Calculating Machines and numerical tables – a reciprocal history

Liesbeth de Mol, Marie-José Durand-Richard

Liesbeth De Mol[*], Marie-José Durand-Richard[†]

Calculating Machines and numerical tables – a reciprocal history.

Springer

[*] CNRS – UMR 8163 STL, Lille 3, e-mail: liesbeth.demol@univ-lille3.fr
[†] SPHERE, Université Paris Diderot – Paris 7, France, e-mail: mjdurand.richard@gmail.com

Contents

Abstract

The aim of this chapter is to study the history of the design and use of calculating machines to produce and use numerical tables, from the time of Babbage's plans for difference and analytical engines to the modern digital computer, including in this analysis the role played by analog machines between the two. We will examine how the place and role of numerical tables changed throughout the different stages of this mechanization. It is our hypothesis that these processes of mechanization and, ultimately, automisation and digitalization, result in the progressive internalization of aspects of the dynamical relation between humans and tables into the machine. This chapter considers several major stages in the devising of machines for the making and use of tables, from difference engines to computers. Following these stages, we investigate how increasing demands for calculating power go hand in hand with advances in the new technological possibilities offered by machines. The way by

which machines were introduced and accepted for mathematical computations will also be examined. Even though this chapter is thus structured according to the changes in the machines, the focus will be on how these changes have affected numerical tables. By doing so, it will become clear how it is not only the machine which shapes the history of the numerical table, but, also how the numerical table has impacted on the design and use of calculating machinery. In fact, as we will argue, the history of calculating machinery is reciprocal to the history of the numerical table. With the development of modern computers, in which data and programs can be manipulated in the same way, this reciprocity is expressed by an increasing interchangeability between tables and algorithms.

0.1 Introduction: issues raised by the mechanization of tables

The aim of this chapter is to investigate the history of the relation between numerical tables and machines. We consider this relation to be a reciprocal one: advances in table-making or changing needs motivated the construction or adaptation of machines. Conversely, advances in machine design and use impacted on the history of the numerical table. In this reciprocal history, mechanization and, ultimately, automization and digitalization of the whole process of making tables results in the progressive internalization of aspects of the making and use of the numerical table into the machine and, as a consequence, in a different type of numerical tables, viz. internalized tables.

The first significant attempt at mechanizing table making is largely due to Charles Babbage in the first hallf of the 19th century. His designs for a difference engine were picked up in late 19th century resulting in the construction of other different engines. These machines are often understood as more isolated cases. However, we will show that there is an important continutity in the history of the mechanization of numerical tables which places these apparently isolated attempts in a broader history. This will be done by including in our analysis also a dissussion of analog machines. Such continutity however can only be observed if one enlarges our view on the history of mathematics, one which does not fit into a view of science as being universal and independent of historical contingencies but one that assumes that such history is embedded in a society which includes different practices (Bonneuil and Joly, 2013). More particularly, our approach here is supported by Bertrand Gille's analysis of history of technology, asserting that technical devices do not exist by themselves, but can only become effective devices if they are part of a new technical system (Gille, 1978, pp. 3-78). As such, our study will be structured by means of technological advances that belong to different social constellations and which help to understand not only why the earlier attempts at buidling difference engines did not immediately result in a larger-scale project of mechanizing tables, but, in general, how the mechanization of tables is a history of increased collaborations accross disciplines.

As we will see, even if Babbage's engines really correspond to his ambition of a general method for mathematical analysis, several changes had to occur before mathematical machines could be fully incorporated into the process of table computations. The real expansion of mathematical machines required that scientific rationality crossed technical rationality. Fundemental issues had to be explicitely raised and addressed by "learned men *and* "practical men , so that they could think and work together. So, new configurations came about, which realized a new balance between cultural, social and economical factors. During the second part of the 19th century, analog computers succeeded to create such a new equilibrium between mathematicians, physicists and engineers, providing results that could not be attained by a purely theoretical mathematics, but with an accuracy which did not meet the requirements of all. It will be only with computers that this new equilibrium will concern the whole field of computations, and then the making of tables will become a by-product of their work. So, to understand why such machines succeeded or not to be built and used, we have to investigate the gap which was existing between the ambitions of their authors and the effective needs and means of the "milieux" where numerical tables were produced.

0.2 Mathematical machines

During the second half of the 19th century, difference engines were specially engineered and used, so as to draw up numerical tables, successively by Georg Scheutz (1795–1873) and his son Edward (1822–1881), and then Martin Wiberg (1826–1905) in Sweden, George B. Grant (1849–1917) in the United States, and Christel Hamann (1870–1948) in Germany. All of them were special-purpose machines, and single models. However, these single-model engines have their roots in Babbage's difference engines. Indeed, in the first half of the 19th century,

the mathematician Charles Babbage (1791–1871) designed several models of such engines, which were only partially built, together with an impressive plan for an "analytical engine". In their whole, Babbage's engines were really conceived as mathematical machines, and at the core of an algorithmic view of algebraical analysis: the difference engines could give tables of functions by the special method of differences while the "analytical engine" generalized it and could finally compute the values of any function from its Taylor's expansion.

0.2.1 The division of labour and the mechanization of tables

History of computing rightly praises Charles Babbage (1791–1871) for his first really determining advances in calculating machines designs and achievements. Indeed, since 1822 and all along his life, Babbage was more and more involved in his general program of mechanisation of algebraical analysis. His inventive inclination outweighed his disappointments with craftsmen and government fundings when he forsook the making of the difference engine for the plans of the analytical one in 1834. And he came back to a new difference engine in 1855, where he introduced some improvements initiated for the analytical engine (Hyman, 1983). But what were the underlying motivations which induced Babbage to turn from his important researches on algebraical analysis to a lifetime devoted to the design of machines? In order to tackle this question, it is important to consider how Babbage's engines realized a fascinating association between his theoretical views as one of the founding members of the English algebraical network and his own major interest for the manufactures organization (Durand-Richard, 2011).

Babbage's own concerns can easily be apprehended in his book *On the Economy of Machines and Manufactures* (1832). Two of its chapters were devoted to Adam Smith's principle of division of labour (Smith, 1776), which he examined both for manual and mental labour in a very innovative way, illustrated by the difference engine (Babbage, 1832, Chaps. XIX-XX). His motivation for devising a difference engine was related to the necessity of clearing logarithmic and trigonometrical tables from any error for astronomy and navigation. It was directly rooted in his involvment with the *Astronomical Society*, which he just created in 1820 with John F. W. Herschel (1792-1871) and some friends around a core of "business astronomers" (Ashworth, 1821) such as Henry T. Colebrooke (1765–1827) and Francis Baily (1774–1844), who worked initially as actuaries, and brought their computational methods for astronomical computations. First of all, the inaugural address of this society (Anonymous, 1821) was a real manifesto to structure the whole observational and computational activity of astronomers according to the principle of division of labour. This principle had previously been applied in France when Gaspard Riche de Prony (1755-1839), then director of the *Ecole des Ponts et Chaussées*, was commissioned for supervising the making of new trigonometric and logarithmic tables, because of the recent adoption of the metric system. Babbage was given some pages of the Sines tables by the printer Didot when he visited France with Herschel in 1820. Even if Ivor Grattan-Guinness gave great details on the relationships between De Prony and Babbage enterprises (Grattan-Guinness, 2003), it must be resumed De Prony organization of the 'manufacture" of tables, as recounted by Babbage on several occasions (**?**):

> The first was composed of four or five geometricians[1] of very high merit; this group occupied itself purely with the analytical part of the work, and the calculation of some fundamental numbers [for the tables].
> The second group contained seven or eight calculators, who possessed a knowledge of analysis, and had considerable experience in converting formulae into numbers: their duty was to deduce from the first group's general calculations numbers serving as starting points, with which to form the top most rows of each sheet in the grand folio volumes, on each of which had been drawn one hundred lines; the ninety-nine remaining lines were to be filled in by the workers of the third group.
> This third group comprised no less than seventy or eighty individuals; but it was the easiest to form, because, .. the one essential condition, for their admission [to the group], was for them to know the first two rules of arithmetic.

Babbage's difference engine mechanised the work of the third section of De Pronys computational organization, the most elementary one, for which only additions and substractions were required. The machine consisted of columns bearing the values of the function and of its successive differences. Each column was formed by a stack of coaxial rotating disks, each of which displaying a digit for a number. Gears a the top of these columns operated additions from the value on one column to the value of the next one. Only the first values of the function and of its successive differences were first introduced as data, and the machine internalized the addition process, together with the transmission of numbers from its operating to its printing sections.

In another paper, Ivor Grattan-Guinness also insisted on another main characteristic of Babbage's work, related to his twofold cultivation of mathematics and machines (Grattan-Guinness, 1992). Babbage proceeded as an "algorithmic thinker", so that operations could be carried out as well on symbols of functions. And he extended this algorithmic way of thinking in his various papers since his first major contributions on the Calculus

[1] The mathematician Adrien-Marie Lefendre (1752-1833) was one of them.

Annexe 1
Machine aux différences n°1
aujourd'hui exposée au *Science Museum* de Londres

Page de garde de l'autobiographie de Babbage
***Passages of the Life of a Philosopher*, 1864**

Reproduit de *Mathématiques, Informatique et Sciences Humaines*, n° 120, 1992, p. 79-82.

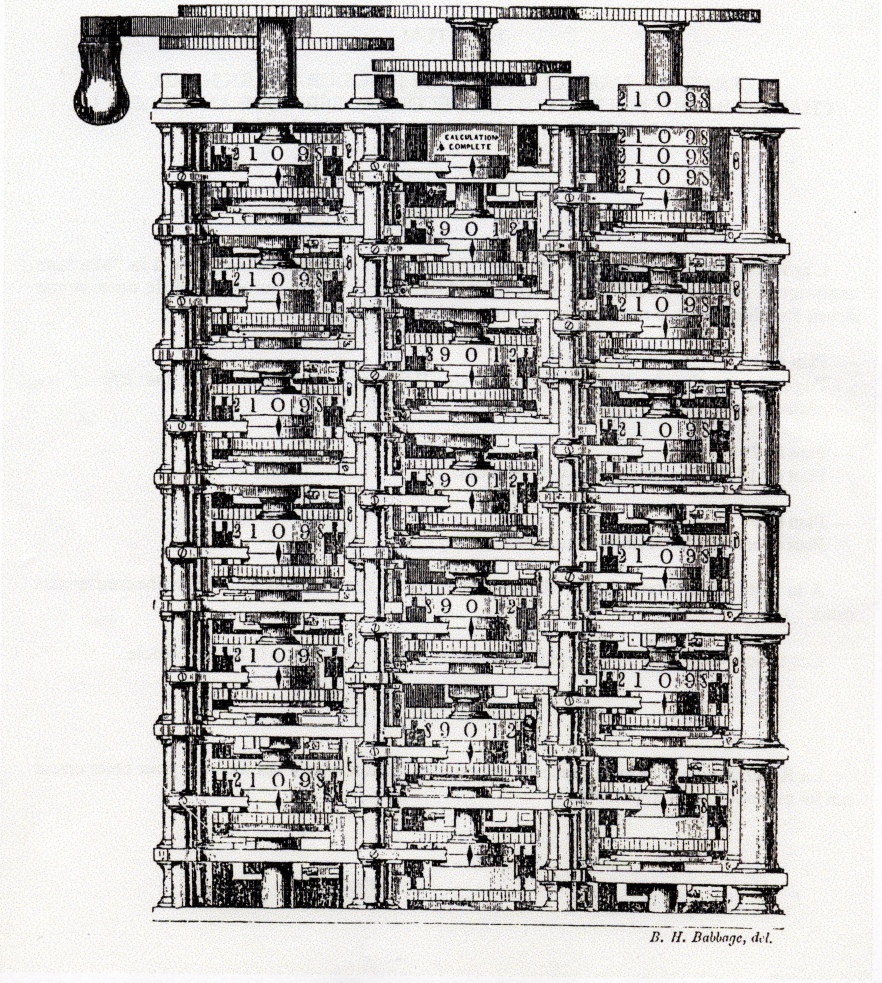

Fig. 0.1 The assembled portion of Babbage's difference engine n°1

of Functions (Babbage, 1815) (Babbage, 1816). It was directly related with the theoretical view of the English algebraic network.

0.2.2 Tables in the algorithmic view of algebra

In fact, at the beginning of the 19th century, algebra did not consist only on the research of a general theory of equations. It was essentially conceived as an analytical tool for solving problems otherwise than by geometry. And for solving equations or for giving functions values, algebra also needed numerical means, for which series and tables were extensively needed, both to deal with more complicated functions than the elementary ones, and to obtain approximate values when exact formulas were unknown or too difficult to be worked up easily. This state of algebra, or rather of algebraic analysis, was well illustrated by the preface of the single volume

of *Memoirs of the Analytical Society*, published in 1813. Babbage wrote this preface, mainly helped by John F. W. Herschel (1792–1871) with remarks by George Peacock (1701–1858). Actually, it exhibited an impressive research program, intended to "reimport" in Great-Britain what was developed on the Continent all along the 18th century, essentially about infinitesimal calculus and its differential notation. And in its presentation, This preface presented Lagrange's algebraical approach of this calculus, and combined with approximations technics developed inland since Newton and his followers.

This program highly focused on infinite series, which may define all sorts of new functions, and gave approximations of their numerical values. Series were claimed as the central unifying element of the analytical science. First of all, although a general method for solving equations was missing, "the inverse function of any expression, such as $a + bx + cx^2 + dx^3 + \ldots$. may readily be exhibited in an infinite series" (Babbage, 1813, pp. iii). The theory of the expansion of functions in Taylor series was asserted as the most prominent feature of the differential calculus (Babbage, 1813, pp. iv). And finally, "the development of functions has lately been made, under the name of "Calcul des Fonctions génératrices", the foundation of a most elegant theory of finite differences" (pp. iv Babbage, 1813, p. v). Mainly, the development of functions was crucialfor solving differential equations, when no known function was solution.

After this emphasis given to series, tables were immediately referred to in this preface, making clear that Babbage's interest for tables was already much larger than for the sole logarithmic and trigonometrical functions. As he said, with "the invention of integral calculus [tables] received a wast addition to their utility". As few integrals are already known, Babbage planned first to classify known integrals and to reduce each new one to a classified form, and then "When this is accomplished, all that remains for the perfection of this branch of Analysis is to calculate tables which shall afford a value of the integral for any value of the variable" (pp. iv Babbage, 1813, p. vi).

Further, Babbage will insist on the general use of tables, particularly for transcendants, for which trigonometrical functions are only a special case. As regards elliptic integrals for example, Legendre's classification was known in Great-Britain, at least since the first part of the translation by William Wallace (1768-1843) in 1809 of his "Mémoire sur les transcendantes elliptiques" (1796). Babbage commented the *Exercices de calcul intégral* (1825) of Adrien-Marie Legendre (1752–1833) but, it was before Legendre published his complete tables in his *Traité des fonctions elliptiques* (1825), and Babbage insisted on the need of tables for these functions, and on the best way to obtain them (Babbage, 1813, pp. iv-v):

> It was next proposed to admit as known transcendants, all integrals which. could be reduced to the rectification of the conic sections. But, besides the preposterous idea of limiting an Analytical expression by the properties of a curve, no tables had been constructed for them, and of course, the determination of their arcs could only be performed by the actual calculation of the integral under consideration: nor, indeed,would it have been possible to form useful tables of any moderate length, without first discussing the properties of the transcendants themselves in the fullest manner.

For the next important class, the logarithmic transcendants, Babbage reported on the works of Legendre and John Landen (1719–1790). He deplored that the former did not know the important works of Willaim Spence (1777–1815) on the subject – which he attributed to the interrupted intercourse through the Channel because of the war. Spence's tables for logarithmic transcendants followed the table calculated by Christian Kramp (1760–1826) in his *Analyse des réfractions astronomiques et terrestres* (1799) for the integral $\int e^{-x^n}$, for the case $n = 2$.

Thus tables, as viewed by Babbage, are really a general tool, the numerical translation of series as an algorithmic approach of algebraical analysis. As such, as we will see later 0.3, it can be easily understood that Babbage's difference engine already contained the seeds for the idea of building a "universal" computing machine", viz. the analytical engine. In his machines, this generalizing and algorithmic view on tables and mathematics in general, was combined with an emphasis on the division of labor previously referred to and with the need to relieve the Analyst of the algorithmic work of table computations. So, while Babbage praised the practical interest of tables, he immediately underlined how boring was the task of setting tables:(Babbage, 1813, p. viii)

> The ingenious Analyst who has investigated the properties of some curious function, can feel little complaisance in calculating a table of its numerical values; nor is it for the interest of science, that he could himself be thus employed, though perfectly familiar with the method of operating on symbols; he may not perform extensive arithmetical operations with equal facility and accuracy; and even should this not be the case, his labours will at all events meet with little remuneration.

This remark already points out the nascent interest of Babbage for the mechanisation of tables, and his elitist way of thinking mathematics, separating theoretical and numerical work. All along his later work on mechanisation of tables, he will insist on the importance for the analyst to be relieved of this work.

0.2.3 The pivotal interest for the method of differences

So, Babbage and the whole *Analytical Society* – even with its few members –nurtured a collective and intellectually ambitious project of unifying analysis. And the theory of finite differences was an essential part of the practical investigation of this project, especially since the Lagrange's theorem established a theoretical link between finite and infinitesimal calculus Durand-Richard (2012).

Indeed, Lagrange established a symbolical expression of Taylor's theorem in 1772 from the analogy he noticed between Taylor's formula and the expansion of e^x in series. It stood as :

$$\Delta u = e^{h\frac{du}{dx}} - 1$$

$$\Delta^n u = \left(e^{h\frac{d}{dx}} - 1\right)^n$$

Laplace improved its demonstration in 1776, and 1800. The same year, Louis–Franois Arbogast (1759-1803) separating the symbol of operation from the symbol of quantity, gave a new symbolical expression of Lagrange's theorem:

$$\Delta = e^{h\frac{d}{dx}} - 1$$

$$\Delta^n = \left(e^{h\frac{du}{dx}} - 1\right)^n$$

which opened the way to a new form of calculus on operators, even if it seemed meaningless (Durand-Richard, 2000, pp. 152–154).

