably his wife Klara. Her central contribution to the Monte Carlo work has, with the exception of previously cited comments by Nick Metropolis and recent coverage by George Dyson, barely been mentioned.[77] The story told here fits, in its broad outline, with Galison's famous depiction of the first Monte Carlo simulations as a trading zone, yet stepping down from his lofty perch to look more closely at the details of the computations deepens our understanding of what is being traded and by whom.

According to Galison, Monte Carlo led physics to a "netherland that was at once nowhere and everywhere."[78] This description of its intellectual legacy also describes its unconventional social structure, creating opportunities in the shadows. The practice of Monte Carlo engaged not only the great men populating his story, who were diverse in their disciplinary backgrounds, but a broader cast of characters. In particular, Klara von Neumann, mentioned by Galison only as one of a list of early female computer programmers, emerges as a surprisingly central participant in these exchanges. Entering the trading zone with no scientific credentials, carrying little more than her natural talent and some social capital borrowed from her husband, she was soon running a successful stall of her own.

The ENIAC Monte Carlo simulations executed from spring 1948 onward stand out among the programs executed during the 1940s for their complexity and the fidelity of their diagramming and coding style to the ideas of John von Neumann and his circle of collaborators. Our analysis illuminates the evolution of the program over a two-year period from an initial computing plan, through a series of flow diagrams to an initial ENIAC program and onward through a major cycle of revision and improvement. This gives a uniquely detailed and richly substantiated look at a landmark program.

The program challenges some generalizations about scientific computing. We tend to think of the speed and practical scope of early scientific computer problems as governed largely by computation speed and memory requirements with minimal requirements for input and modest output needs, in contrast with administrative data processing jobs where throughout depended on the speed at which data could be pushed in and out of the machine from card or tape units. ENIAC's original task of calculating trajectory tables certainly fits this model, as do the celebrated "first programs" run on the Manchester Baby and EDSAC (both of which performed long series of calculations with no data input and a tiny output consisting of solutions).[79] In contrast, the first program run on ENIAC after its conversion to the new code paradigm was a complex simulation system that might take days to complete its tasks, depending primarily on the speed at which data could be fed through the machine.

The program included a number of key features of the modern code paradigm. It was composed of instructions written using a small number of operation codes, some of which were followed by additional arguments. Conditional and unconditional jumps were used to transfer control between different parts of the program. Instructions and data shared a single address space, and loops were combined with index variables to iterate through values stored in tables. A subroutine was called from more than one point in the program, with the return address stored and used to jump back to the appropriate place on completion. Whereas earlier programs, such as those run on the Harvard Mark I, were written as a series of instructions and coded numerically, this was the first program ever executed to incorporate these other key features of the modern code paradigm.

Acknowledgments

This project was generously funded by Mrs. L.D. Rope's Second Charitable Settlement. Thanks to archivists Susan Dayall at Hampshire College, Lynn Catanese at the Hagley Museum and Library, Susan Hoffman at the Charles Babbage Institute, Valerie-Ann Lutz and the other archival staff at the American Philosophical Society, and the staff of the Library of Congress Manuscripts Reading Room. Nate Wiewora, Alan Olley, and Peter Sachs Collopy provided us with copies of documents. George Dyson and Marina von Neumann Whitman both shared unpublished material with us from the latter's personal collection of papers concerning Klara von Neumann. Susan Abbey provided handwriting analysis services to clarify the authorship of numerous documents. Anne Fitzpatrick, Steve Aftergood, Robert Seidel, J. Arthur Freed, and Alan B. Carr all did what they could to help us navigate the maze of restrictions surrounding access to historical

materials from Los Alamos. William Aspray, Jeff Yost, Atsushi Akera, Paul Ceruzzi, and Martin Campbell-Kelly kindly answered our questions on specific topics and shared their perspectives on computing in the 1940s. We also benefitted from suggestions made by the *Annals* reviewers and by those who discussed the draft at the informal Workshop on Early Programming Practice, organized by Gerard Alberts and Liesbeth De Mol.