As soon as 1813, in the *Memors of the Aanalytical Society*, Herschel wrote two papers essentially concerned with equations of finite differences, whose integration founded a general method initiated by Laplace for solving functional equations (Herschel, 1813b,a). In these papers, as Babbage did in the preface, Herschel paid a special attention to the notation, first to Laplace notation f for a function, named the "characteristic", and mainly to Arbogast's notation, with the symbol $D = \frac{d}{dx}$ he considered as a "symbol for an operation". This separation between symbols of operations and symbols of quantities will be at the core of Peacock's efforts to think algebra as "the language of symbolical reasoning". It will be referred to by most of the members of this algebraists network, and will be specially materialized in Babbages devices, where numbers and operations were embodied by specific parts of the machines. In 1814, Herschel also published "Considerations on various points of analysis" in the *Philosophical Transactions*. This title repeated the one of Laplaces paper "Sur différents points danalyse", where the operating properties of the functional notation are fully carried out in exploring Laplaces theory of generating functions. Finally Herschels investigations on that subject culminated:

- firstly with an original treatise on the subject of finite differences which he substituted to Lacroixs appendice on differences and series, in the Lacroix *Elementary Treatise of the Differential and Integral Calculus*, translated by Peacock, Babbage and him in 1816 (Lacroix, 1816).

- 2dly, in 1820, with his *Collection of Examples of the Applications of the Calculus of Finite Differences*, published in Cambridge with two other volumes of examples, one by Peacock on integral calculus, and one by Babbage on functional equations, so as to help tutors in Cambridge colleges to teach analysis with the differential notation (Herschel, 1820).

So, when Babbage began to consider how to mechanise the making of tables, the method of differences was already the subject of a profound specific theoretical interest, from the Continent (Lubet, 2014) to Cambridge algebraists. Moreover, the whole work of this network on the calculus of operations can be viewed as a real will to extend Laplace's work on generating functions, as it was a very powerful tool for operations on functions. Laplace established parallel operating possibilities on the coefficients of infinite series and on their generating functions, so that this method was extensively used in interpolation processes and for giving approximate solutions of differential equations (Panteki, 1992).

0.2.4 The comprehensive plan for tables computations with the Difference Engine

The method of differences was supported by a specific property of polynomials: the n-th difference of a n-order polynomial is a constant, and its $(n+1)$-th difference vanishes. Still more important is the fact that, for any function, this property also give an approximation, the precision of which depends on the order chosen for the

difference[2]. Thereby, only from the initial values of a function and its differences, the method of differences allows to obtain successive values of a function – either exact values in the case of a polynomial, or approximate values in case of a function given by a series – just by performing additions – possibly with negative values.

For instance, from the Taylor series of of the logarithmic function, its successive differences can be written:

$$\Delta l(x) = l(x+h) - l(x) = l(1+\frac{h}{x}) = M\big(\frac{h}{x} - \frac{h^2}{2x} + \frac{h^3}{3x} - \ldots\big)$$

$$\Delta^2 l(x) = l(x+2h) - 2l(x+h) + l(x) = l(1+\frac{2h}{x}) - 2l(1+\frac{h}{x}) = -M\big(\frac{h^2}{x^2} - \frac{2h^3}{x^3} + \ldots\big)$$

$$\Delta^3 l(x) = l(x+3h) - 3l(x+2h) + 3l(x+h) - l(x) = l(1+\frac{2h}{x}) - 3l(1+\frac{2h}{x}) + 3l(1+\frac{h}{x}) = M\big(\frac{2h^3}{x^3} + \ldots\big)$$

if the values $x = 20000$ and $h = 1$ are chosen, the folowing values of the differences are obtained :

$$u = l(x) = l(10000)$$

$$\Delta l(x) = 0,000043427276863$$

$$\Delta^2 l(x) = 0,00000000434207$$

$$\Delta^3 l(x) = 0,000000000000868$$

so that $\Delta^4 u = 0$ gives a good approximation of the function $l(x)$ (cois Lacroix, 1802, p. 12). The precision of the approximation will depend on the order of the difference according to the width of the interval chosen between two successive values of the function. As Luigi Menabrea (1809-1896) and Lady Ada Lovelace (1816-1852) will later review on Babbages engines(Menabrea, 1842) (Lovelace, 1843, pp. 96–97):

> The theorem on which is based the construction of the [Difference Engine].. is a particular case of the following more general theorem: . if in any polynomial whatever the highest power of whose variable is m, this same variable be increased by equal degrees, the corresponding values of the polynomial then calculated, and the first, second, third, etc., differences of these taken the m-th differences will all be equal to each other. If from a polynomial we pass to a series having an infinite number of terms, ., it would at first appear, that in order to apply the machine to the calculation of the function represented by such a series, , if we observe that for a great number of functions the series which represent them may be rendered convergent, so that, according to the degree of approximation desired, we may limite ourselves to the calcultion of a certain number of terms of the series, neglecting the rest.

In 1823, presenting the first prototype model of the Difference Engine for an German review, the "business astronomer" F. Baily clearly stressed the unifying role of the method of differences, and the great ambition of the corresponding machine for tables, which went largely beyond their use for astronomical tables. He gave an historical account of the existing tables, but more essentially, an impressive list of the whole range of tables which could be computed with the assistance of the Difference Engine. Although he first distinguished between two classes of general tables, those consisting of natural numbers, and those consisting of logarithms, he pursued without maintaining this distinction. Thus, tables referred to by Baily included both products of numbers, square numbers, cube numbers, the higher powers of numbers, the square roots and cube roots of numbers, the reciprocals of numbers, as natural sines, cosines, tangents, etc., the logarithms of numbers, logarithmic sines, cosines, tangents, and cotangents, hyperbolic logarithms. They could also be used to find the logarithms of the sum or the difference of two quantities, from the given logarithm of each of them. Baily added also other subsidiary tables, such as, giving the powers of $0,01$; $0,02$; $0,03$; etc., the squares of the natural sines, cosines, tangents, etc., figurate and polygonal numbers, lengths of circular arcs, as tables for determining the irreductible case of cubic equations, and tables of hyperbolic functions: sines, cosines, etc. and logarithmic hyperbolic sines, cosines, etc. So;,it is clear that the uniform means of computation offered by the Difference Engine contributed to erase the distinction for instance between rational and irrational numbers, and more generally between finite and infinite expansions of functions.

Then, Baily presented more complicated computations where tables were involved, the main ones being for astronomy and navigation. For the use of the seaman in navigation, the general tables of the sun and moon had to be determined , and also, the various equations for determining their apparent places, for every day of the year, and even for different hours in the same day. Were also needed : tables giving the places of some stars, which depend on precession, aberration and nutation; tables for the lunar distances at every third hour in the

[2]This property was inductively known, and extensively used a long time before Karl Weierstrass (1825–1897) gave its demonstration in 1865.

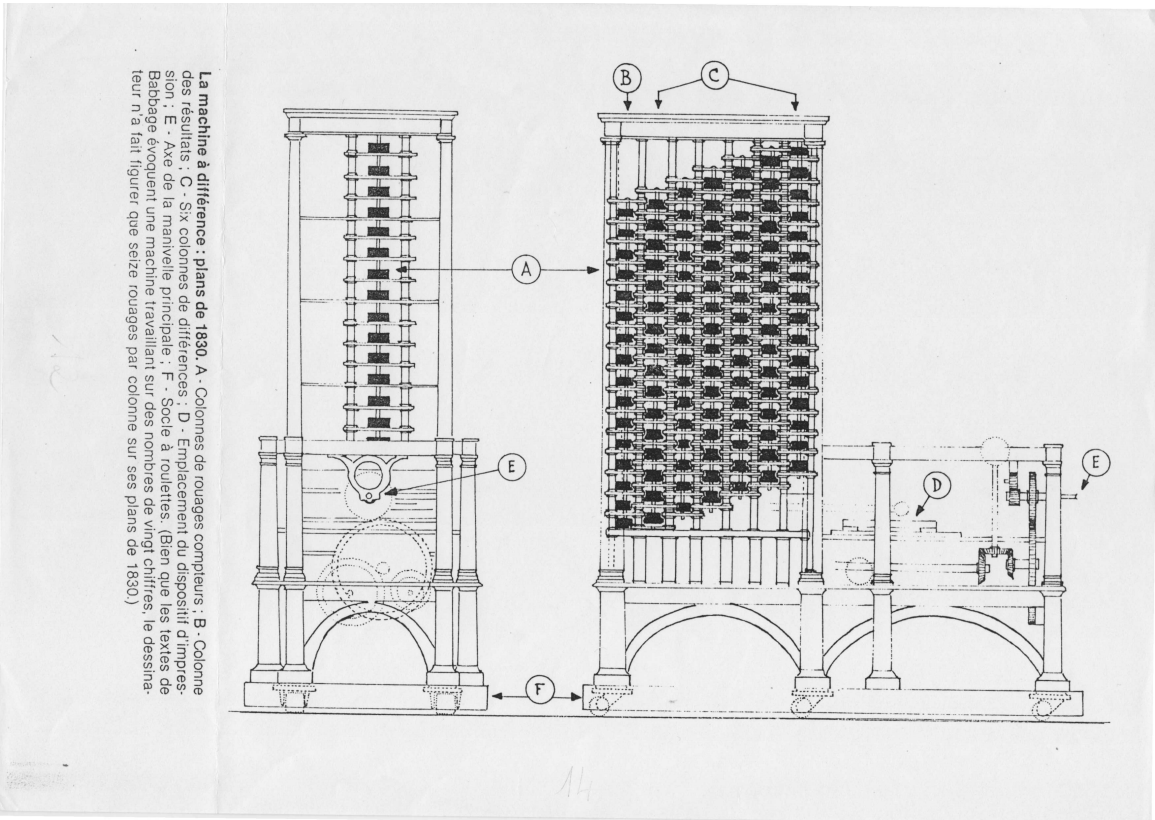

Fig. 0.2 Babbage's plan for this difference engine

day, which depend likewise of other complicated tables, and then, the Requisite Tables, published by the *Board of Longitudes*, and the usual logarithmic tables. Baily deplored the delay in the calculations of new tables, in spite of their importance, because of the huge mental and manual labour they required. He also deplored the numerous errors which were notified in the existing ones. Tables for the new planets were missing. And the recent improvements in physical astronomy, together with the frequent change of the place of planets and even fixed stars made necessary to launch a new program of computing tables. As Colebrooke already alarmed in 1822, and as Babbage repeated in 1823 (Colebrooke, 1825 (1822, p. 59) (Babbage, 1823, p. 43):

> A time will arrive, when the accumulating labour which arises from the arithmetical applications of mathematical formulae, acting as a constantly retarding force, shall ultimately impede the useful progress of the science, unless this or some equivalent method is devised for relieving it from the overwhelming incumbrance of numerical detail.

Particularly, Baily assessed that the Difference Engine was particularly suitable for the repetitive computationns required for determining the equations of mean motions, with their corrections depending on the sine or cosine of a given arc. In fact(Baily, 1823, p. 54):

> Astronomical tables of every kind are reducible to the same general mode of computation: viz. by the continual addition of certain constant quantities, whereby the mean motion of the body may be determined ad infinitum; and by the numerical computation of certain circular functions for the correction of the same. The quantities depending on these circular functions, let them arise from whatever source they may with equal ease, expedition, and accuracy by the help of the machine. So that in fact, there is no limit to the application of it, in the computation of astronomical tables of every kind.

The Difference Engine could also contribute to the computations for interests and annuities tables. And we have to remember that Babbage was involved for a short time in the creation of the Protector Life Assurance Company in 1824, for which he computed tables.

Actually, the creation of the *Astronomical Society* was part of a whole movement of criticism against the main traditional institutions in charge of astronomy. The *Royal Society* was considered as focusing too much on observations and neglecting mathematical theorizing. And the *Board of Longitudes* underwent severe attacks from the same "business astronomers" because of numerous errors in the tables of the *Nautical Almanach* (Ashworth, 1821). Clearly, as analytical computations were promoted in Great-Britain, the Difference Engine

appeared as an alternative to renew completely the methods of computations of the *Board of Longitudes*, before the creation of the *Nautical Almanach Office* in 1831(Baily, 1823). So, the project of a Great Difference Engine, as the one which was planned by Babbage int he 1820s 0.2.6, then improved in the 1850s, and finally built at the *London Science Museum* in the 1990s from the last Babbages drawings, was planned to play a major role for the whole work of computation in various activities which were then still scattered. The proponents of the Difference Engine also expressed the very large unifying ambition of the "network of Cambridge" (**?**)to reconcile learned men and practical men, later taken up by the *British Association for the Advancement of Science* (BAAS) (Thackray, 1981). In this trend of thought, Babbage's engines materialized the very large hopes of a general plan for the reorganization of computing, supported by tables and the theory of expansion of functions in series. But in this period, making tables was an handwork activity: it had to become a manufactured one. What were at stake were the conditions of this change from the one to the other. If Babbage's engines would have been achieved, the unifying project of the *Analytical society* would have found materialization, and tables would have been carried out so that computations would have been generalized and unified through its theoretical thought and its concrete results obtained by machines. But such an achievement needed profound transformations in the relationships between science and industry, which Babbage called for, but which could not occurred from a government planning, as he praised it.

But if these young reformers clearly thought the epistemic transformations of computation methods, they also obviously failed to think the material and human conditions of these transformations for mechanising computational methods. Babbage did not even consider the new possible synergies between scientists, human computers, and industrial world when he persisted in building his machines with craftmen in his personal workshop (Hyman, 1983). In fact, he could not really consider them as he maintained a strict hierarchical view of the relationships between these worlds, what can be seen in his treatise *On the Economy of Machines and Manufactures* (1831) where he clearly praised the government to intervene for regulating the industrial development and to guarantee the dominion of scientists on practical men. For Babbage, a superior authority was always needed (Durand-Richard, 2011). Clearly, even a governmental planning would not have been sufficient.

0.2.5 The open way to the "analytical engine"

The Analytical Engine has been extensively studied in the literature (Dubbey, 1978) (Ligonnière, 1987) (Hyman, 1983) (Bromley, 1987) (Bromley, 2006). But the major view of all these studies was generally to examine how the mechanical structure and the working of this engine anticipated the modern computers. Our own view on history of mathematics is rather to understand the conditions which led Babbage to think this new kind of machine, and our focus in this chapter will be in its significance for tables and their mechanisation.

Clearly, the Analytical Engine could have mastered other computations than the sole addition in themethod of differences, as it ought to perform the four operations of arithmetic. Historians often related Babbage's way of thinking its structure to his improving understanding of the relations between mechanism and operations on the Difference Engine (Ligonnière, 1987) (Hyman, 1983). Adn there are evidences that Babbage's eforts on the Difference Engine prepared his reflexion on the mechanical aspects of the Analytical Engine. However, Babbage and his reviewers repeatedly insisted on the fact that the Analytical Engine was not an extension of the Difference Engine (Lovelace, 1843, pp. 94, 124-125). They asserted that its principiple was radically different, that its original design was completely independent, and was not at all related to the interruption of the work on the previous machine, even if the two events took place together in 1834. In fact, these assertions were more concerned by the mathematical aspects than by the mechanical aspects of the two kinds of machine.

Babbage presented the Analytical Engine to a group of Italian scientists when he was invited by the astronomer Giovanni Plana (1781–1864) at Turin in 1840 (Hyman, 1983, pp. 181-184). The engineer Luigi F. Meanabrea (1809–1896), later on Prime Minister of the newly unified Italy, wrote a very well informed review of the machine, in French, in the *Bibliothèque universelle de Genève* in 1842 (Menabrea, 1842). And in 1843, Lady Ada Lovelace (1815-1852) completed her translation of this review in English, with a long series of precise notes on the possibilities of the machine, where elements of programming were detailed (Lovelace, 1843). In the addition he wrote anonymously to accompany Lady Lovelace's translation, Babbage noticed the numerous attempts he made to design and draw "a method of mechanical addition possessed of the utmost simplicity", and without which, he said, "the Analytical Engine could not exist". With mechanical means for the four operations of arithemtic, he was perfectly aware of the enormous possibilities of his new machine (C. Babbage, 1843, p. 84):

> Mr Babbage [he said] discovered a principle of an entirely new order, the poser of which over the most complicated arithmetical operations seemed nearly unbounded. The invention of simpler mechanical means for executing the elementary operations of that engine, now acquired far greater importance than it had hitherto possessed.

This radical affirmation of a new and very different principle for the Analytical Engine inclines to remember that Babbage first conceived it in 1834, just as his friend Peacock, then tutor at Trinity College in Cambridge, produced his symbolical view of algebra. Indeed, Peacock published it first in his *Treatise of Algebra*, written for Cambridge's students in 1830, and presented it too in his famous *Report on the recent progress and present state of certain branches of analysis*, given at the *British Association for the Advancement of Science* for its third Cambridge meeting in 1833, and then edited apart in 1834 This symbolical view defined algebraical operations solely by their laws of combination, and separated radically the logic of operations so induced from the numerical values of the terms and results involved. In other words, the meaning of symbols – which Peacock named their interpretation – was completely independent from the logic of operations. The aim of such a conception was to found algebra as a science, so as to substitute it to Euclidean geometry in Cambridge curriculum (Durand-Richard, 1996). It was supported by Locke's philosophy of language and his considerations on the combination of ideas and the arbitrary character of signs (Durand-Richard, 1990). And on the mathematical side, it extended to any kind of symbols the program of separation of symbols of operations from symbols of quantities produced by Arbogast regarding the differential calculus (Arbogast, 1800). For Peacock, symbols of operation themselvees were considered as arbitrary, and operations were strictly and only defined by their laws of combination. And as Babbage distinguished between the work of practitionners and the work of the Analyst, Peacock distinguished between the Arithmetical Algebra, which operated on symbols but limited their use to their arithmetical meaning, and the Symbolical Algebra, which operated on symbols without considering at all any interpreation. So the logic of operations was strictly independant from any interpreation of symbols.

The Analytical Engine materialized the same kind of abstraction as Peacock conceived with his Symbolical Algebra. It was conceived to carry out all the operations of mathematical analysis. Its methods for computations were larger than the sole method of differences [3]. Lady Lovelace insisted in her Notes on this point "perhaps Therefore, ioo little kept in view" (Lovelace, 1843, pp. 116):

> The particular *numerical* data and the *numerical* results are determined by means and by portions of the mechanism which act quite independently of those that regulate the *operations*.

As Peacock previously did for Symbolical Algebra, Lady Lovelace insisted on this clear distinction between data and operations in the analytical engine. What Peacock considered as a specific way to escape 'the shifting meaning of many of the symbols used in mathematical notation" definitely forged the structure of the Analytical Engine, where the *store* received data, and the *mill* was reserved for computations [see fig]. Lady Lovelace severally repeated in various forms that "the Analytical Engines is an embodiment of the science of operations", and regularly referred to Arbogast (Lovelace, 1843, pp. 98, 117–119). So, she considered it as a general purpose machine, while the Difference Engine was "The embodiment of *one particular and very limited set of operations, [since it] can do nothing but add*"[4] (Lovelace, 1843, pp. 119–120). More generally(Lovelace, 1843, pp. 97-98):

> The first machine [...] is, so to speak, merely the expression of one particular theorem of analysis; and [....] its operations cannot be extended so as to embrace the solution of an infinity of other questions included within the domain of mathematical analysis. It was while contemplating the vast field which yet remained to be traversed, that Mr Babbage, renouncing the plan of another system of mechanism whose operations should themselves possess all the generality of algebraical notation, and which, on this account, he denominates the *Analytical Engine*.

Moreover, both the impressive proportions of the new engine and its computing possibilities signed it as"the *material and mechanical representative* of analysis [....] through the complete control which the engine gives us over the *executive manipulation* of algebraical and numerical symbols" [cite[p. 121]BAB:Lovelace. The store consisted of double columns, one of them keeping a number in memory, and the other being driven through the rack to the mill for operations. Each column consisted of forty discs[5], so that numbers with forty figures could be represented (Lovelace, 1843, p. 100) and Lady Lovelace review planned up to two hundred columns (Lovelace, 1843, p. 128). The capacity of this memory, where fractional as well as irrational numbers were represented as decimals, was quite considerable, and well adapted to answer the needs of the whole mathematical analysis.

[3] One little part of the Analytical Engine was conceived as an auxiliary center for computation, and had to carry on the method of differences [see fig, above the rack].

[4] Even if F. Baily insisted on the fact that the difference engine could give a lot of tables, the mathematical principle on which the Analytical Engine was completely different, as it could calcuate any devlopment of a function in a Taylor series, and the Analyst can decide until what precision to do it.

[5] These columns were of the same kind as those of the Difference Engine

Morevoer, columns in the store could represent either numbers, or symbolical expressions, such as polynmials, if some power of x was affected to each column by a special card. So it was regularly insisted on the fact that the Analytical Engine was able to carry out computations, not only on numerical data, but on symbolical expressions.

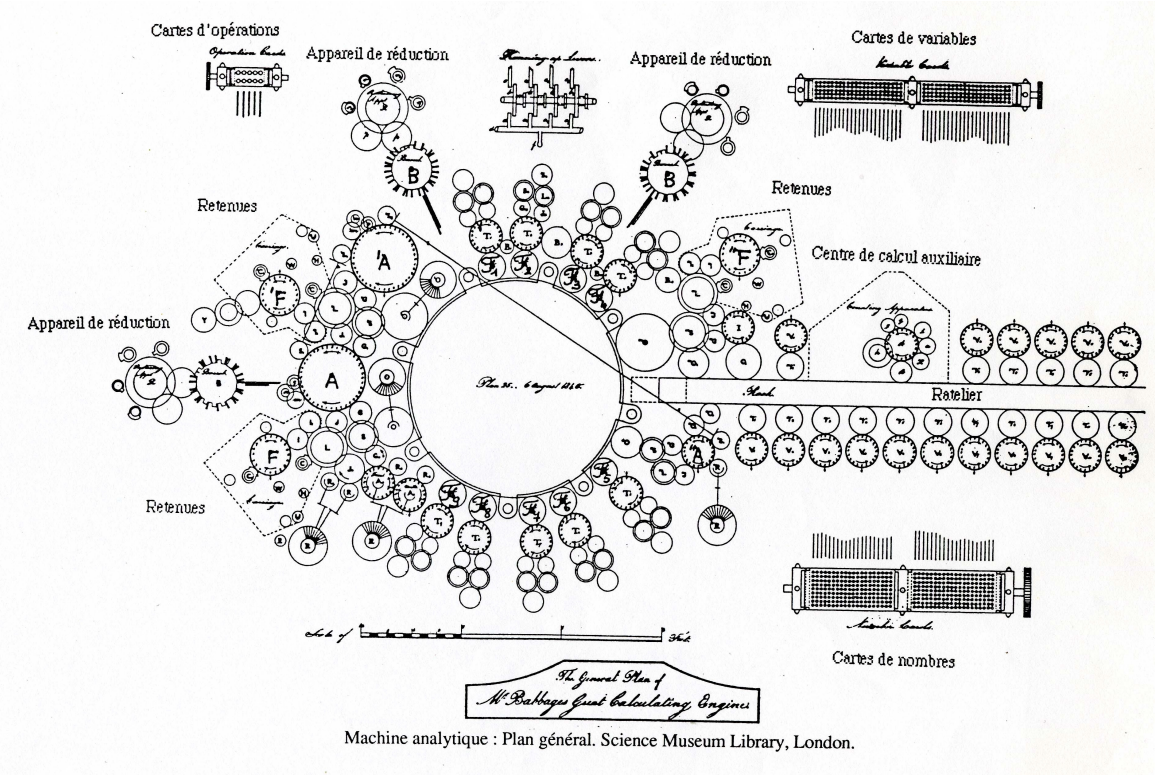

Fig. 0.3 Babbage's general plan of 1840 for his analytical engine

Everything in the machine was conceived at the same level to answer the unbounded capacity of the machine. On the mechanical side, the succession of operations was controlled essentially by three barrels also formed of several tens of stacked discs, whose each of them could be provided with studs. The lateral and rotational motion of the barrels could indicate when they were implied or not in the computations. The succession of operations was controlled by three kinds of punched cards – cards of variables, cards of numbers, and cards of operations –, the same as those of a Jacquard loom [6], from which they directly derived. These cards constituted the conceptual part of the mechanism. They were referred to as "a translation of the generality of analysis", especially as "their [was] no limit to the number of cards that [could] be used" (Lovelace, 1843, p. 112):

> if [....] every analytical calculation [is reducible] to that of the coefficients of the several terms of a series, [....], all the operations of analysis come within the domain of the engine. [...] The use of the cards offers a generality equal to that of algebraical formulae, since such a formula simply indicates the nature and order of the operations requisite for arriving at a ertain definite result,and similarly the cards merely command the engine to perform these same operations [...]. The same series of cards will serve for all questions whose sameness of nature is such as to require nothing altered excepting the numerical data. In this light the cards are merely a translation of algebraical formulae, or, to express it better, another form of analytical notation.

In this regard, what could be the role of the Analytical Engine in the mechanisation of tables ? Clearly, it was essentially conceived to work on series, and on operations on series, either numerically, or symbolically(Lovelace, 1843, p. 107):

> Generally, since every analytical expression is susceptible of being expressed in a series ordered according to certain functions of the variable, we perceive that the machine will include all analytical caclulations which can be definitively reduced to the formation of coefficients according to certain laws, and to the distribution of these with respect to the vaiables.

[6] Peacock visited Lyon in France in 1830, and Babbage too in 1840. Babbage already received as a gift a portrait of Jacquard, woven on a Jacquard loom, and bought there another one

And even if the Analytical Engine was not conceived as an extension of the Difference Engine, its use appeared yet as an extension of the previous machine, since, "while the Difference Engine can merely *tabulate*, and is incapable of developing, the Analytical Engine can *either tabulate of develop*"(Lovelace, 1843, p. 115):

> [It was] not merely adapted for tabulating the results of one particular function and of no other, but for developing and tabulating any function whatever. In fact, the engine may be described as being the material expression of any indefinite function of any degree of generality and complexity.

The new machine could not only "*reduce to tables almost all equations of finite differences*", but above all, "*the object on the Analytical Engine . [was] to convert into numbers all the formulae of analysis, and to work out the algebraical development of all formulae whose laws are known*". The Analytical Engine was conceived "for tabulating and developing any function whatever" (Lovelace, 1843, p. 115). And a special part of the machine was still reserved to work as a difference engine. But what was really new here about tables was the appearance, about some specific examples, of another kind of diagrams, describing the succession of operations in the whole running of the machine. In fact, Lady Lovelace resumed examples already given by Menabrea and detailed on more complete diagrams each step of the process, with the interactions between the results, making clear both parallel and sequential parts of the computation. In the following example, solving a system of two equations with two unknowns :

$$\begin{cases} mx + ny = d \\ m'x + n'y = d' \end{cases} \tag{0.1}$$

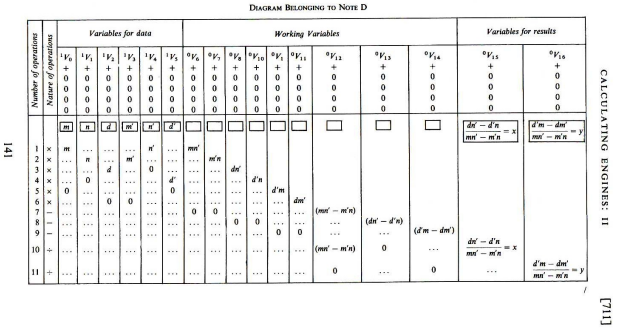

Fig. 0.4 Lady Lovelace's presentation of the computing organization for a system of two equations with two unknowns

This kind of diagram appears as a first step of algorith presented in a new kind of table 0.14. Lady Lovelace detailed it also for the computations of Bernoulli numbers from the formula which gives B_n in terms of the preceding ones, indicated what columns of data were concerned by the successive operations, what data were successively involved, and on what columns results had to be driven. It can be considered as a partial program of computation. Beyond these specific examples, Lady Lovelace developped also the notion of program about some special pieces of computations, which she named "cycles" because there appeared as repetitions of a succession of operations. These cycles needed the machine to be provided with a special mechanism, the "backing" – inspired too by Jacquard loom – in order to come back to such cycles at any step of the computation (Lovelace, 1843, p. 153).

Lady Lovelace as well as Menabrea insisted on the fact that, in spite of its sophistication, the machine was "*simply an automaton*" (Lovelace, 1843, p. 98). It was not a thinking machine, but only one which executed the mechanical part of the labour for understanding operations of matter and their laws, but not the part reserved for pure intellect. As in Peacock's work, interpretation was reserved to human beings. In that way, it was clear that the machine had no implicit knowledge, and so, that its author had to think every step and every contingency which could advent during computations, even a conditional choice (Lovelace, 1843, p. 98). Effectively, the sole limit of the Analytical Engine would have been in the writing of programs, but this could not appear since the machine was not built. Anyway, the project of the Analytical Engine had the very huge ambition to "facilitate the translation of [the] principle of the [Creator's works] into explicit practical forms" (Lovelace, 1843, p. 121). And this project could not be the project of a sole man. Its "deeper interest for the human race", was certainly a too huge one to be percevied by those who could have helped Babbage to achieve it!

0.2.6 The difficult development of Babbages prospect

Some difference engines were effectively engineered during the second half of the 19th century, whose production have already been investigated (Williams, 2003). But analyzing them one by one, just as the invention of their author, and enumerating their detailed characteristics, induces a fragmented history, which does not help to understand whether these machines were just isolated attempts to mechanise tables, or if they constituted a lineage of machines.

As it is well known, Babbages difference engine was never completely built, and his analytical engine not at all. His first little working model – the one which earned the gold medal of the *Astronomical Society* in 1822 – had two orders of differences and gave the values of a quadratic function with 6 digits values. The model Babbage began to build was never finished. Only some parts of it were assembled in 1832, exhibited at Kings College until 1842, and then given to the Science Museum. But the original plans forecasted a machine for a function with 6 orders of differences, and 16 digits numbers. The second difference engine whose Babbage drew up the plans from 1846 could work with with 6 differences, and gave 18 digits values. Both of these two last machines included a printing apparatus, which eliminated any human intervention on computation and layout of tables. However, the assembled models worked quite well and were remarkably precise, not to mention the full model produced by the Science Museum of London (1991-2002) from the latest Babbages plans [see fig. So, a machine of this size was conceived to meet the pivotal role of the difference engine in Babbages monumental program of tables mechanisation.

At least technically, Babbage's work marked a starting point of the mechanization of tables. It clearly inspired the following difference engines, even if it is difficult to see a deep continuity between them. George Scheutz (1785–1873) and his son Edward knew Dyonisius Lardners review of Babbages engine(Lardner, 1834) for the *Edinburgh Review* in 1834. Both of them were encouraged by Babbage himself, who favoured their plans and received them in London in 1854. Martin Wiberg (1826-1905) knew Scheutz success when he decided to produce a difference engine to print tables for scientific publications, and Babbage, who saw his engine in London in 1861, wrote recommendation letters for it to be approved by the French *Académie des Sciences*. In the United States, if George B. Grant (1849–1917) thought of mechanization first by himself, he knew Babbages difference engine in the 1870s, before presenting his own machine to the Centennial Exhibition at Philadelphia in 1876. Later in Germany, when the engineer Christel Hamann (1870–1948) engineered his own machine, he was already a specialist of scientific instruments, and worked at the request of the astronomers Julius Baushinger (1860–1934) and Johann Theodor Peters (1889–1941), in order to help human computations in the interpolation process by a machine with only two orders of differences. These two last machines are now unfortunately lost.

If the technical lineage of these machines is unquestionable, they were built in very different conditions from Babbages ones. Babbage was first a mathematician, carrying a large project for giving certainty to numerical calculations by machine, and for which his inventive mind produced new ideas throughout his life. He wanted to reorganize the entire production of computations by submitting it to scientific rationality, which twas materialized by his calculating machines. More widely, he was willing to convince the government and the Parliament of the necessity for the authorities to harmonize the industrial developments by submitting them to this scientific rationality. He maintained his hierarchical position, working with craftsmen in his own workshop rather than with engineers in an industrial firm. Babbages successors did not work at ali in the same state of mind. They were not scientists concerned by such a whole rationalizing scheme. They were pursuing a more limited and specific project for tables, and their much more limited ambitions could help to understand the successful manufacture of their machines, if not of their running and use. Scheutz, Wiberg, Grant and Hamann were mechanics or engineers, completely involved in their enterprise, with was a specific isolated one. Scheutz was a Swedish printer, his son was studying in a engineers school, and their second machine was used for the English Life Table by the General Register Office in London to realize a new life table. Wiberg was concerned by producing tables for scientific publications, and the interest tables computed by his machines were printed more later. Grant was only producing his Philadelphia machine. All of them were nearer the engineers "milieu"" than Babbage, who rather maintained straight relationships with the London governing class. The fact remains that these machines, even if built, were rather underemployed, either because of operating problems, or because of the difficulties of their integration of the traditional computing system.

In fact, it seems that neither Babbage nor his followers even imagine that rationalizing computation processes by machines did not only raise scientific or technical issues. As for every technical system, the reorganization Babbage planned could not be successful without developing a new consistency between epistemic, technical, but also social and political factors. Of course, Babbage worked a lot to convince political authorities of the importance of his project, and obtained, all along his working years on machines, a substantial amount of money,

even if he had to invest his own funds in the affair[7] . But Babbage considered he had nothing to wait from the industrial world. His book *On the Economy of Machines and Manufactures* was intended to think how to organize it for a better social equilibrium, Within his general view on the relationship between science and power, Babbage failed to thinkthat his own view on rationalizing computations could not be adopted without a consequent increase of computational needs, and without a social reorganization of the traditional way of dealing manually with computations. Indeed, between the first and the second half of the 19th century, the situation began to change. Needs of tables began to increase both in administrative circles with statistics, and in the industrial world where physics became involved. In fact, the convergence between the various factors previously mentioned, and consequently between mathematics, physics and industry, will be realized first by these analog machines in the second half of the 19th century.

0.3 Engineer's machines

When Babbage worked to set up digital machines supporting whole ranges of mathematical calculations, he endeavoured to embody a unitary view of science, trying to promote an specific approach of nature, experimental apprehension of which was subordinate to a theoretical structuring by the symbolical language of algebra. His 1857 paper, "On Tables of the Constants of Nature and Arts" was prepared with this outlook. It promoted a quantitative approach of natural phenomena, collecting and classifying "all the facts which can be expressed by numbers in the various sciences and arts". So, it supplied materials for his approach to be developped (Babbage, 1857).

In the second half of the 19th century, other kinds of mathematical machines were designed, and effectively built. So, it might be thought that Babbage's ambitious project was reaching completion, as numerous natural phenomena began to be apprehended by physical sciences. However, scientific context had undergone major transformations. Physicists did not work anymore in their private cabinet or workshop, but in institutional laboratories. They had received a strong mathematical training. And they met engineers and firms so as to build specific instruments or machines, where pure mathematics was insufficient to overcome phenomena understanding. Tables making mobilized both collection of data, mathematical theories and machines. So, Babbages hierarchical view of the place held by science in society was being greatly challenged.

In all cases, mathematical machines were built in other contexts than the one Babbage imagined. Moreover, their working principles differed completely from those of the difference or analytical engines. They were not concerned with arithmetical calculations, but essentially with integration issues. So, not only the algebraical calculus was completed by other methods for setting tables, but the new machines did not work on discrete values. For input as for output, they worked on graphs, either graphs of mathematical functions or graphical drawings of experimental phenomena. Lately enough in the second part of the 20th century, such machines were named analog machines, as opposed to digital computers.

0.3.1 Tide tables

At the turn of the 19th century, the *Mécanique Céleste* (1799–1823) of Pierre Simon de Laplace (1748–1827) completely renewed the researches on tides, which Laplace himself considered as "the thorniest problem of celestial mechanics". Written with differential notation, this work did not just reconcile Newton's and Leibniz's approaches of the system of the world. Regarding tides, it gave for the first time the analytical equations for the motions of the sea, which will nourish research in this realm up to nowadays. Mainly, Laplace quantified the respective influences of Moon and Sun, and he determined clearly three main species of tides – semi-diurnal, diurnal, and annual – , whose influences were differently combined according to the geographical location of ports. Understanding the whole phenomenon of tides needed to deepen Laplace's theory, so as to include new observations from one port to another, and to resolve differential equations of the sea motions, for which no general analytical solution did exist (Cartwright, 1999). While mathematical techniques of integration did not provide numerical results. harmonic analyzers will be conceived and built in the second half of the century, making possible to apply Fourier analysis, These new machines embodied recent little instruments for engineers, planimeters, in more complex mechanical systems.

n their daily practice, engineers as well as insurers, notaries and accountants, each of them for their own work, regularly needed to know how to measure the area bounded by a closed line, such as fields, forests or

[7]Even if this outlay was fairly consistent, it was funds were derisory compared to the huge amounts invested in the war effort at the time were the first computers were built

territories on a map. And even if the integral calculus was developed since the 17th century, it could not answer this question if the curve equation is not known, or even if it is known but too much difficult a function to be integrated. Nevertheless, from the beginning of the 19th century, engineers began to mechanise the integration process with small instruments named planimeters. While an operator followed the closed line by hand with a pointer, the integrator system of the planimeter totalized the successive elementary rotations of a small roller, whose each of them corresponded to the infinitesimal decomposition of the area under the curve. A lot of such instruments was realized in the whole Europe all through the 19th century and beyond (Durand-Richard, 2010).

During the 1870s, the Scottish physicist William Thomson (1824–1907) – knighted as Sir William Thomson (1867), and then as Lord Kelvin (1892) – with the help of his brother, the engineer James Thomson (1822–92), conceived a special integrator system, the disc-globe-cylinder system, by which the integral of the product of two functions could be obtained. But more essentially, they combined several of these integrators in a whole machine, the harmonic analyser, which was working with two other machines, the tide gauge and the tide predictor, for analysing and predicting tides, setting tide tables (Durand-Richard, 2014b).