References and Notes

1. J. von Neumann, "First Draft of a Report on the EDVAC," *IEEE Annals of the History of Computing*, vol. 15, no. 4, 1993, pp. 27–75.
2. T. Haigh, M. Priestley, and C. Rope, "Reconsidering the Stored Program Concept," *IEEE Annals of the History of Computing*, vol. 36, no. 1, 2014, pp. 4–17.
3. T. Haigh, M. Priestley, and C. Rope, "Engineering 'The Miracle of the ENIAC': Implementing the Modern Code Paradigm," *IEEE Annals of the History of Computing*, vol. 36, no. 2, 2014, pp. 41–59.
4. An overview of perspectives on the history of software toward the beginning of this period is given in U. Hashagen, R. Keil-Slawik, and A.L. Norberg, eds., *Mapping the History of Computing: Software Issues*, Springer-Verlag, 2002.
5. Changes in the focus of software history over time are explored in M. Campbell-Kelly, "The History of the History of Software," *IEEE Annals of the History of Computing*, vol. 29, no. 4, 2007, pp. 40–51.Recent contributions include M. Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*, MIT Press, 2003; T. Haigh, "How Data Got Its Base: Information Storage Software in the 1950s and 1960s," *IEEE Annals of the History of Computing*, vol. 31, no. 4, 2009, pp. 6–25; M. Priestley, *A Science of Operations: Machines, Logic, and the Invention of Programming*, Springer, 2011.Our specific topic, the programming of ENIAC, has recently been explored in B.J. Shelburne, "The ENIAC's 1949 Determination of π," *IEEE Annals of the History of Computing*, vol. 34, no. 3, 2012, pp. 44–54; and M. Bullynck and L. De Mol, "Setting-Up Early Computer Programs: D.H. Lehmer's ENIAC Computation," *Archive for Mathematical Logic*, vol. 49, no. 2, 2010, pp. 123–146.
6. See, for example, the perspectives gathered in N. Oudshoorn and T. Pinch, eds., *How Users Matter: The Co-Construction of Users and Technology*, MIT Press, 2003.
7. P. Galison, "Computer Simulation and the Trading Zone," *The Disunity of Science: Boundaries, Contexts, and Power*, P. Galison and D.J. Stump, eds., Stanford Univ. Press, 1996, p. 119.
8. Galison, "Computer Simulation and the Trading Zone," p. 120.
9. M.S. Mahoney, "Software as Science—Science as Software," *Mapping the History of Computing: Software Issues*, U. Hashagen, R. Keil-Slawik, and A.L. Norberg, eds., Springer-Verlag, 2002, pp. 25–48. U. Hashagen, "The Computation of Nature, Or: Does the Computer Drive Science and Technology?" *The Nature of Computation. Logic, Algorithms, Applications*, P. Bonizzoni, V. Brattka, and B. Löwe, eds., LNCS 7921, Springer-Verlag, 2013, pp. 263–270. The philosophical status of early Monte Carlo simulation was recently explored in a PhD thesis: I. Record, *Knowing Instruments: Design, Reliability, and Scientific Practice, History and Philosophy of Science and Technology*, Univ. of Toronto, 2012.
10. The jump takes place at the bottom of page 130. Pages 130–135 then discuss the application of Monte Carlo methods to the Super, including ENIAC calculations performed in 1950.
11. A. Fitzpatrick, "Igniting the Light Elements: The Los Alamos Thermonuclear Weapon Project, 1942–1952 (LA-13577-T)," Los Alamos Nat'l Lab., 1999.
12. D. MacKenzie, "The Influence of Los Alamos and Livermore National Laboratories on the Development of Supercomputing," *IEEE Annals of the History of Computing*, vol. 13, no. 2, 1991, pp. 179–201.
13. W. Aspray, *John von Neumann and the Origins of Modern Computing*, MIT Press, 1990.
14. The series of 1947–1948 reports on "Planning and Coding Problems for an Electronic Computer" are reproduced in W. Aspray and A.W. Burks, *Papers of John von Neumann on Computing and Computer Theory*, MIT Press, 1987, pp. 151–306.
15. S.M. Ulam, *Adventures of a Mathematician*, Scribner, 1976, p. 148.
16. Fitzpatrick, "Igniting the Light Elements," p. 269.
17. Ulam, *Adventures of a Mathematician*, pp. 196–201. Another first-hand account is given in N. Metropolis, "The Beginning of the Monte Carlo Method," special issue, *Los Alamos Science*, 1987. Several secondary treatments are cited in subsequent notes.
18. Aspray, *John von Neumann and the Origins of Modern Computing*, pp. 111, 288, footnote 50. This public mention of Monte Carlo seems to precede the well-known paper by von Neumann and Ulam presented in September 1947 and published as S.M. Ulam and J. Von Neumann,

"On Combination of Stochastic and Deterministic Processes: Preliminary Report," *Bull. of the Am. Mathematical Soc.*, vol. 53, no. 11, 1947, p. 1120.