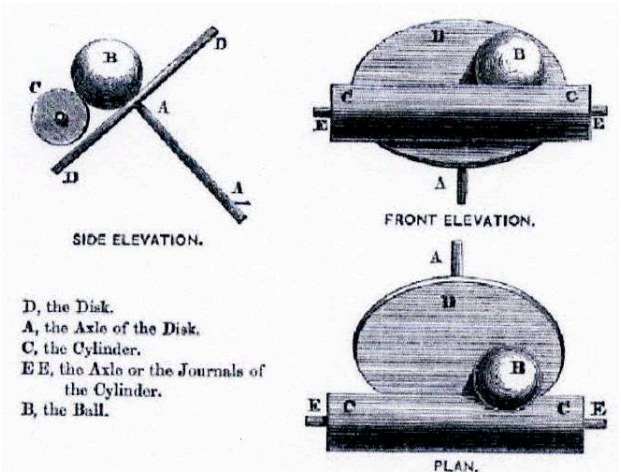

Fig. 0.5 Cutting plane of the system disc-ball-cylinder. From Kelvin 1876

The self-recording tide gauge was independantly designed in the 1830s by the British civil engineer Henry R. Palmer (1795–1844) one of the few founders of the *Institution of Civil Engineers* and by the less known French hydrographer Antoine M. Chazallon (1802–1872). Improved throughout the century, it was mainly composed of a floater resting on the water in a tide well, and a clockwork mechanism driving a rotating cylinder. Tide motion of the floater was transmitted to a marking pencil, which recorded it on a graphical paper rolled on the cylinder. Such an automatic record replaced previous isolated observations on a tide scale by an often careless staff, and eliminated errors from reading and copying related to this manual recording mode. The cylinder completed one révolution by 24 hours. And as the whole time between two high waters is not exactly twelve hours, curves for a week or a fortnight could be recorded on the same paper[8], even if Thomson admitted that "the appearance was very confusing" (Thomson, 1881, p. 28). The novelty of this continuous recording of tides was emphasized

[8]Some self-recording tide gauges even drew curves for a month, as the one at Loanda (West coast of Africa).

by engineers and hydrographers in charge of installation, supervision and improvement works of ports. They also stressed the fact that new regularities were appearing, especially for distant ports, which were dominated by other tidal regimes than that of the Atlantic zone.

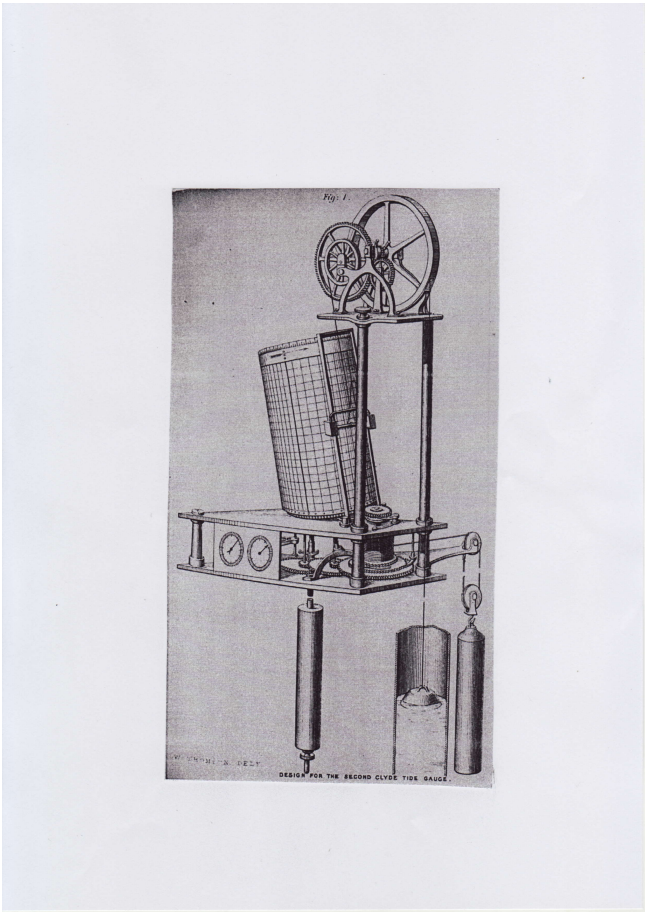

Fig. 0.6 William Thomson's Tide Gauge. 1881

From the periodic tidal curve given by the tide gauge, the harmonic analyser gave several Fourier coefficients – as much as disc-globe-sylinder systems in the machine –of the corresponding function. The curve recorded by the tide gauge was rolled on a specific cylinder of the machine, As for the planimeter, an operator had to follow the curve by hand with a pointer. But in addition, with his other hand, he had to turn a crank which simultaneously set all the integrators in motion, as they were linked together by a fork bar[9]. In the case of tides, the system disc-globe-cylinder gave the integral of the product of a function by a trigonometric function, sine or cosine. The disc was inclined at 45°, and its angle of rotation around its center was proportional to $\sin nx$ or $\cos nx$. The horizontal axis of the cylinder was parallel to the disc plane, and the globe was substituted to the roller of the planimeter. The distance from the center of the disc to the contact point between the globe and the disc is equal to the function $f(x)$ during the whole rotation of the cylinder. The rotation of the disc gave $\sin nx$ or $\cos nx$ or $\sin nx$ or $\sin nx$. The two movements are transmitted to the cylinder, so that the elementary rotation of the cylinder will be proportional to $f(x).\cos nx.dx$ and the whole rotation of the cylinder determinate the corresponding coefficient of Fouriers decomposition of the function. The first Harmonic Anlyser gave ten tide components. As W. Thomson declared in 1881: "The accuracy of the results obtained by the machine would largely depend on the precision with which the mechanism had been made", particularly for the homogeneity of the metal elements (Thomson, 1881, p. 32).

Then, the tide predictor – also named the Tide Calculating Machine – summed the Fourier components just obtained, and gave a prediction of the tides, whose accuracy depended on the number of components considered. The adopted principle was quite simple. A flexible wire went under and over a succession of coplanar pulleys,

[9] As Thomson noted, "the manipulation might appear to be very puzzling, . [but it] became easier after little practice" (Thomson, 1881, p. 28)

Fig. 0.7 William Thomson's first Harmonic Analyzer. 1876

whose each axis had a vertical harmonic movement, according to the reproduced tidal component. One of the end of the wire is fixed, and a counterweight maintained it outstretched up to the other end, whose movement gave the double of the sum of the pulleys movements. Then a pen drew the trace of the resulting movement on a graph paper winded upon a rotating cylinder. The first tide predictor included ten tide components. The period-ratios of these ten motions were introduced once for all into the machine, and they would be adjusted to suit any particular port. Then, graphical papers were given as tide tables. They were read by engineers, astronomers, and navigators. However, predicted graphical tide tables were better than discrete tables for navigators, as they gave the water heights in ports at every moment of the tide, rather than just for high and low waters.

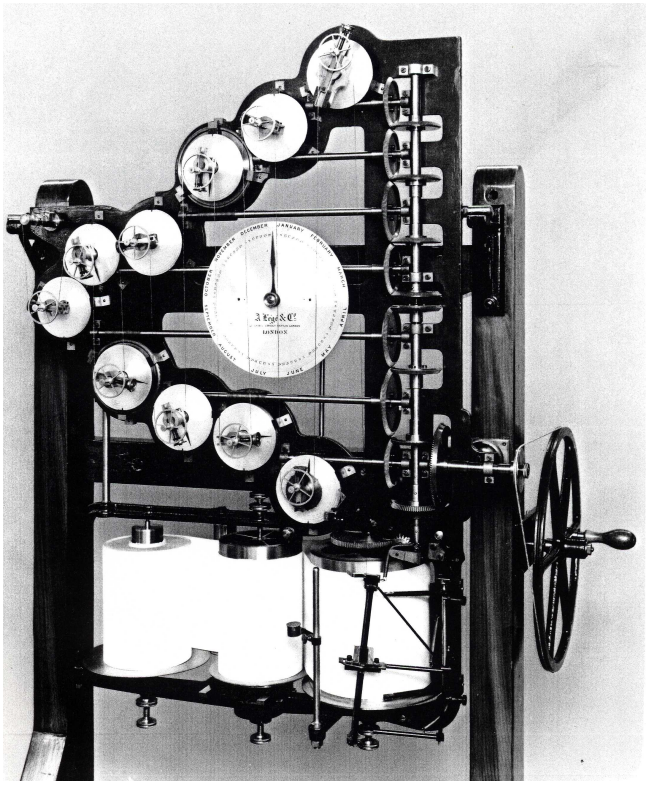

Fig. 0.8 William Thomson's first Tide Predictor

Clearly, the operating principles of thse machines greatly differed from those of Babbage's engines. All of them were analog machines. Their main characteristic was to deal with graphical representations of functions, that is with their continuous values. The accuracy of the whole process largely depended on the skill of the operator following the curve with the pointer in the harmonic analyser. But from tide gauges to the tide predictors, this

accuracy was largely based on the precision of mechanisms and graphical paper. For instance, on the two first tide predictors, the two geared wheels of the driving mechanism for each main component gave an accuracy reliable at 1/10000000 or the ratio of the speeds of the pulleys. In the tide predictor n°3, Thomson simplified the mechanism from two to one geared wheel, and the same accuracy was still 1/100000, which was considered as reliable enough for professional needs. In any case, they were efficient machines, and marvelous embodiments of the mechanical technics of the time. And the whole process was reliable enough for the needs of engineers, navigators and physicists (Thomson, 1881, pp. 11-13).

As sources are missing on ancient computation processes, it is difficult to compare (Wilkins, 2003, pp. 297-300). The operational autonomy of the harmonic analyser is the one of the clockwork mechanism driving the cylinder rotation on which the curve is rolled up. In 1879, the tide predictor could draw a tide curve for one year in four hours, from 10 components for the ports of India and Australia. Here was a particular new aspect of the machine: as it was automatic, it could be set over night, and gave the curve ready on the next morning, even if staff is needed twice in the night to raise up the counterweight. Another very important characteristics of the need and use of the harmonic analyser is that the whole process was particularly reliable for distant portsfor which hydrographers did not get enough observations on the tide heights and times for a long time. Beforehand, for the British and French ports of the Atlantic coasts, astronomers could use observations on a 19 years period a cycle between two lunar nodesfor which they could use Laplace and Lubbock computational methods.

In fact, such a program was not born in the 1870s. Tide gauges were quickly set up from the 1830s, both in British and French major ports and in their respective colonies, as well as in other European countries[10]. Discussions took place about their respective qualities, especially about their cost and about the best layout of the cylinder, horizontal or vertical (Thomson, 1881, pp. 49-56). Extensive researches were carried on by the *Royal Society* and by the *British Association for the Advancement of Science* from its foundation in 1831, so as to deepen Laplace's theory of tides both theoretically and practically. Numerous papers of John Lubbock (1803–65) on physical astronomy, and of William Whewell (1794–1866) on the propagation of tides through the oceans attested to a general concern for controlling this phenomena on the entire globe. These papers were still quoted in the discussion of tidal instruments at the *Institute of Civil Engineers* in 1881. Among the numerous committees the BAAS initiated to strengthen the relations between science and industry, a committee was funded in 1867 and "appointed for the purpose of promoting the extension, improvement and harmonic analysis of Tidal Observations". It brought together astronomers – particularly the Astronomer Royal George B. Airy (1801–92) – , mathematicians, fellows of the *Royal Society*, officers of the *Royal Navy*, and civil engineers (Durand-Richard, 2014a). This committee published numerous reports up to 1876, from which resulted the coordination between these three machines. The first garmonic analyser, "the only one hitherto made" in 1881, whose objet was "to substitute brass to brain in the great mechanical labour of calculating the elementary constituents of the whole tidal rise and fall" (Thomson, 1881, p. 9), was built in 1876 with fundings of both the *Royal Society* and the BAAS, and soon adopted by the Government of India. The machine gave two comonents respectively for the mean lunar semi-diurnal (M), for the mean solar semi-diurnal tide (S), for the luni-solar declinational diurnal tide (K_1), for the slower lunar semi-diurnal tide (O), and for the slower solar semi-diurnal tide (P). The eleventh integrator gave the mean water level (A_0). Another simpler harmonic analyser, with only six integrators, was built in 1878 by the firm Munro for the *Meteorological Office* where it harmonically analysed daily variations of temperature and barometrical pressure, as well as wind velocity, terrestrial magnetic force and electrical air potential. As for the tide predictor, after two trial models in 1872-73, the first three of them were built in five years. The first one was made in 1876 by Mr Légé & Co with ten components, and was not widely used[11]. The second one, with 20 components, was constructed for the Governement of India in 1879 also by Mr Légé & Co, and superintended by Mr Edward Roberts[12] – the calculator of the BAAS committee – for the *Nautical Almanach Office*. And Thomson designed and built a third one with 17 components[13] in 1881, for which he simplified the pulleys supports, and which was built by Mr White, the constructor of marine instruments in Glasgow.

[10]The main locations of tide gauges quoted in the discussion of tide gauges at the *Institute of Civil Engineers* in 1881 were : on the Thames, at Southampton and Bristol, as well as in several places in Belgium, especially at Ostende, Antwerp, Dendermonde. In India, tidal observations began without tide gauges in Kurrachee in 1857, but also with it at Bombay as soon as 1846, and twenty tide gauges were constructed for the Government of India in 1878-79 (Thomson, 1881, pp. 66-68). In France, Chazallon set up tide gauges in Brest, Toulon, Saint-Servan, Cherbourg, Le Havre, Rochefort et Enet, and alos at Algiers as soon as 1843 (Durand-Richard, 2014a)

[11]This tide predictor, now numbered as Tide Predictor n 1, is exhibited at the Science Museum in South Kensington, London.

[12]This tide predictor is often also numbered as Roberts tide predictor n 1, because Mr Roberts then conceived and superintended the construction of another tide predictor in 1906, The Universal Tide Predictor, with 40 components.

[13]This tide predictor was sold to the French Hydrographic Office in 1901. When this Office bought a new tide predictor to the firm Kelvin, White and Baird in 1966, the third one was deposited at the CNAM where it is still, but not exhibited.

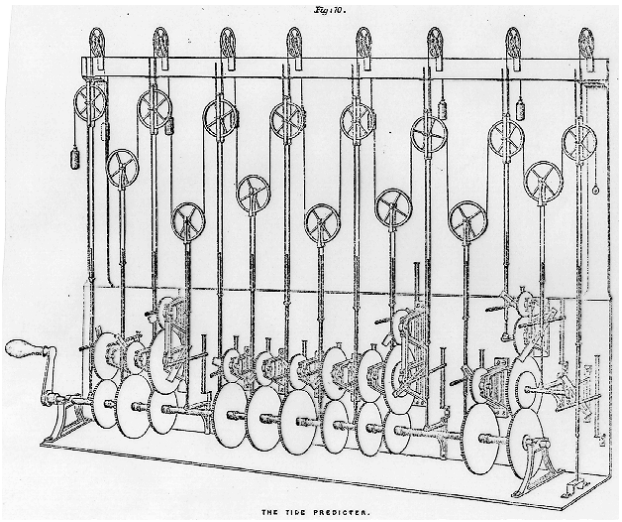

Fig. 0.9 William Thomson's Tide Predictor 1881

In other words, the context in which tables were mechanised changed completely since Babbage's attempts. Harmonic analyzers were obviously mathematical machines, as they materialized Fourier analysis. But they were conceived, built and used in a "milieu" where several professional groups were cooperating: astronomers, physicists, engineers, hydrographers and navigators [14]. Thus, in 1881, Mr Roberts challenged the way Thomson reported the invention of the tide predictor principle. And Airy, who doubted Babbages engines capacities when he was a young Astronomer Royal, still doubted Thomsons harmonic analyser to give the coefficients of tidal components for each species of tides (Thomson, 1881, pp. 39-45). While Babbage was working alone with some craftsmen in his own workshop specially built for this purpose, Thomson's machines were constructed by firms. And more essentially, these new machines showed a significant change in scale for setting tables. At the end of the 19th and at the beginning of the 20th century, these machines were still improved and mutliplied for this purpose. Other harmonic analysers were built with new types of integrators, as the Henrici-Coradi analyser which could give 50 coefficients of Fourier decomposition of a function – and even 150 if the curve was analysed three times –, and another one by Mader-Ott firm, which was largely used in research laboratories all along the 20th century (Durand-Richard, 2014b). And a long line of tide predictors, with more and more harmonic components, were produced by the firm Kelvin, Bottomley and Baird, and exported all through the world: Brazil (1910), Japon (1914 and 1924), Argentina (1918), Canada (1927)8. They were directly replaced by the computers in the 1960s.

0.3.2 Grapho-mechanical resolution of differential equations

The transition from mathematical machines to computers was not directly issued from the harmonic analyser. The major step between them was the differential analyser, which was built in USA in the 1920s, but was quickly

[14] Anyway, their cooperation did not always take place smoothly.

copied in Great-Britain in the 1930s, and afterwards in Europe. Nevertheless, there was a direct thread between these two analysers, as the second one could be built when was overcome an unsolved technical problem on the first one if it was required to solve differential equations. The "milieu" did not change in nature between the two, but the relationships between engineers, mathematicians and physicists – and beyond, between science and industry - were reinforced during the Interwar, particularly in USA. And the range of graphical tables obtained with the differential analyser was hugely enlarged, as the numerical approximate solutions so obtained greatly exceeded the case of Fourier analysis.

At the very same year when the engineer James Thomson designed the integrator disc-globe-cylinder so as to integrate the product of two functions, his brother William did not only connected several such systems to build the harmonic analysis. He immediately planned to connect together several such integrators in another way in order to obtain numerical solutions for linear differential equations of the second order, by successive approximations (Thomson, 1876a).

If such an equation was reduced to the form :

$$\frac{d}{dx}\Big(\frac{1}{P}\frac{du}{dx}\Big) = 0$$

and let u_1 be any function of x, for instance $u_1(x) = x$, with the following integrations :

$$u_2 = \int_0^x \mathrm{P}\Big(C - \int_0^x u_1 dx\Big) dx$$

$$u_3 = \int_0^x \mathrm{P}\Big(C - \int_0^x u_2 dx\Big) dx$$

then, u_2, u_3, and so on, are successive approximations converging to the solution of the initial equation, vanishing for $x = 0$. Moreover, from this feedback principle, Kelvin could consider the integration of any differential equation of any order.

This method of resolution required a mechanical device to transfer continuously the result between brackets,, obtained from the output of a first integrator, as an input of a second one, which will integrate its product by P. But in the 1870s, no mechanical means allowed Kelvin to succeed in. Il is not very well known that the first engineer who tried to overcome this mechanical obstacle was Alexei Nikolaevich Krylov (1863–1945) [15]. The mathematical problems to be treated in ship floodability issues required numerical solutions. Krylov became aware of the theory of planimeters[16] and of Lord Kelvins works on the mechanical integration. For his researches on the vibration of hulls, he began the construction of a differential analyzer, intended in principle to the "mechanical integration of ordinary differential equations whatever its order and whatever its form", in fact to that of linear differential equations of the fourth order with variable coefficients. This differential analyser was really built[17], but its implementation was challenging the same difficulties in the transmission of movements between integrators as those already faced by Thomson.

This mechanical problem was finally resolved between 1927 and 1931 by Vannevar Bush (1890-1974), from the Electrical Engineering Departement at the *Massachusetts Institute of Technology* (MIT) in the United States. Bush was not only an engineer and a mathematician. He also carried on important responsabilities before and after the World War II. He already served in the submarine detection of the US Navy during the first World War. He became vice president of the MIT in 1932, and successively presided the Carnegie Institue in Washington, and mainly, the *National Defense Research Committee*, which became the *Office of Scientific Research and Development,* when they were respectively founded in 1940 and 1941. This last institution coordinate the war efforts, mobilizing more than six thousand scientists within universities, industry and Defence department (Ségal, 2004, p. 79-89)

Bush and his colleagues of the Electrical Engineering Department overcame the mechanical obstacle encountered by Thomson by using a servo-mechanism named a torque amplifier, which operated on the same principle

[15] In industrial and maritime Russia, this naval engineer began his career in 1884 at the Central hydrographic Service and in the arsenals of the Franco-Russian naval Company. He became professor of mathematics and naval construction at the Naval Academy in St. Petersburg, where he will teach forty years, and will develop an important shipbuilding school, while assuming important administrative and political functions. In 1898, he received the Gold Medal of the *Royal Institution of Naval Architects* in London for his *Theory of oscillations of the vessel*, which was a very rare event. Its buoyancy tables were quickly known and used worldwide.

[16] Krylov first conceived and made a model of planimeter, unfortunately lost today.

[17] It seems that this analyser is still at Saint Petersbourg. But, protected by the secrets of Defense, it cannot be seen.

as a power-operated capstan whose the two drums are driven in opposite directions by a motor. Integrators are thus connected two by two, and enclosed in transparent boxes located laterally. Electrical devices were used for drives and controls. And "A very flexible system of 'bus' shafts has been provided by means of which these units can be interconnected or back coupled at will" (Bush, 1931, p. 450). The shaft representing the variable was also driven by a motor, and drove the rest of the machine in accordance with the equation. Tables were arranged on both sides of the machine, one for reading curves as inputs of the machine, and the others to draw new curves as outputs of the machine. So, in this kind of machine, as well as for the differential analyser, numerical tables became graphical tables, on which numerical values could be read when necessary on graphical paper.

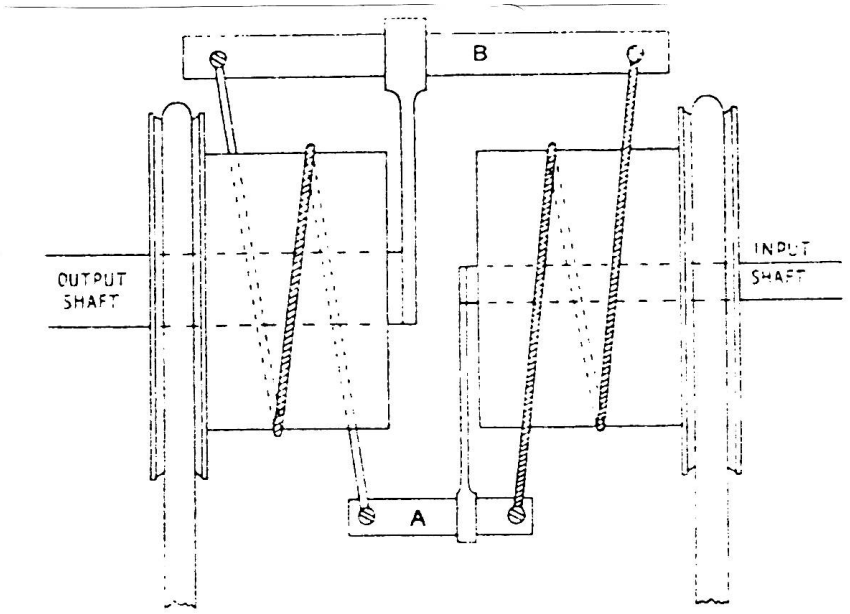

FIG. 4. Principle of torque amplifier.

Fig. 0.10 Drawing of the torque amplifier as given in Hartree 1938

The first differential analyser in USA was operational in 1930, and three full-size models were soon realized in the USA, one at the MIT, the second at the *Moore School of Electrical Engineering*, at the University of Pennsylvania[18] , and the third used by the *Ballistic Research Laboratory* of the Army Ordnance Department, in Aberdeen Proving Ground [19]. In Great-Britain, the physicist Douglas R. Hartree (1897-1958) was immediately interested by the machine, which he had become familiar with reading in Bush's paper. During the World War I, Hartree had already worked on numerical calculations for ballistic in anti-aircraft defense: he assisted his fathers computations in numerical methods, by hand, for the resolution of differential equations [20]. So, Hartree was mainly concerned with computational methods applied in the resolution of differential equations in ballistics, atomic physics and quantum mechanics all over his life. He gained a solid reputation as a fast and advised human computer. As soon as 1932, Hartree visited Bush who gave him the whole help he needed to build a similar machine[21]. He got a first machine built in Manchester University in 1935, by a Manchester firm, Metropolitan Vickers Electrical Co Ldt, with fundings from the superintendant of the university. A second machine was built for the Cambridge Mathematical Laboratory at the outbreak of the war. By the 1940s, the differential analyser was the most important machine used in scientific calculations all over the world. Several other items existed in Great-Britain and in Europe, particularly in Oslo before the war. From this time, Hartree started an exploratory

[18] Dr J. Prosper Eckert and Dr John Mauchly previously worked on this model before developing the ENIAC.

[19] Other full-scale differential analyzers were also built in USA (California, Texas), in Great-Britain (at Queen's University of Belfast in Ireland, and at the Royal Aircraft Establishment in Farnborough) and other parts of the world (University of Sydney, Oslo University, Gothenburg in Sweden, and Leningrad)

[20] He pursued his activities in numerical analysis for his researches on the structure of atom during the Interwar period, before dealing with the propagation of radio waves during the Second World War. Hartree exerted as Beyer Professor of Applied Mathematics (1929-1937), then as professor of Theoretical Physics (1937-1946) at Manchester University, from which he leaved to Cambridge as Plummer Professor of Mathematical Physics (1946-1958). During the World War II, he was affected to the Ministry of Supply.

[21] After visiting Bush, Hartree made a first model of the machine with Meccano elements in 1934. This Meccano model is on display at the Science Museum in London.

work for the Superintendent of External Ballistics. Arrangements were then made with the Ministry of Supply and with the Armamants Research Department to make the two analysers available for the national war effort. During the Second World War they were used for military research, including work on firing tables, magnetron, and on the bouncing bomb used to attack German hydroelectric dams. Thus, the machine was an essential tool in applied physics as well as in mathematical physics, since a general analytical theory of differential equations was missing. As Hartree wrote in 1937, from a conference given at Oslo in 1936 (Wormesley, 1937, p. 353):

> The Differential Analyzer had made feasable to undertake the investigation of many problems of scientific and technical interest leading to Differential Equations which have no convenient formal solution, and which are too elaborate, or for which the range of solutions required is too extensive for calculation of the solutions by numerical methods to be practicable.

This mechanical resolution of differential equations largely extended the range of mathematical tables, and the range of methods used to obtain them, as it involved all sorts of such equations. Moreover, it required the collective effort of physicists, mathematicians and engineers, a coperation which should not be ignored, as it prepared the design of computer programs. The process of working out the machine was first to translate operations indicated in the differential equation as a sep-up diagram for the machine with a mechanical notation which specified each operation and specified the whole organization of mechanical computations. Essential for the effective operation of the engine were the choice of shafts representing various variables or functions and the gearing ratios between them "so that the relations between the rotations of the different shafts satisfy various differential equations" (Hartree, 1935a, p. 940). The wide range of possibilities of such interconnections ensured the adaptability of the machine as a whole to represent and solve a wide range of differential equations[22]. Graphs were prepared showing how changing variables in the equation altered with respect to one another. These graphs, placed on the input tables, were followed by human operators, or by automatic systems, moving pointers geared to the mechanism. Output tables on the analyser produced graphs showing the solution to the equations set up on the differential analyser; from which the numerical values could be read.

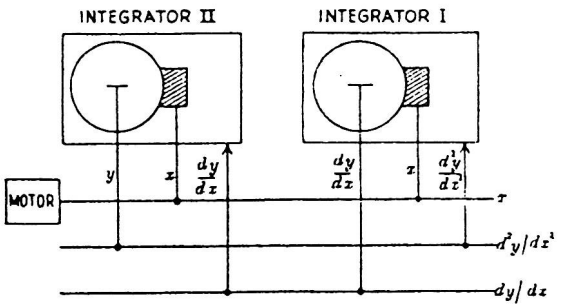

FIG. 6. Set-up diagram for equation $d^2y/dx^2 = -y$. First stage; scale factor and signs disregarded.

Fig. 0.11 Hartree's diagram for the setting-up of the differential analyzer for the equation $\frac{d^2y}{dx^2} = -y$

This is, for instance, the diagram corresponding to the differential equation :

$$\frac{d^2y}{dx^2} = -y$$

This diagram occurred as part of the one for the following equation, where the main variable is t :

$$x'' + kx' + n^2x(1 + ax) = \int \cos pt$$

[22] As it can be noticed, the major part of the space occupied by the machine was not taken by the integrators, but by the connections between the different parts of the machine, which materialized the structure of differential equations. These connections between operations involved technical feedbacks or "hidden sneak circuits", as Shannon also named them when he worked on Bush machine in 1937. The simplification of these circuits was the starting point of his Master thesis on"A Symbolical Analysis of Relay and Switching Circuits", where he gave the equivalence between electrical circuits and the calculus of propositions, and expressed both of them in logical terms, providing a common language for engineers and mathematicians.

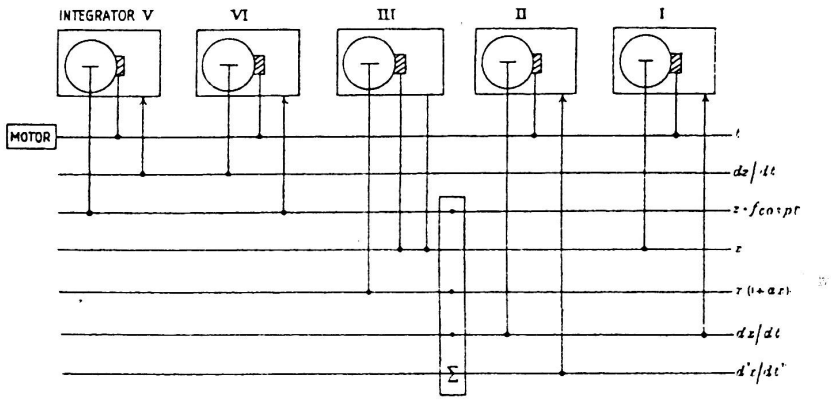

FIG. 10. Preliminary set-up diagram for equation
$\ddot{x} + k\dot{x} + n^2x(1 + ax) = f \cos pt.$
(Scale factors and signs disregarded.)

Fig. 0.12 Hartree's diagram for the setting-up of the differential analyzer for the equation $x'' + kx' + n^2x(1 + ax) = \int \cos pt$

as the integrators IV and V generated directly the second member of this equation by integrating the auxiliary equation :

$$\frac{d^2y}{dx^2} = -p^2z$$

which is analogous to the first equation. When he gave these examples, Hartree focused on the experience which was needed to shorten time in setting up the machine. As a practical man, he began a library of such partial diagrams, which could be used as sub-routines for more complicated cases.

The various steps in setting-up the machine were very time-consuming operations, and saving time was a leading concern for the whole team working on the machine, and of course, for those who needed these problems to be solved. But however, the machine processes saved a lot of computational time. As Hartree wrote in 1935 as soon as the Manchester machine was built (Hartree, 1935b, p. 4):

> The process of setting-up the machine may take from half a day to a day, nut once it is set up, a solution may be obtained in 10 to 15 minutes, this time being practically independent of the complexity of the equation so long as it is within range or the machine, whereas the process of solution of a comparatively simple equation by purely numerical means would probably take someone with experience of calculations of this kind 4 hours to a day of work which needs continuous careful attention, as although it is of a routine nature, it is not altogether simple and straightforward.

So the machine was more efficient when "what [was] required [was] a number of solutions of a single equation (for example with different initial conditions), or solutions of a series of equations of the same general type (for example with different values of a parameter) which [could] be treated with only minor changes of machine set-up" (Hartree, 1946, p. 329). Then "the total time required to obtain a specified set of solutions may be reduced by a factor of 10 or 20" (Hartree, 1935b, p. 5).

The differential analyser largely extended the range of resolution of differential equations, of the kinds of problems solved, and of the graphical output tables so obtained. Hartree first used it to solve ordinary differential equations first in atomic physics. He also adapted the differential analyser to evaluate the solutions of some partial differential equations with two independent variables, in some cases of specific boundary conditions. Hartree used an approximation method previously established [Richardson 1910, Thom 1933, Kindball and Shortley 1934], using and exemplified it on the equation of the heat conduction for one dimension :

$$\frac{\partial\theta}{\partial t} = \frac{\partial^2\theta}{\partial x^2}$$

with boundary conditions : $\theta = 0$ for $x = 0$, and $x = 2l$ for $t > 0$

and initial conditions : $\theta = f(x)$ at $t = 0$ for $0 \leq x \leq 2l$

The approximate solution was established in two stages. The first derivative was first replaced by a ratio of finite differences. So an ordinary differential equation was obtained, which could be treated by the differential analyser. Then, the limit of the evaluated solution was taken, and a correction for the approximation involved in

the process was evaluated and applied. The method was also applicable when the medium was not homogeneous, for instance in the case of the thermal breakdown of dielectric materials in alternating fields (Hartree, 1946, pp. 330-335). So Hartree could treat partial differential equations, first about heat flow under conditions relevant to the steel industry, then about rocket tubes, and finally, about the study of the performance of control systems in automatic following, fire control, and similar purposes. For instance, the mathematical equations that describe the flight of a projectile may be expressed as a set of two partial differential equations where the variations of these empirical data have to be considered, and corrections of a standard resolution had to be introduced.

During the war, one of the major topic needing such numerical solutions of differential euqations was the making of firing tables. It will play a fundamental role in the making of great computers. They gave the necessary informations to aim enemy targets for new armaments produced during the war. The trajectory of a projectile depends on many factors : the elevation of the gun, the direction of fire, the muzzle velocity of the shell, the shell weight, diameter and shape, the wind velocity and direction, the temperature of the gunpowder, humidity and temperature of the air, and even the rotation of the earth for long-range flights. The mathematical equations that describe the flight of a projectile may be expressed as a set of two partial differential equations where the variations of these empirical data have to be considered, and corrections from a standard resolution had to be introduced. The differential analyser authorised to take account of these variations, and to consider them in a very experimental way. One day or more were needed to change from one type of trajectory to another one. The Moore School performed this work using both analog and numerical methods, the former being based on the use of the differential analyzer, the latter being carried out by a separate group of human computers assembled by the Moore School. Gradually, electronic devices were added, particularly in orter to get the input curve followed automatically rather than by an operator.

Clearly, the size of the differential analyser makes manifest the change of scale that had occurred since Kelvin's harmonic analyser. Moreover, this new machine was very well adapted to the needs of research in mathematical physics and its industrial applications. As Hartree wrote in 1946 : 'The accuracy of a full-size machine [was] of the order of 1 in 1,000, and the cost of such a machine with eight integrators – [as it was the case for the Manchester and the Cambridge machines] – would be some tens of thousands dollars" (Hartree, 1946, p. 330). On the other side of Atlantic Ocean, in the United States during the Second World War, Bush supervised the *National Development and Research Center*. With his colleague Samuel Hawk Caldwell (1904–1960) from the Electrical Engineering Department of the MIT, he planned the Rockefeller Differential Analyser, a huge electronic model dedicated for the MIT, a one-hundred-ton machine with 2000 vacuum tubes and 150 motors. It was announced, in spite of the war secrecy, as "one of the great scientific instruments of modern times" . But it was made obsolete by the electronic digital computer *Electronic Numerator Integrator Analyzer and Computer* (ENIAC), and the project failed at the end of the war. Nevertheless, analog and digital computers cohabited in the 1950s. An analogue computer manufactured in 1958 and running with a disc-cylinder integrating mechanism is still on display at the Science Museum of London.

At all events, the harmonic as well as the differential analysers fall under computation practices and a view of knowledge quite different from Babbage's views. For Babbage, steeped in Locke's empiricist philosophy, knowledge came from perceptions and from operations of mind, so that truth lies only in mathematics, and the analyst had to get rid of computations. His engines attempted to embody such operations, from discrete numbers to symbols. On the contrary, for the physicist William Thomson, and for engineers and physicists more generally, truth stood in nature, whose phenomena appeared as continuous. And their machines attempted to simulate such continuous natural phenomena. However, machines working on discrete numbers did exist already at the end of the 19th century. Besides the prototype models of difference engines already reviewed 0.2.6, they were used in other circles than scientific context. and their introduction at first did not look for a complete automation in computations.

0.4 Calculating machines : from desk to punched-card machines

In this period, other areas of computing were concerned by increasing needs, both in quantity and in precision. So, they had to adapt their initial means of computations to this new situations, and to conceive how human computers really worked, in order to explicit the various computation steps to the machine. In these areas, machines were first introduced as an aid to human computers, before taking more and more importance in the whole process of computation.

0.4.1 The issue of granting trust in machine computations

Just as Babbage seized the pretext of recasting trigonometrical and logarithmic tables to develop his own mathematical views on the mechanization of computations, astronomers circles were particularly reluctant to include the use of machines in their computational methods. For instance, all along his long career (1835-1881), the Astronomer Royal George B. Airy (1802-1892) 0.4 expressed a very characteristic distrust against the mechanisation of astronomical or tidal tables [23]. His first judgment relied on Babbage's Difference Engine in 1842, when he was consulted by the Lord Chancellor of Exchequer on the opportunity for the Government to refund Babbage's project of a great engine with six orders of differences and 10-figure numbers :

> An absurd notion has been spread, that the machine was intended for all calculations of every kind. This is quite wrong. The machine is intended solely for calculations which can be made by addition and subtraction in a particular way. This excludes all ordinary calculation. Scarcely a figure of the Nautical Almanac could be computed by it. Not a single figure of the Greenwich Observations or the great lunar Computations now going on could be computed by it. Indeed it was prepared only for the computation of new Tables (as Tables of Logarithms and the like), and even for these the difficult part must be done by human computers. The necessity for such new tables does not occur, as I really believe, once in fifty years.
>
> I can therefore state without the least hesitation that I believe the machine to be useless, and that the sooner it is abandoned, the better it will be for all parties[24].