19. C.C. Hurd, "A Note on Early Monte Carlo Computations and Scientific Meetings," *Annals of the History of Computing*, vol. 7, no. 2, 1985, pp. 141–155. The report reproduced there is the source for much subsequent discussion of the planned computation, including Galison, "Computer Simulation and the Trading Zone," pp. 129–130, and Record, *Knowing Instruments*, pp. 137–141.
20. Richtmyer's reply (also reprinted by Hurd in his 1995 article) points out that the "slower-down material" could be omitted for "systems of interest to us [at Los Alamos]" (that is, bombs). This suggestion was followed in the first version of the program, although the layer was eventually reintroduced to allow simulation of bombs with uranium hydride cores.
21. Von Neumann also proposed recording the current zone number to save having to calculate this based on the neutron's position.
22. R.D. Richtmyer, "Monte Carlo Methods: Talk Given at the American Mathematical Society, April 24, 1959," p. 3. Stanislaw M. Ulam Papers, Am. Philosophical Soc., series 15. (Further citations to this collection are abbreviated SMU-APS.)
23. Hurd, "A Note on Early Monte Carlo Computations and Scientific Meetings," p. 149.
24. These limitations are discussed further in T. Haigh, M. Priestley, and C. Rope, "Engineering 'The Miracle of the ENIAC,'" p. 43.
25. Hurd, "A Note on Early Monte Carlo Computations and Scientific Meetings," p. 152.
26. G. Dyson, *Turing's Cathedral: The Origins of the Digital Universe*, Pantheon Books, 2012, p. 210.
27. J. von Neumann to Ulam, letter, 27 Mar. 1947, SMU-APS, series 1, John von Neumann folder 2.
28. "I am hoping to hear very soon from the 'Princeton Annex' some word of the first Monte Carlo." C. Mark to J. von Neumann, letter, 7 Mar. 1948, Papers of John von Neumann, Manuscripts Division, US Library of Congress, box 5, folder 13. (This collection is cited hereafter as JvN-LOC).
29. Dyson, *Turing's Cathedral*, pp. 175–189, focuses on Klara von Neumann, as does M.v.N. Whitman, *The Martian's Daughter: A Memoir*, Univ. of Michigan Press, 2012, pp. 22–23, 38–39, 48–54.
30. A letter dated 28 Aug. 1947 from A.W. Kelley to Richtmyer confirms that the "necessary approvals have been obtained" for her employment by Los Alamos. JvN-LOC box 19, folder 7. However, her informal involvement seems to have preceded this.
31. K. von Neumann, "Grasshopper in the Very Tall Grass," memoir (n.d.), Papers of Marina von Neumann Whitman, 1946–2013, Schlesinger Library, Radcliffe Inst. for Advanced Study, Harvard University. Transcription by M. von Neumann Whitman.
32. Dyson, *Turing's Cathedral*, p. 188.
33. These quotations are taken from Klara von Neumann's "Grasshopper in Very Tall Grass.".
34. W. Aspray and A. Bucks, "Computer Programming and Flow Diagrams: Introduction," *Papers of John von Neumann on Computing and Computer Theory*, W. Aspray and A. Bucks, ed., MIT Press, 1987, pp. 145–150.
35. The best developed version of the first run flow diagram measured approximately 24 inches by 18 inches and is neatly written in the hand of Adele Goldstine with the heading "MONTE CARLO Flow Diagram 12/9/47," JvN-LOC, box 11, folder 7. (There are two copies, one a negative.) A copy with two later handwritten annotations is in the Herman H. Goldstine Collection, 1941–1971 Archives, Hampshire College (hereafter cited as HHG-HCA). This diagram is available online from our project website www.EniacInAction.com as is a technical report by M. Priestley and T. Haigh, "Monte Carlo Computation Analysis," charting in more detail the evolution of planning for the computation.
36. Ten manuscript pages numbered I, II.a–II.g, III, and IV, in JvN-LOC, box 11, folder 8. An undated manuscript page on squared paper in JvN-LOC box 11, folder 8, contains a plan of ENIAC's three function tables, labeled "FT I," "FT II," and "FT III," and shows that all three tables could be used for the storage of program instructions or numeric data. Common practice, followed in the Monte Carlo programs, was to use two tables to store the program code and the third, referred to as the "numeric function table," to hold data describing a particular physical situation. This strategy preserved the separation of the setup of the Monte Carlo procedure itself from what von Neumann referred to in his original letter to Richtmyer as "numerical constants" describing a particular "criticality problem." (Hurd, "A Note on Early Monte Carlo Computations and Scientific Meetings," p. 149).
37. Undated manuscript page headed "Refresh Random No.," JvN-LOC, box 11, folder 8. J. von

Neumann was working personally on the methods for the generation of random numbers, so this might well have been written before or separately from the rest of the program.