Forty years later, in 1881, when William Thomson exbibited to the *Institution of Civil Engineers* the whole way he devised the mastering of tidal prediction by the tide gauge, the harmonic analyser and the tide predictor, Airy expressed how he doubted that a mechanical device could extract the Fourier coefficients of the recorded periodic tidal curve. The discussion reviewer wrote (Thomson, 1881, p. 43):

> He [Airy] did no think it possible that it could manage the breaking up to which he had referred; he did not think any mechanism could do it. . . . [S]peaking in general terms, he did not think that it was possible that mechanism could do it with greater ease than it could be done by the use of figures upon paper.

As David Aubin emphasized (Aubin, 2009, pp. 278-283), first when Airy directed the new Cambridge Observatory as Plumian Professor of astronomy and experimental philosophy, then at Greenwich Observatory, Airy gained great fame for the rigorous organization he established for computations and for human computers he established. It could be the main reason why he denied the possibility of substituting brass to brain in astronomical computations. In the hierarchical structure Airy set up of the various steps of computations, "the algorithmic part of the work remained the Astronomer Royal responsibility" (Aubin, 2009, p. 283). At this stage of computations, the various steps of algorithmic processes were not made so explicit as nowadays. Airy's hierarchical organization was not so much unbending as that implied by Adam Smith's principle of the division of labour, as Babbage wanted to set up in his engines. At each level of Airy's organization, human computers had to master several kinds of operations, and they could jump from one level to another when they gained more mathematical knowledge. Moreover, in astronomical computations information had to be extracted from astronomical tables, and used with trigonometrical and logarithmic tables, and then introduced in different trains of arithmetical operations required to obtain the position of a planet or a star (Norbert, 2003, pp. 189-190). Airy's doubts could so be understood as very well founded for the period considered, especially as astronomical computations required much greater accuracy than those engineers needed.

In fact, the way to decompose algorithmically computations connected together in elementary operations was not only linked with classical human calculations on approximation methods. It had to do with the contents and the organization of work. Graphical methods and mechanical analog devices did not open the way to this decomposition, and they were competing numerical analysis at the beginning of the 20th century (Slavyanov et al, 2000, p. v). Moreover, as long as mathematicians carried out their calculations by themselves, they did not need to write the various steps of their work. An experience such as the Mathematical Tables Project, created in 1935 by the Works Project Administration – one important element of Roosevelts New Deal in 1935 –certainly played a major role for materializing computational algorithms. Arnold Lowan had to organize the work, and Gertrude Blanch to prepare worksheets for unemployed workers very unskilled in arithmetic. So, human computation was reduced to strictly mechanical aspects. Actually, these worksheets played the same role than setting up diagrams for analog machines. This kind of table making could also easily meet first punched cards machines, and then, electromechanical and electronic computers (Grier, 2003, pp. 275-277).

[23] More details on the opposition between Babbage and Airy about the mechanization of computations can be found in (**?**)

[24] It is a pleasure here to thank David Aubin, who gave us this reference when he examined Airys papers in Cambridge University Library some years ago.

0.4.2 The crossing of astronomical needs and commercial machines

In parallel, in spite of this reluctance facing mechanisation of computations related to calculation habits in astronomical circles, computing devices were first introduced in a rather modest way. At the turn of the 20th century, the computational requirements increased dreadfully as astronomical instruments gained in precision, and as astronomers regularly needed to develop approximation formulas with more and more terms, so as to reconcile with new observational data the planetary theories represented by tables and instructions to use them (Wilkins, 2003, p. 296). Desk machines were progressively introduced to help and relieve the work of human computers, but the process took place slowly, as human computers were usually few and poorly paid, while buying expensive machines needed to find funds. More and more numerous and sophisticated existing automatic machines from commercial and statistical developments had to meet the computational requirements and the financial capabilities of astronomical work, first of all in London, and quickly more internationally. As the astronomer Wallace J. Eckert (1902)1971) wrote in 1940 about using these machines at the *Thomas J. Watson Astronomical Computing Bureau*[25], in the Department Astronomy of Columbia University (Eckert, 1940, p. 1):

> The main question in any case is not 'Can the problem be solved by these machines ?' but rather 'Have I enough operations of this type or that to justify such powerful equipement ?' It might be mentioned also that, with standard parts, special machines for almost any imaginable purposes could be constructed, but except in very rare cases the expense and delay involved make the standard machines preferable.

In London, the astronomer Leslie J. Comrie (1893–1950) gave a first essential contribution for thinking new relationships between human and mechanical computations. He was initiated to mechanical computations on a Brunsviga machine while graduating at University College (London) just after serving in the New Zealand Expeditionary Corps[26] during the World War I. And he quickly replaced the *Companion to Observatory* by a *Handbook of the British Association for the Advancement of Science*, when he created and directed its Computing Section in 1919. Since 1924, when he supported his Ph. D. thesis at Cambridge on the prediction of planetary occultations, he quickly gained a specific expertise for adapting to each other computational methods in astronomy and various types of machine. As soon as the 1924 meeting of the *American Astronomical Society*, he planned to get astronomical computations rid of logarithms, what he succeeded before 1932 with the "complete mechanisation of calculations" (Comrie, 1932a, p. 523). Actually, Comrie soon acquired strategic positions so as to test his views about interactions between computational methods and machines, and get them to be adopted in the right places. At the *Nautical Almanac Office* from 1925, where he was soon Deputy and Superintendent (1930-36), and as a member (1928-36 and 1929-50), and secretary (1930-36) of the *Mathematical Tables Committee of the BAAS*, he launched new publications and conceived new techniques so as to use the desk and punched card machines for great amounts of iterative operations. Rather than prompting firms to build special-purpose machines for scientific computations – as their financial burden would have been unsustainable –, he looked at the best way to manage his methods on existing machines, preferring his collective work of many engineering minds with now professional computers to "the product of a single brain" (Comrie, 1932a, p. 523).

Discussing with the leading manufacturers – as Block and Anderson, the British agent for the Brunsviga machine, or with Hollerith and the British Tabulating Machine Co –, giving many talks and papers on machines, suggesting new applications not considered by the builders, Comrie obtained machines on loan for trial and expertise. Leading organizations consulted him for advice on computation problems, as the *Colonial and Military Survey or the Armament Research*. And it was also the case for individual scientists. For instance, before the differential analyser was built in England, he convinced his assistant and future successor at the *Nautical Almanac Office*, Donald H. Sadler (1908–1987), to help Hartree in his calculations on self-consistent atomic fields (Sadler, 2008, p. 31) [Sadler, 2008, p. 31]. In the United States, the famous astronomer Ernest W. Brown (1866-1938), who just published his new *Tables of the Motion of the Moon* (1919), was very much excited when Comrie showed him the speed of the Hollerith machine (Comrie, 1946a). Brown transmitted his enthusiasm to his former Ph. D. student Eckert who soon founded the *Astronomical Computing Bureau* (1937) at Columbia University, where he generalized the use of punched-card machines for solving problems in Celestial Mechanics – particularly to improve Brown's theory of lunar motion[27] – for which he published the textbook *Punched Card Methods and Scientific Computation* (1940). Obviously, as William Thomsons time coincided with the

[25] Doctor Thomas J. Watson was the president of the IBM Corporation, and his cooperation and encouragement was essential to the creation of this laboratory from the very beginning.

[26] Comrie had his left leg torn off during the war, and wore an artificial one.

[27] Brown is known for his computations of the lunar orbit which guided Apollo missions.

extension of tidal observations at a world scale, Comrie's time corresponded to the same kind of expansion for astronomical observations. So, his work did not only address to professional astronomers in observatories, but to amateurs, who needed detailed methodological explanations. The extension of mechanical computation to more general scientific activities was a major concern for him, mainly since 1936, when he was dismissed from the *Nautical Almanac Office* – because he carried out computations on its machines from outside – and founded the *Scientific Computing Service* (SCS), the first commercial firm specialized in scientific computation. After the war, this expansion will be much more evident as several government departments and their computing services or laboratories were concerned, as for instance in Great-Britain, *Admiralty Computing Service* (1943), Cambridge Mathematical laboratory (1937), National Physical Laboratory Mathematical Division (1945). It was materialised in the West World by the variety of institutions concerned with tables : the *Nautical Almanac and Astronomical Ephemeris*, published from 1767 to 1959 first by the *Board of Longitudes*, and then by the *Nautical Almanac Office*, changed its title to *The Astronomical Almanac* when it merged in 1961 with its counterpart in the United States produced by the Naval Observatory. Meanwhile, international agreements induced the ephemeris offices of various countries to share basic calculations and other astronomical ephemerides (Observatory, 1965, p. iv). Clearly, Comrie worked at a period of momentous geographical and conceptual expansion of table needs for astronomy and applications in physical sciences. His enthusiasm for computing machines and his high level of demand for accuracy tables undoubtedly made him a major actor in this general unifying process, which will be still more amplified with the emergence of electronic and programmable computers.

0.4.3 The mechanization of human work in astronomical computations

As soon as 1924, Comrie's address at the *American Astronomical Society* meeting was a real program for the mechanisation of astronomical work on tables. Unlike Babbage's one, it did not intend to impose from above a general view of mathematical analysis, and to substitute a machine for the whole work. Comrie started from the state of facts : as calculating machines became more reliable, more sophisticated and cheaper during the last twenty years, he first considered as possible to support mechanically the repetitive parts of human computers work facing so huge an increase of the mass of calculations in the same period. His first aim in 1924 was expressly to shake the inertia related to the organization of work on logarithmical method – "the century old logarithmic method of Bessel occup[ying] the field" –, by a systematic comparison with more "natural" new ones, tested on the various existing machines, carefully discussed with their advantages and disadvantages, for an extended range of astronomical problems : ephemerides of comets and minor planets, reduction and prediction of solar eclipses and lunar occultations, stars reduction from the mean of apparent place, determination of orbits from three occultations (Comrie, 1925, p-. 243-246).

Thereby, Comrie's papers clearly deployed this strategy, associating description of machines with computational methods, and retracing the historical background for the support of computations by instruments and machines. First eager to focus on the method of differences he adapted on commercial machines, Comrie scored them in the lineage of difference engines, to which he often referred in 1928 (F.R.Philips et al, 1928, p. 106). He pointed out Babbage's project[28] and his gold medal of the *Royal Astronomical Society* in 1822. He gave specific informations on the difference engines successively built by Scheutz, Dunkin, Wiberg, Grant, and mainly the one by Hamann at Berlin-Friedenau, with which Professors Bauschinger andPeters prepared the 8-figues logarithmic and trigonometrical tables commissioned by the *Royal Prussian Academy of Sciences* of Berlin and by the *Imperial Academy of Sciences* of Vienna at the dawn of the century (Comrie, 1928a, p 450). As it was already the case for the unpublished *Tables du Cadastre* (1794–99) of Baron de Prony, the method of differences was used to interpolate between known values with second differences, and Comrie adopted this way of using it on commercial machines. What was now at stakes for him was to involve this method to reckon not only these fundamental tables[29], but all kinds of functions, through the totalizing possibility of desk machines. Comrie could rely on the first initiative of T. C. Hudson, from the staff of *Nautical Almanac Office*, who first introduced a Burroughs Adding Machine, which could also perform subtractions and print the results. It could only perform a single series of summations, but Hudson contrived it to work from the second difference by feeding

[28] Comrie often referred to Babbage, even later on when reviewing the Harvard IBM ASCC of Howard H. Aiken (1900–1973) (Comrie, 1946b)

[29] As "the accuracy of observations has so much increased during the last 50 years, the common seven-place tables no longer satisfy the demands of those who have engaged in refined work, especially in astronomy". Very accurate tables, essentially for logarithms of sines, cosines, tangents, and cotangents, to fourteen places at intervals of ten seconds, also with logarithms at eighteen places of certain numbers, were still computed strictly by hand, without the help of any machine, in 18 months, and published by H. Andoyer in 1911. This volume will be followed by three other ones between 1915 and 1918 (Brown, 912, p. 366).

the machine two times, which also allowed to detect at once any error in the summation[30]. But even if a new Burroughs machine with an extra register could now work directly from the second differences, Comrie preferred the Brunsviga-Dupla machine in 1928, in spite of its lack of a printing device, not only because it costed one third the price of the Burroughs, but essentially for several technical advances, which offered new possible computations. These technical advances were supported by a new second register, which authorized cumulative sums and direct transfer from one register to the other. So, this new register allowed the method of differences and its double summation to be entirely mechanised, even with positive and negative second differences [see fig. So well fitted to mechanical integration, it could also be used in a direct way to reckon second differences from the function itself, opening to the automatic control of tabulated values. So it could be completely used to practice classical interpolation by the method of differences, some pivotal values being obtained directly from the function, while the others were computed by interpolation. Comrie compared in details the method of differences, preferred in astronomical circles, to the Lagrangian method preferred in statistical circles. He opposed Karl Pearson (1857-1936) and showed Lagrange interpolation was not well adapted on the new machines, that it could leave undetected errors, and that it needed to handle large quantities and large coefficients [which were not] progressively diminishing as in ordinary interpolation[31] [Comrie, 1928b, p. 507-510]. In its place, he preferred the Bessels formula of interpolation, which brought an almost quite complete security and lightened the labour of computations. Comrie also devised a new way of using the difference method, the end-figure differencing, for which he acknowledges his debt to Hudson, who first conceived it from graphical methods (Comrie, 1928b, p. 523).

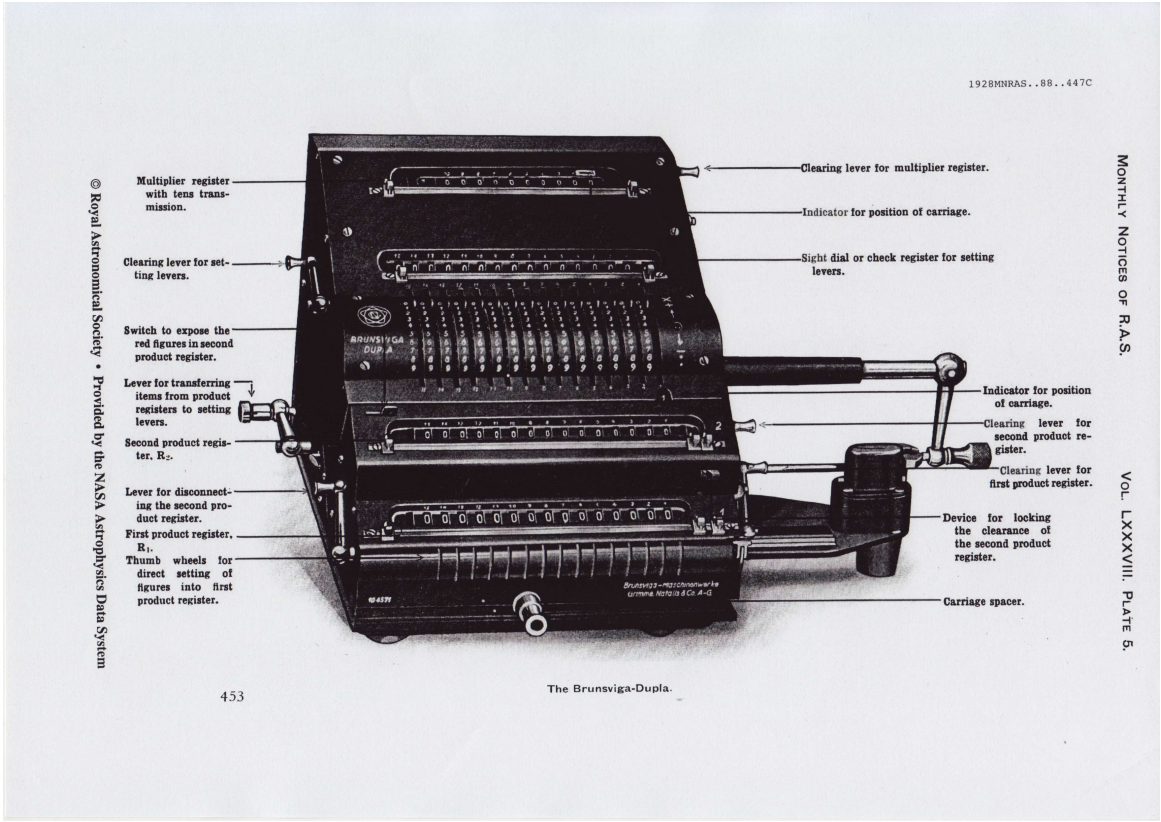

Fig. 0.13 The Brunsviga-Dupla machine with its two registers, presented in Comrie. 1928

The performances of each new machine built, and quickly introduced in the NAO, allowed to enlarge the whole range of computations : the time for repetitive operations was shortened, their checking was improved, and

[30]It was presented at the Napier Tricentenary Celebrating Exhibition, which has such an impact that another great exhibition on calculating machines was organised in Paris by the *Socit dEncouragement pour lIndustrie Nationale* (SEIN) in 1920

[31]Some years later, Eckert will be more circumspect than Comrie, and will choose the method according to the nature of the problem and to the previous computations. For instance, in the case of integration of the differential equations of planetary motion, he selected the method of differences because of the check it offered with the differences; but in the the interpolation of the perturbations for the integration, he preferred Lagrange's method. He also referred to Bessel's and Stirling's formula, and to Simpson's rule (Eckert, 1940, p. 44-47).

embedded in the methods themselves. Comrie built these methods to draw the best from commercial machines, while sticking closer to the work habits of his colleagues. First of all, with its two product-registers and their transfer feature, possibly connected or disconnected at will, the Brusnviga-Dupla could still work as a difference engine, but it also opened the way for new types of computations, such as multiplication or division by a constant factor, correlation or the method of least squares. These examples "[did] not exhaust the scientific possibilities of this remarkable machine", as Comrie concluded, possibilities which were unsuspected by its builders, and could inspire human computers since it was "now within the reach of all scientific institutions" (Comrie, 1928a, pp. 447-459). Sub-routines could be already established such as cycles of operations, with the list of operations to be performed and their results. These avantages of the Brunsviga-Dupla machine were also presented at the meeting of the *Royal Astronomical Society* in 1928, where Andrew C. D. Crommelin (1865–1939) applauded these new possibilities, while his own work on Halleys comet with Philip H. Cowell (1870–1949), the previous Superintendent of the *Nautical Almanac Office* (1910-1930), elaborating the well-known "Cowells method" for analyzing perturbations, had still been entirely done by hand (F.R.Philips et al, 1928, p. 107). Later on, Comries technical advances on machines were very efficient to compute tables using Cowells method (Comrie, 1928a, p. 449). The Burroughs Class II, reviewed in 1932, was "really a difference engine, realizing the ambitions of Charles Babbage". Examples were given of its use, for instance to obtain and control the mean positions of stars for the annual precession, the mean longitudes, longitudes of nodes, perigee, as moments for finding means and standard deviations, and also Pearson functions [Comrie, 1932a, p. 535]. But essentially, its running efficiency was so impressive that it deserved to be pointed out with great details (Comrie, 1932a, pp. 540):

> The work of this machine was thoroughly organised in 1930, and a strict record of its performance was kept throughout 1931. Three operators kept the machine running 8 hours per day, no one operator being on the machine more than 2 hours at a stretch, or more than 4 hours a day. The total number of function values developed and printed was $1,200,000$. This involved $6,000,000$ motor-bar operations, and the printing of about 30,000,000 figures. It would take a copyist, working ordinary office hours, 7 years to copy thse figures without any calculation! There can be no doubt that this is a record for any Burroughs machine, due to the particular nature of the cyclic operations, namely, a small keyboard setting and the printing of two sub-totals between each setting.