38. Seven undated manuscript pages numbered 0 to 6 and a single page headed "Shifts" in JvN-LOC, box 11, folder 8. The structure of the overview flow diagram on page 0 is reproduced in the shaded area of Figure 5. Additional diagrams on pages 1–3 represented the operation boxes and the connections between them in each of the 12 regions. Pages 4–6 contained detailed timing estimates for each box and region.
39. Three storage tables can be seen in Figure 1, one attached by a dashed line to the line between boxes 1* and 1.2* and e.g. one to the right of box 7*.
40. Von Neumann talks about the "square and take the middle digits" approach to generating pseudorandom numbers, and testing the resulting distribution, in letters to A.S. Householder (3 Feb. 1948) and C.C. Hurd (3 Dec. 1948), *J. von Neumann, Selected Letters*, M. Rédei, ed., Am. Mathematical Soc., 2005, pp. 141–142, 144–145.
41. This was another minor optimization: two of the four points at which new numbers had been generated were, by late 1947, modified to make use instead of particular digits within the number already generated.
42. The idea of a subroutine was familiar within the ENIAC team as early as 1945: "*It is possible to have the main routine divided into sub-routines, in which case one stepper is used to feed another stepper, thus allowing the proper sub-routine to be chosen in the course of a regular routine.*" J.P. Eckert, J.W. Mauchly, H.H. Goldstine, and J.G. Brainerd, *Description of the ENIAC and Comments on Electronic Digital Machines*, AMP Report 171.2R, distributed by the Applied Mathematics Panel, Nat'l Defense Research Committee, 30 Nov., Moore School of Electrical Eng., 1945, pp. 3–7. This predates the earliest use of the term recorded in the *Oxford English Dictionary*, which documents a usage by J. von Neumann in 1946.
43. M. Campbell-Kelly, "Programming the EDSAC: Early Programming Activity at the University of Cambridge," *Annals of the History of Computing*, vol. 2, no. 1, 1980, pp. 7–36, 17. Campbell-Kelly attributes the terminology used for the two types of subroutine to Douglas Hartree.
44. To give credit to Wheeler, who has been credited as the inventor of the closed subroutine, the Monte Carlo programs used a simple method to process the return address and relied on global variables as parameters and arguments. Campbell-Kelly shows that EDSAC practice soon moved beyond these particular mechanisms. Also, ENIAC's use of function table memory eliminated the possibility of automatically relocating subroutines from a library, which was a major focus of early work on subroutines both by Goldstine and von Neumann (in the final installment of the "Planning and Coding" reports cited earlier) and by the EDSAC team. The loss of this particular "first" takes little away from the substance of Wheeler's innovations.
45. The original simple sequence of operation box numbers was confused by alterations to the original diagram. Small insertions were placed in new boxes with decimal numbers, such as 20.1*. More radical changes led to new numbering sequences distinguished by overlining, or the use of the ° symbol. For the second run, the boxes were renumbered sequentially, with each functional region being allocated a block of 10 numbers. As before, though, modifications soon led to the introduction of a variety of ad hoc symbols.
46. Undated page containing handwritten flow diagram with nine boxes, JvN-LOC, box 11, folder 8.
47. Hurd, "A Note on Early Monte Carlo Computations and Scientific Meetings," p. 155.
48. For a comparison of alternative techniques of census taking, see E. Fermi and R.D. Richtmyer, "*Note on Census-Taking in Monte-Carlo Calculations*," LAMS-805, series A, Los Alamos Nat'l Lab., 11 July 1948.
49. You might wonder why ENIAC could not simply start processing the first generation of daughter neutrons immediately, rather than waiting for the card it had just written to be fed back into its input hopper. The answer is subtle. After reading a card, ENIAC could process the data several times because it was cached in the relay memory of the constant transmitter. This meant that even after the initial neutron data had been modified while tracing the progress of the first daughter neutron, it could be retrieved from the constant transmitter without needing to read the card again. No other operation, including writing a card, could modify the contents of this relay memory.
50. For example, he wrote, "Klari survived the Aberdeen expedition this time better than the last one," letter to Ulam, 18 Nov. 1948, JvN-LOC, box 7, folder 7. The term seems to have been in common use in the von Neumanns' circle. Carson Mark also referred to a series of "rather major calculation expeditions" from Los Alamos to ENIAC in his testimony during the 1971