Nevertheless, the masterpiece of the *Computing Bureau* of the NAO will undoubtedly be the Hollerith machine, a punched-card machine built by the *International Business Machines Corporation* (IBM).

0.4.4 Crossing with statistical machines

The Hollerith machine was installed first for seven months and Comrie immediately explained how it could tabulate for the in 1928. From its first use, "the mechanical methods which have been applied to certain portions of the work have eliminated much fatigue, increased tenfold the speed with which results can be obtained, and reduced the cost to one quarter to its former amount" (Comrie, 1932a, p. 694). Its main use was the summation of harmonic functions in the calculation of the lunar orbital elements. Brown's tables exemplified his renewed lunar theory, initially published as a succession of papers[32] in the *Memoirs of the Royal Astronomical Society* between 1897 and 1908. They exhibited twelve years of work, in three volumes of 180 tables spread over 660 pages, and were intended to obtain the lunar positions from series of harmonic functions whose more than 1400 periodic terms, and coefficients as large as $0''003$ were given in the tables. But the instructions to use them were very complex, so that manual tabulation was still very painstaking and could generate numerous errors. So, if "copying was done once only, [if] the addition was done mechanically, and the results of the addition printed", the task was already substantially alleged. Comrie exhibited with details how the mechanical tabulation was modeled on the tabulation by hand, especially for calculating the long- and short-period terms of nutation[33].

In manual tabulation, data had to be taken in rows and columns of tables in an irregular way, depending on the specific orbital element to be obtained, before executing the appropriate summations. With the Hollerith accounting machine, each table was represented by a pack of cards[34]. Packs of cards were often prepared in advance, even from other establishments[35], and sent to the staff with the detailed instructions and checks, to

[32] Brown received the Gold Medal of the Society in 1907 for this work.

[33] Sadler more generally sketched this transition from manual to mechanical tabulation in his "Personal History of *H. M. Nautical Almanach Office*".

[34] Cards in Comrie's machine had 45 columns, Eckert's one 80 columns. "The cards were fed and ejected by an electric motor, the punching is done magnetically, and information from one card can be automatically transferred to another by the duplicator" (Eckert, 1940, p. 8).

[35] For instance, in 1929, punching was started six months before the arrival of the sorter and the tabulator, as it was necessary to punch $20,000,000$ holes in half a million of cards" (Comrie, 1932b, pp. 706).

be set and plugged on the sorter and tabulator by an operator[36] for the particular layout of the cards. The choice of the cards to be used in good order for successive dates was still achieved by hand to get new groups of cards to be provided to the tabulator for adding and printing. This tabulator could either print every item added and the total, giving a "listing, or printed only the total, giving a tabulation". But the two main and new technical elements of the Hollerith were of course the sorter, by which punched cards could be separated into groups, and the plugboard, analog to a telephone swtichboard, by which the sequence of operations of the tabulator was controlled automatically, for "it was possible, by means of plugwires, to select the positions on the cards from which the numbers are to be taken and the positions in the adding mechanism where they are to be added" (Eckert, 1940, p. 12). With all these combined elements, "the mixing process, tabulation, sorting and replacement proceeded continuously until the entire calculation is completed". Eckerts Hollerith machine was also able of multiplications.

Comrie felt very concerned by the efficiency problem, both for the work of human and mechanical computers – even for cards themselves –, and for the proper use of the invested funds. As Sadler pointed out : "[The methods devised] required the willing cooperation of staff to do repetitive tasks at high speeds with a very high standard of accuracy [...] and Comrie, as a matter of principle, insisted on making optimum use of staff and machine" (Sadler, 2008, p. 23). In his main papers on the mechanization of tables, Comrie systematically compared the costs for buying or renting machines, according to their own performances. And he detailed the gain of time for each element of the machines and part of the computation. For instance, in his paper on the Hollerith machine, he specified that "a first-class operator will punch 300 cards an hour', ...the sorter [worked] at the rhythm of 24,000 cards an hour (Comrie, 1932b, pp. 699), the speed of the machine when listing [could attain] 4500 cards per group", and that, "for the calculation of long-period terms of nutation, the tabulation for one year [took] less than an hour" (Comrie, 1932b, pp. 703). He concluded for tthe double-entry tables (Comrie, 1932b, pp. 703):

> It was in this particular section of the work that the greatest efficiency was attained; on the average 20 figures were added per second, and the totals printed. The totals for 44 tables for two years (1460 dates) were done by three girls and the machines in one day.

And as Comrie insisted on the fact that the machine possibilities had to be used at their maximum of rate, he explained why tables were prepared with the Hollerith for until the year 2000 (Comrie, 1932b, pp. 706):

> The work described on short- and long-period nutation, and on double-entry tables, as well as that of most of the single-entry tables, was carried to the year 2000. The greater part of the cost was incurred in doing the first ten years, which would have sufficed for immediate needs. But to continue for the next 55 uears with a trained and organised staff added very little to the cost, and was certainly more economical than re-training and re-organising the years later. Moreover, there is little likelihood of Brown's *Tables* being superseded before the end of the century; any acquisition to our knowledge of the Moon during the next seven decades is almost certain to be expressed in the form of corrections to Brown's Tables, not in the form of new tables.

So, in parallel with analog machines, digital accounting machines were essential to support the whopping size of computations necessary to the state of physical and engineering sciences in the first part of the 19th century. It could be said that Babbage's program for quantiying all natural phenomena and treating the results by mathematical analysis 0.3 was superseded to attain now artificial phenomena from human achievements. At the beginning of the World War II, collaborations where W. Thomson was engaged some decades earlier between scientists, engineers and industrials, were now extending to commercial and financial circles already with the accouning machines. Eckert insisted on about the interest and generosity of Dr Thomas J. Watson, president of the IBM Corporation, in the creation of the *Th. J. Watson Astronomical Computing Bureau* –with its computing laboratory –, a scientific non-profit-making institution which was organized in 1937 at Columbia University (Eckert, 1940, p. iii-iv). Inside the laboratory, the research on computations by machine ensured to combine human thought and new mechanical possibilites –as it was the case for the differential analyzer – so as to simplify its processes (Eckert, 1940, p. 25):

> The successful and efficient application of the machines to a scientific problem requires careful considerations of human and mechanical elements. The machines are not infallible, so numerical checks must be provided. The machines are not infinitely efficient, so it is necessary to take advantage of all short cuts. The obvious direct method is frequently much more costly than necessary. It is often possible to substitute a little manipulation of the cards by the operator for long machine processes. On the other hand, the operator should not be assigned tasks which are psychologically difficult to perform without error. [...]

[36] These operators were often women. Sadler particularly remembered the dexterity of the expert operator Mrs Frayne, who was also "magnificient in handling the crises", probably originated from different rhythms between the machine and the staff. He noticed that Comrie was often accompanied by Miss L. H. Burr to demonstrate when he gave lectures (Sadler, 2008, p; 23). Some years later, in the Unitd States, Sadler also appreciated how Miss Lilian Felnstein has contributed largely to the success of several of the applications in part III and had personally performed most of the computations" (Eckert, 1940, p. iv).

> The planning of an extensive program thus requires a careful analysis of many factors in the light of all available data and experience.

On the conceptual side, setting diagrams for differential analyser, and schemes for automatic control on the Hollerith machine constituted specific and precise modelling for future programming on electronic computers. Anyway, unification was still missing between various scientific circles, as it can be understood when Comrie suggested in 1932 how it could be possible to use the Hollerith machine for tide prediction, arguing that it was still limited to the times and heights for low and high waters, and that it would be economical to introduce the Hollerith to obtain hourly predictions, but only for some very busy ports (Comrie, 1932b, pp. 707). Obviously, Comrie was completely ignorant of the developments of tide predictions by analog machines. Even if Eckert's textbook on *PunchecCard Method and Scientific Computation* in 1940 intended to enlarge its use to other domains than astronomy – essentially by means of the construction and use of special tables of tabular functions – it still needed to "enable a scientist so to formulate his problem that any skilled operator of the machines could carry it out" (Eckert, 1940, p. iii). Differential equations and harmonic analysis were fundamental fields of research in physical sciences, and were now treated by approximations from different perspectives. It is to be noticed that, already with accounting machines, tables began to be embedded in the machine, as Eckert insisted on the fact that printed tables appeared now only as punched cards, and that long series of operations could be carried out without reading or writing one number by hand. But it will be only with general programming on electronic computers that the process will be reverse, each scientific domain becoming a possible application of symbolical computations on the same kind of machines. A new issue will be raised with the making and running of electronic computers, as they will progressively make invisible these diagrams and schemes which represented the mathematical structure of the problems, as thought by humans. This internalizing by machines will a a new step in the reification of human work. But as it will appear in the next section, human skill and thought must still be present in the process for an intelligent use of mathematical machines, even of computers;

0.5 Transitional machines

In the 1940s a new type of calculating machines was developed: the electronic, digital general-purpose computer. These machines, which are considered as the "first" computers in the sense we understand it today, have part of their roots in earlier machines such as, for instance, the differential analyzer and hence also in table-making. However, because of their speed combined with their programmability they also entail a rupture with these roots. One of the earliest examples of such machines is the ENIAC. In an interview Ida Rhodes, member of the Mathematical Tables project, recounts the following story about this computer (Tropp, 1973):

> When the ENIAC was finished, [Dr. Lowan] was invited to watch it. [He] came back and said, "We're finished. They don't need us anymore. Do you know," he said, "what they do? They don't look up Tables. They actually compute each value ab ovo." And to me that sounded so impossible, so incredible[...] To compute each value ab ovo. Not to have to look up one of our marvelous Tables. That sounded like [the] death knell. We were [quite] unhappy about such a possibility.

This situation is very telling about the impact these new computers had on table makers and table making: given the speed of machines such as ENIAC it seemed no longer required to use tables in a computation: the machine itself could compute each value of a table as it was needed. Hence the future of numerical tables seemed, at best, precarious. This pessimism about the numerical table is taken up in the book *The history of mathematical tables. From Sumer to Spreadsheets* (Campbell-Kelly et al, 2003, p. 5) where it is stated that the "*computers have been the death of the printed table-as-calculating-aid*". The only kinds of tables from the digital era that are discussed in-depth are spreadsheets considered to give "*new and vigorous life [to the] table-as-data-presentation format*". But in what sense exactly did the computer result in the death of the table-as-calculating-aid and what kind of transitions and transformations can we detect in the (early) period of digital computing? We will show in the remainder of this section how methods for computing and using tables are affected by the introduction of the computer and how it resulted in the development of tables that are used as calculting aids, not by humans but by the machines themselves.

0.5.1 Towards electronic general-purpose computing

As we explained in Sec. **??**, in the first half of the 20th century, there was a growing need for different kinds of tables and it is for this reason that people like Hartree and Comrie were very much concerned with the speed of the complete computational process including the setting-up of a particular calculation on the machine and

the additional mechanical and human operations (See pp. 25 and 32). The second world war only increased this need. Especially firing tables were very much needed. In the U.S., the Ballistic Research Laboratory (BRL) was responsible for the computation of firing tables. They worked in close collaboration with the Moore school for Electrical Engineering at Penn University, which was about one hour with the train from the BRL. It would also be here that ENIAC would be developed.

In 1937, the BRL financed the construction of a differential analyzer at the Moore school "*under an agreement that allowed ballistic researchers to use this machine in times of war [...] In June 1942, proving ground officials notified the University of Pennsylvania that they needed to use the differential analyzer for ballistics research*" (Grier, 2005, p. 258). Since qualified staff was scarce at BRL, a training programme was set up at the Moore school for female students, to help with the prepation of firing tables using desk calculators. Despite these efforts, firing tables were produced at too slow a pace. The differential analyzer needed about 15-30 minutes to compute one trajectory but was not as precise as hand calculations. As a consequence, additional staff was needed to compensate for the machine's shortcomings. Hand calculation aided by desk calculators, though more precise, required about two 8-hour days for one trajectory (Polachek, 1997; Grier, 2005). As a consequence, despite "*the extensive arrangements the [BRL] made with the University of Pennsylvania for assistance, the backlog continued to grow*" (Polachek, 1997, p. 25) It was exactly in this context of extensive but too slow computations that Mauchly's proposal for the ENIAC was perfectly timed. Indeed, contrary to the differential analyzers or machine-aided hand calculation, it was an electronic computer and could thus compute at a speed which is several orders of magnitude larger than those of its predecessors. In 1943, the US army decided to finance the construction of the machine, hoping that it would help to resolve the firing table bottleneck. However, being built at the same location as the differential analyzer there were certainly also some important continuities. In fact, it is clear that given the context in which ENIAC was built and used, several influences and competences from other machines like the differential analyzer but also the IBM relay calculators that were around, must have affected its design and use.[37]

Being an electronic machine, ENIAC was also a digital machine: it did not work with continuous inputs and/or outputs but with discrete data. Besides, ENIAC was a general-purpose computer: even though it was presented to the Army as a special-purpose computer to compute firing tables, its architecture was flexible enough to use it for a variety of different purposes and problems. In fact, once it was capable of discrimination, it was possible, in theory,[38] to set-up any computatonal problem.

The idea that one machine is used for a variety of purposes is certainly not new: we already know that Babbage's analytical engine was conceived as a kind of general-purpose machine and both the differential analyzer and Comrie's machines had a tendency towards general-purposeness. For instance, Hartree's differential analyzer could basically compute any differential equation which made possible its use to study many different kinds of problems (See e.g. pp. 0.3.2). What is new, is that it can be used not just for one class of problems (for instance, differential equations) but for *any* class of computable problems, going from interpolation problems to problems in number theory.

ENIAC was revealed to the public in 1946 and publicized widely in the scientific and popular press. The following quote from a Fox Movietone news broadcasting gives an impression of how the machine was announced to the general public:[39]

> *Are people becoming obsolete?* [m.i.] A giant electronic brain is starting calculating at the University of Pennsylvania [I]t can add up a column of figures a yard long in a second [R]ight now it is solving mathematical problems for the U.S. army but who knows someday a machine like this may check up on your income tax.

Despite its popular reception, ENIAC is by no means comparable to current digital computers: it was, as Martin Davis once called it, a "*behemoth of a machine*" (Davis, 2001, p. 191). Not only did it fill up a whole room but it also did not have anything that resembles a language-oriented programming interface. In its original format, the machine had to be programmed directly and locally by setting switches on the different components and by wiring cables to interconnect these components and control the sequencing of the program's operations.[40] Contrary to the differential analyzer, the operator did more than just a mechanical job: she actually helped in "programming" a problem. For more details on ENIAC's architecture see (Goldstine and Goldstine, 1946; Goldstine, 1946)

[37] For a more detailed account of this see (Burks and Burks, 1981), though it must be pointed out that this account does not do full justice to ENIAC and its design team and perhaps assigns too important a role to John von Neumann.

[38] Viz., by making abstraction from its memory limitations and the practical problems with programming the machine (see below).

[39] Available from youtube: http://www.youtube.com/watch?v=OSYpYFEwr4o

[40] ENIAC, in its original constellation, had a permanent team of six female programmers/operators, though it is also clear that several others worked with the machine (for instance, Adèle Goldstine, John or John Mauchly)

The difficulties involved with programming the machine is also the reason why Jean Bartik, one of the female programmers, described the machine as "a son-of-a-bitch to program" (McCartney, 1999, p. 94). In this sense, ENIAC, especially in its original constellation, certainly was a transitional machine, soon to be replaced by the stored-program computer. In fact, in 1947, it was moved to Aberdeen Proving Ground where it was completely rewired into a so-called stored-program computer (Haigh et al, 2014).

Notwithstanding these programming problems, the machine significantly impressed and affected many computer pioneers. Combined with its general-purpose character, the speed of the machine made possible its usage by a variety of different people and for different purposes. It was used by logicians, number theorists, physicians, meteorologists etc and its applications vary from computing composite numbers to making calculations for the H-bomb.[41] And it was because so many different people actually used the machine that its general-purpose character could be fully demonstrated. It is in this sense that machines such as ENIAC created a context of collaboration between different milieux. The programming bottleneck only stimulated this collaboration since in order to set up a computation on the machine, one not only needed the right background to turn the problem into an algorithmic problem, but one also needed detailed knowledge of how the machine works. In such collaborative context which developed around a very new kind of machine against the background of the ever growing needs of the military, it is not surprising that ENIAC also resulted in the development of new fundamental techniques that are still used today. The Monte Carlo method, the use of random numbers in numerical experiments, is probably one of the most famous of these methods (See p. 37). As such ENIAC had a wide-felt impact on the history of computing and serves as a good paradigmatic case to help understand what happened with the numerical table in the early years of digital computing.[42]

0.5.2 From tables to algorithms

Besides the difficulties involved with programming the machine, ENIAC also did not have a very large electronic memory – it would take several more years before the so-called memory bottleneck would be technologically resolved.[43] ENIAC's main electronic memory consisted of three function tables and the constant transmitter. The function tables, which were designed for storing tabular data, could each store 104 entries consisting of 12 digits and two signs each. The constant transmitter, which was mainly used to convert data coming from punched cards to electronic pulses, could also store up to 20 digits and 4 signs (Goldstine and Goldstine, 1946, p.105). This is not much. As a consequence, if larger amounts of data were needed, one had to rely on the mechanical external memory (the punched cards) which meant that ENIAC was still very much relying on older and slower technologies, viz., those we discussed in some of the previous Sections. This seriously slowed down computations. Furthermore, since often new cards needed to be inserted manually, the computation could not be fully automatic. Hence one could not take full advantage of the electronic speed of the machine. Because of this problem, alternative strategies which avoid the use of large data sets were developed, when feasible: this meant the replacement of these data by algorithms so that ENIAC, rather than having to copy down these data from the slow external memory, computed its own values when needed. It is exactly in such context that the replacement of numerical tables as calculating aids by algorithms is not only a mere possibility because of the speed of the machine but also a practical necessity.