ENIAC patent trial ("Testimony: September 8, 1971," *Honeywell vs. Sperry Rand*, vol. 48, p. 7504, ENIAC Patent Trial Collection, Univ. of Pennsylvania Archives and Records Center UPD 8.10). A meteorologist who worked with von Neumann also wrote later of "ENIAC expeditions," the first of which was a "remarkable exploit" that "continued 24 hours a day for 33 days and nights." G.W. Platzman, "The ENIAC Computations of 1950—Gateway to Numerical Weather Prediction," *Bull. Am. Meteorological Soc.*, vol. 60, no. 4, 1979, pp. 302–312, quotations pp. 303, 307.

51. Fitzpatrick, "Igniting the Light Elements," p. 268.
52. J. von Neumann to Bradbury, letter, 6 Feb. 1948. Herman H. Goldstine Papers, American Philosophical Society, Philadelphia (hereafter cited as HHG-APS) series 1, box 3.
53. J. von Neumann to Simon, letter, 5 Feb. 1948, HHG-APS, series 1, box 3. Simon to J. von Neumann, letter, 9 Feb. 1948, JvN-LOC box 12, folder 3.
54. J. von Neumann to Mark, letter, 13 Mar. 1948, JvN-LOC, box 5, folder 13.
55. "ENIAC Operations Log (After November 21, 1947)," Sperry-UNIVAC Company Records, Hagley Museum and Library.
56. The use of certain accumulators for the temporary storage of variables, the usage of the various digits of the random number ξ, the layout of the numeric function table and the constant transmitter registers, and a few numeric constants are listed on four undated manuscript pages on squared note paper in JvN-LOC box 11, folder 8. The punched card layout is described on the December 1947 flow diagram.
57. Richtmyer, "Monte Carlo Methods," p. 4.
58. "Receipt of Classified Materials," 16 Jan. 1948, JvN-LOC, box 19, folder 7. In case anyone concerned with national security is reading, we should make it clear that the accessible archival documents do not include these physical constants.
59. "ENIAC Operations Log," entries for 1 and 2 Apr. 1948.
60. J. von Neumann to Ulam, letter, 11 May 1948, SMU-APS, series 1, John von Neumann folder 2.
61. J. von Neumann to Ulam, letter, 14 May 1948, SMU-APS, series 1, John von Neumann folder 2.
62. JvN-LOC, box 12, folder 6, contains a 17-page full manuscript of "Actual Technique" and a typewritten transcription of the manuscript with insertions and corrections by John von Neumann numbered $\times 1$ to $\times 83$ noted in the right margin. Eight larger passages of handwritten text on separate sheets are marked for insertion at various points. A five-page manuscript with the same title is in fact an incomplete later draft. This document evolved into the "Actual Running of the Monte Carlo Problems on the Eniac" discussed later.
63. J. Von Neumann to Ulam, letter, 4 Nov. 1948, SMU-APS, series 1, J. von Neumann folder 2.
64. Fitzpatrick, "Igniting the Light Elements," pp. 269.
65. Three drafts of the "Actual Running…" report are held in JvN-LOC, box 12, folder 6. These are a manuscript in the hand of Klara von Neumann and two typed versions of the same text. One typescript has been annotated and corrected all the way through, primarily by Klara von Neumann. Metropolis later wrote to Klara, "Here is your manuscript together with a rough typewritten copy … The flow diagrams will definitely be finished on Monday and will be sent to you on that day." (N. Metropolis to K. von Neumann, letter, 23 Sept. 1949. JvN-LOC, box 19, folder 7.) A transcription of the report with all marked corrections made is available from our project website, www.EniacInAction.com.
66. K. Von Neumann, "Actual Running …" (typescript version), JvN-LOC, pp. 5–6.
67. We are not sure how Klara von Neumann and Metropolis handled the exponential growth in punched cards that would take place during the course of a calculation. One suspects that either a very small number of census times were used or that some ad hoc manual adjustment took place when the deck became unmanageable.
68. It is not entirely clear to us why ENIAC still stopped following the course of a neutron when a census time was reached, resuming only when the card it had just punched was read back in. It could easily have been programmed to output a census card, for analytical purposes, and then to continue immediately by determining the fate of the neutron during the next census period. Still, this would have eliminated the possibility of doubling the neutron's "weight" between census periods, as described above, and unless the output deck was sorted before being reintroduced to ENIAC for the processing of fission cards, the census card would still have been read (and immediately ignored) at some future point.
69. J. von Neumann to Ulam, letter, 18 Nov. 1948, JvN-LOC, box 7, folder 7.
70. These documents are all found in JvN-LOC, box 11, folders 7 and 8. The program code has a title page reading "Card Diagram//FLOW DIAGRAM//Coding/Function Table III Values//Monte Carlo//Second Run." Of these, only the Coding section remains. A note added by J. von

Neumann reads, "Will be needed in LA in early January, but should then come to Princeton for reporting, etc. JvN." The earlier draft flow diagram is a little messy and can be distinguished from others in that folder by its lack of numbering. The later version is a mirror image negative that has corrupted somewhat over the years and is hard to read without image processing. Our project website, www.EniacInAction.com holds copies of the program code manuscript, a spreadsheet file holding an annotated transcription, a flow diagram created from the code, and technical report by M. Priestley and T. Haigh, "Monte Carlo Second Run Code Reconstruction and Analysis."

71. Some instructions included addresses or data as well as a two-digit operation code, but most did not. The code in the second run program used approximately 2.5 digits per instruction.
72. Teller's campaign for hydride weapons is discussed in G. Herken, *Brotherhood of the Bomb: The Tangled Lives and Loyalties of Robert Oppenheimer, Ernest Lawrence, and Edward Teller*, Henry Holt and Co., 2003.
73. J. von Neumann to K. von Neumann, letter, 7 Dec. 1948. Personal collection of Marina von Neumann Whitman, copy provided by George Dyson. This collection is cited hereafter as KvN-MvNW.
74. J. von Neumann to K. von Neumann, letter, 13 Dec. 1948, KvN-MvNW.
75. Ulam to J. von Neumann, letter, 7 Feb. 1949, JvN-LOC, box 7, folder 7.
76. LAMS-868, "Progress Report T Division: 20 January 1949–20 February 1949", March 16, 1949, quoted in Fitzpatrick, "Igniting the Light Elements," p. 269. The original report remains classified.
77. One exception is C. Rope, "ENIAC as a Stored-Program Computer: A New Look at the Old Records," *IEEE Annals of the History of Computing*, vol. 29, no. 4, 2007, pp. 82–87.
78. Galison, "Computer Simulation and the Trading Zone," p. 120.
79. As reconstructed for the 50th anniversary celebration, the Baby's first program consisted of 19 instruction lines, read no input (understandable as switches were the only input device), and ran for 52 minutes with the intention of giving the hardware, in particular the novel memory unit, a thorough workout. See www.computer50.org/mark1/firstprog.html. The programs run at the EDSAC's inaugural demonstration on 22 June 1949, which printed tables of squares and prime numbers, were longer, consisting of 92 and 76 instructions, respectively, much of which was code to print the results in an attractive format. W. Renwick "The E.D.S.A.C. Demonstration," *The Early British Computer Conferences*, M.R. Williams and M. Campbell-Kelly, eds., MIT Press, 1989, pp. 21–26.

Thomas Haigh is an associate professor of information studies at the University of Wisconsin–Milwaukee. His research interests include the history of computing, especially from the viewpoints of labor history, history of technology, and business history. Haigh has a PhD in the history and sociology of science from the University of Pennsylvania. See more at www.tomandmaria.com/tom. Contact him at thaigh@computer.org.

Mark Priestley is an independent researcher into the history and philosophy of computing, with a special interest in the development of programming. He started his career as a programmer and was for many years a lecturer in software engineering at the University of Westminster before turning to the history of computing. Priestley has a PhD in science and technology studies from University College London. His most recent book, *A Science of Operations: Machines, Logic, and the Invention of Programming* (Springer, 2011), explores the coevolution of programming methods and machine architecture. More information is available at www.markpriestley.net. Contact him at m.priestley@gmail.com.

Crispin Rope has been interested in ENIAC since reading Douglas Hartree's pamphlet on the machine from 1947 and has pursued a vocational interest in its history for more than a decade. His earlier work on this topic has been published in *IEEE Annals of the History of Computing* and *Resurrection: The Bulletin of the Computer Conservation Society*. Contact him at westerfield@btconnect.com.