There are several examples which show that the people involved with ENIAC were very much aware that the machine allows for the replacement of tables by algorithms. A first example comes from the first number-theoretical computation set-up on the machine. During a labour day weekend in 1947 Derrick H. Lehmer, his wife Emma and their children spent their time with ENIAC to compute a table of exponents e of 2 mod p, viz. the smallest value of e such that $2^e \equiv 1 \bmod p$.[44] Lehmer had been asked to become a member of a Computations Committee that was set up at the BRL and whose main task was to test computing machinery,

[41] See (Fritz, 1994) for a list of the different problems ran on ENIAC

[42] To be clear, ENIAC was certainly not the only machine of this type being developed at the time. Another example is the Manchester Baby, built at the University of Manchester, UK (Napper, 2000).

[43] Of course, memory is up to today an important problem in computer engineering, but not in the same way as it was in those early years.

[44] See (Bullynck and Mol, 2010) for more details on this ENIAC computation, including a detailed reconstruction of the wiring.

with a main focus on ENIAC (Alt, 1972). The committee had three other members: F.L. Alt[45], H.B. Curry[46] and L.B. Cunningham.[47]

It was a known fact that Fermat's little theorem could be used as a primality test: If for a given number b, $2^b \equiv 2 \bmod b$ than b is with high probability a prime number. Unfortunately, an infinite set of exceptions to this primality test exists. To turn this into a real test for primality one needs to combine it with a table of exceptions to the test. A table of exponents can be used to compute such exceptions. Before, Lehmer had been using Kraîtchik's tables. These tables, however, only extended to 300 000, and contained rather a lot of errors. Correcting and extending this table without electronic computers meant that "*some 10 or 15 years of recomputing and extending [of] this table*" would be required. With ENIAC, this 10 to 15 years could now be reduced to a weekend: a table of exponents for $p \leq 4.5 \times 10^6$ was computed. This resulted in the publication of a list of errors to Kraîtchik's tables, a list of factors of $2^k \pm 1$ with ≤ 500 and an extension of the existing table of composite numbers n dividing $2^n - 2$ from 10^8 to 2×10^8.[48] Fig. 0.14 gives an extract from the table of exponents published in (Lehmer, 1949).

TABLE OF COMPOSITE SOLUTIONS n OF FERMAT'S CONGRUENCE $2^n \equiv 2 \pmod{n}$ AND THEIR SMALLEST PRIME FACTOR p

n	p	n	p	n	p
100463443	7577	312773	3541	558011	6449
618933	4729	413333	6067	940853	503
860997	9649	495083	1987	120296677	229
907047	5023	717361	1013	517021	2341
943201	5801	111202297	5273	838609	433
101152133	5807	370141	883	121062001	1201
158093	3673	654401	6101	128361	6961
218921	8713	112032001	4001	374241	6361
270251	9001	402981	3061	121472359	4409
276579	6163	828801	6133	122166307	739
954077	1597	844131	3067	396737	2857
102004421	2381	113359321	761	941981	337 · 491
443749	4049	589601	331 · 571	123330371	691
678031	3583	605201	7537	481777	3881
690677	2069	730481	433	559837	4177
690901	5851	892589	919	671671	9631
103022551	6121	114305441	6173	886003	1187
301633	7873	329881	7561	987793	709
104078857	6679	469073	3089	124071977	2089
233141	2441	701341	1229	145473	397
524421	5903	842677	2459	793521	4561
105007549	1033	115085701	1801	818601	2281
305443	2833	174681	773	125284141	4231
919633	4603	804501	5381	686241	6473
941851	1051	873801	1051	848577	2897
106485121	7297	116090081	6221	126132553	5023
622353	433	151661	7621	886447	6793
743073	1699	321617	5393	127050067	5347
107360641	2161	617289	2357	710563	9787
543333	4889	696161	2161	128027831	11161
108596933	7369	998669	1459	079409	5437
870961	2609	117246949	1597	124151	2311
109052113	4993	445987	5419	468957	2927
231229	2699	959221	2053	536561	8017
316593	3697	987841	7681	665319	2383
437751	5231	118466401	1249	987429	4637
541461	6043	119118121	2729	129205781	6563
879837	2707	204809	2383	256273	739
110135821	3967	261113	4657	461617	10177
139499	6427	378351	911	524669	2939

Fig. 0.14 Extract from Lehmer's table of composite numbers

In order to compute these exponents, Lehmer applied a method "*which differed greatly from the one used by human computers*" (Lehmer, 1949, p. 301). One of these differences involves the "routine" used to select the next prime p for the exponent routine: for a human, the evident source at that time was to use existing prime number tables. However, for ENIAC it was decided that it would compute its own prime numbers as they are needed by means of a parallel prime sieve (Lehmer, 1949, p. 302):

[45] A mathematician and later one of the founders of the Association for Computing Machinery. He worked on other machine, more specifically, the IBM relay machines

[46] A logician who developed a theory of programming based on his experience with ENIAC and an itnerpolation program he set-up on it

[47] An astronomer who worked on the punched card section

[48] Note that the number of exponents computed by ENIAC allowed the extension of the composite numbers to 10^9. See below for more details

> The "next value of p" presents an interesting problem to the ENIAC. One of the requirement of the problem, dictated by the circumstances under which the problem was run, was that the ENIAC works for hours without attention. This alone prevented the introduction into the ENIAC of a list of primes p via punched cards. There were at least three other good reasons for not doing this. This meant that ENIAC should somehow compute its own values of p. To this effect a "sieve" was set up which screened out all numbers having a prime factor ≤ 47.

Now, since only numbers with factors ≤ 47 were screened out, this meant that not all ps selected by the sieve were guaranteed to be prime. To this end, an additional prime test was wired into ENIAC (all $p-1$ must be divisible by e). Finally, all "primes" were punched and manually compared with D.N. Lehmer's table of primes. Thus, even though the prime routine for ENIAC with its need for an additional machine-assisted and human test is much more involved *from the human point of view* than the method of selecting primes from a table, this was not true from the *machine's eye*: from that perspective, the prime routine that was actually implemented was much more efficient than the one we humans would use. The gain in speed was enough compensation for the additional complexity on the programming side.

Similar conclusions are drawn by the people involved with the ENIAC's computations for the H-bomb. Amongst others, it was used to explore the neutron chain reactions in fission devices in order to determine the explosive behavior of the triggering fission device in an H-bomb (the Teller-Ulam design) for starting the hydrogen fusion. In (Frankel and Metropolis, 1947) research conducted with the ENIAC on the so-called liquid-drop model of fission used to determine fission thresholds and spontaneous fission rates is described. At some point during this computation use had to be made of elliptic integrals. However, instead of using tables of elliptic integrals, Fraenkel and Metropolis decided to let the machine compute its own values even though this required more complicated programming techniques (Frankel and Metropolis, 1947, p. 916)

> In treating this problem with a desk calculator the use of a table of the first elliptic integral in this step would be far easier. However, in setting up the problem for the ENIAC the economy in program controls and programming labor of this procedure seem to us adequate compensation for the increasing computing time. It seems likely that the use of high speed calculating machines will often effect changes of this kind in the economy of calculating procedures.

This quote clearly illustrates, as was the case with the sieve, that even though the algorithms and programming involved are more involved, this is seriously compensated by the speed gained in replacing tables by algorithms. Hence, there seems to be a kind of trade-off between algorithms and tables in this context: either one decides to replace the use of a table by an algorithm resulting in a serious gain in speed and memory but with additional programming complexities, or one decides to use a table which means a loss in speed and memory but a gain in ease of use.

It was also in the context of the H-bomb computations that Ulam proposed what came to be known as the Monte Carlo method. Von Neumann described the use of this method for the exploration of neutron chain reactions as follows in a letter to Richtmyer (Metropolis, 1987, p.127):

> Consider a spherical core of fissionable material surrounded by a shell of tamper material. Assume some initial distribution of neutrons in space and in velocity but ignore radiative and hydrodynamic effects. The idea is to now follow the development of a large number of individual neutron chains as a consequence of scattering, absorption, fission and escape. [...] [A] genealogical history of an individual neutron is developed. The process is repeated for other neutrons until a statistically valid picture is generated.

Now, in order to apply the Monte Carlo method, one is in need of a source of random numbers. In the late 40s there was already some research on random numbers and by 1947, when the significance of the Monte Carlo method started to become clear, the RAND corporation started with the production of a huge table of 1,000,000 digits of random numbers produced with a kind of electronic roulette wheel. These were stored on punched cards to be used by computing machinery. At the time when these digits were published in a book, it was already understood that such table can in fact be replaced by simple algorithmic procedures which produce so-called pseudo-random numbers. Hence, it is suggested in the introduction of *A million Random Digits with 100,000 Normal deviates* (Corporation, 1955):[49]

> With the high-speed electronic computers recently developed, the storage of such tables is usually not practical and, in fact, much larger tables than the present one are often required; these machines have caused research workers to turn to pseudo-random numbers which are computed by simple arithmetic processes directly by the machine as needed

[49] A separate study of tables of random digits would actually be very interesting since they pose particular problems which are reflected in the guidelines for using them. The following quote gives an indication of this (Corporation, 1955, p. xxi): "*In any use of the table, one should first find a random starting position. A common procedure for doing this is to open the book to an unselected page of the digit table and blindly choose a five-digit number; this number with the first digit reduced modulo 2 determines the starting line; the two digits to the right of the initially selected five-digit number are reduced modulo 50 to determine the starting column in the starting line [...] Ordinarily, the table is read in the same direction as a book is read; however, the size of the table may be effectively increased by varying the direction in which it is read*"

The possibility of using such simple arithmetic processes rather than tables of random digits was also considered in the context of ENIAC (See e.g. (Metropolis, 1987)). It was used to study the random distribution of the digits of π and e (See (Reitwiesner, 1950)) and von Neumann's middle-square method,[50] one of the first pseudo-random generators used in the applications of the Monte Carlo method on ENIAC. To replace a physically generated table of random digits by an algorithm that generates pseudo-random numbers was not obvious at the time given the deterministic character of machine computations (Metropolis and Ulam, 1949, pp. 339–340):

> What is more, it is not necessary to store a collection of such [random] numbers in the machine itself, but *paradoxically enough* [m.i.] the machine can be made to produce numbers simulating the above properties by iterating a well-defined arithmetical operation.

In fact, it was well-known that for instance von Neumann's middle square method was not really a good pseudo random number generator since undetected short cycles can occur.[51] Furthermore, as von Neumann once stated it:

> Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin. For [...] there is no such thing as a random number – there are only methods to produce random numbers, and a strict arithmetic procedure of course is not such a method. [...] We are here dealing with mere "cooking recipies" for making digits; probably they can not be justified, but should merely be judged by their results.

This example illustrates how a transition from tables to algorithms was not always obvious: it not only required the development of new methods adapted to the machine, but also sometimes a new way of thinking about fundamental concepts like, for instance, randomness.

As is clear, in the context of ENIAC, the people involved consciously chose to replace the use of tables as calculating aids by algorithms. In this sense, the conclusions drawn by early table-makers such as Lowan, who concluded "*We're finished*" seem to be correct: the digital electronic general-purpose computer with its high-speed and programmability was the death knell of the table as calculating aid. However, some more detailed research into ENIAC shows that this is only true to some extent.

First of all, as the Lehmer computation illustrates, even though some tables were replaced by algorithms, this was not true for all tables: ENIAC was used to compute a table of exponents which was then used *as a calculating* aid by Emma Lehmer to extend existing tables of composite numbers. Indeed, since machine time was not a luxury at the time and the machine was not big enough to compute the table of exponents *and*, on its basis, the table of composite numbers over one weekend, additional human work was required. However, this was a laborious work which was not adapted to the huge table generated by ENIAC: even though a table of composite numbers extending to 10^9 would have been possible based on ENIAC's results, it was too much work for the human calculator to do so. In this sense, one can say that tables generated by electronic means started to become humanly impractical from the very start and there was a need for internalizing the exploration, analysis and use of such tables into the machine. This was clearly understood by Lehmer, who, after having worked with different electronic computers, wrote (Lehmer, 1966):

> Today tables can be produced in such quantities as to render their publication or even, in some cases, their inspection economically impossible. In these latter cases it is a simple matter to ask the computer to do the inspection internally. From the computer's report one can make new conjectures which may have some interest and whose proof may not be too difficult. Again, in publishing our results we need not give credit where some credit is due." (Lehmer, 1966)

And indeed, what one sees in the history of computers and numerical tables is that as more and more sophisticated techniques can be internalized into the machine, the lesser tables as calculating aids are being produced *for humans* (cfr infra). This, however, is a slow historical process which is not homogenuous for all kinds of tables and which requires a more detailed study. Indeed, to take the example of the firing tables, it is clear that they are no longer required once weapons are developed that are able to compute their own firing trajectories.

0.5.3 Function tables

The fact that machines such as ENIAC were still used to produce tables also as calculating aids, however, is just one reason why the statement of the death of the table as calculating aid is in need of more reflection. In ENIAC one also sees the further development of numerical tables used *inside* of the machine. If one is willing

[50] You take some arbitrary n-digit integer, creating an $2n$-number. You then take the middle n digits and keep repeating this process

[51] Lehmer was one of the first to observe this and proposed a variant on a much better type of generator known as linear congruential generators (Lehmer, 1951).

to extend one's concept of table to include tables that are not only for human use, then ENIAC shows how important tables actually become for the machine itself. In ENIAC such machine-tables or internal tables are embodied by the important function tables. Adèle Goldstine, in her report on the ENIAC, describes the need for such function tables as follows (Goldstine, 1946, p. 65):

> In the course of solving mathematical problems, it is often necessary, at some intermediate stage of the solution, to refer to previously prepared tables which give the values of some needed function. [O]f course, the ENIAC could in many cases be set up to generate the desired function mathematically, instead of looking it up on a table; however, this might often tie up a large number of the computing units of the machine which would be needed elsewhere in the problem. In addition to such transcendental functions, a given problem may require the use of a table of arbitrary functions representing an empirically or experimentally determined relationship for which no mathematical formula is known or readily available. The ENIAC is provided with means for storing in tabular form the above types of information, and with programming means whereby any given stored value may be looked up and transmitted to an accumulator or other arithmetical unit for further use.

As is clear from this quote, even though it was realized that mathematical tables could often be replaced by algorithms, this could not always be done. In that case, it was more opportune to use the function tables, at least, if the tables were not too big.

Fig. 0.15 shows a diagram of the function table and Fig. 0.16 an example of a function table entry. Each

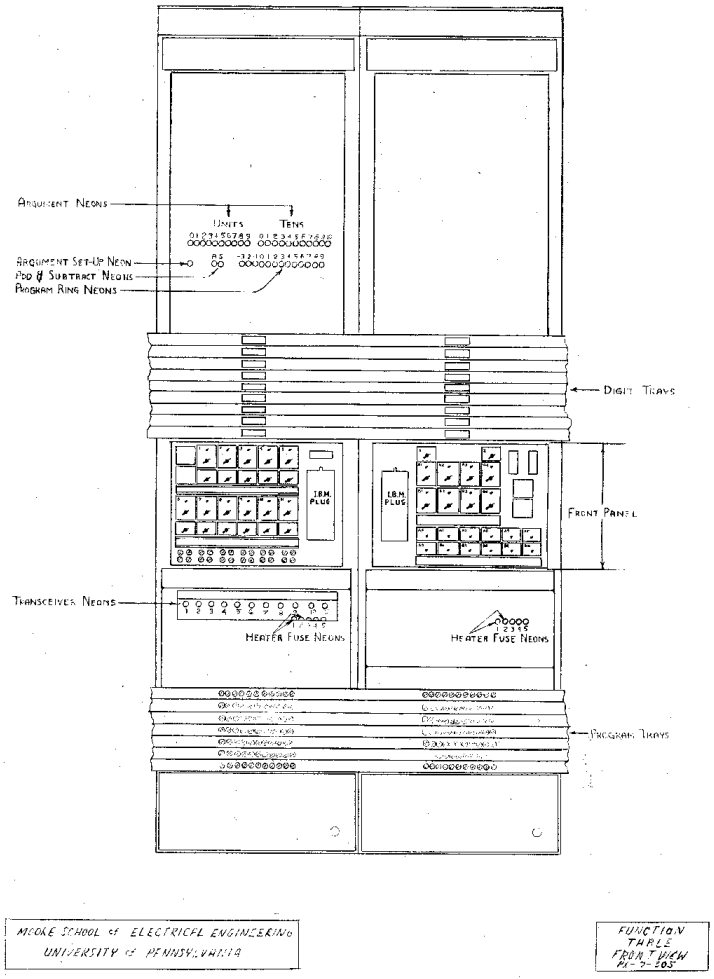

Fig. 0.15 One of the three ENIAC function tables

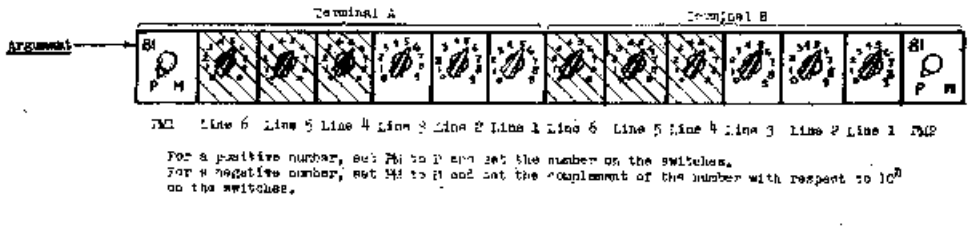

Fig. 0.16 Example of a 12-digit entry on a portable function table from ()

entry has 12 digit switches and 2 sign switches: one sign switch is to the left-end and one to the right-end of the entry. Before a computation begins, the operator had to manually switch each entry to the correct value – this is comparable to current programming practices when one is using predefined values and has to type or copy/paste them into some programming interface. As is also clear from Fig. 0.16, use of colors (black and red) was made to make it more easy for the operator to identify which digit he/she is switching. From the machine's perspective, this coloring evidently has no relevance.

Since these function tables were to be used by the machine and not by a human, these tables are *wired into* the computation. This affects the way a table functions: it not only has to store tabular values in a certain way, it also needs the capability to *transmit* these values. Furthermore, since these tables are internal to the complete machine, several programming facilities were added to manipulate the tabular values. Each function table had no less than 11 program controls (see Fig. 0.17). Each such control gives the option for the operator:

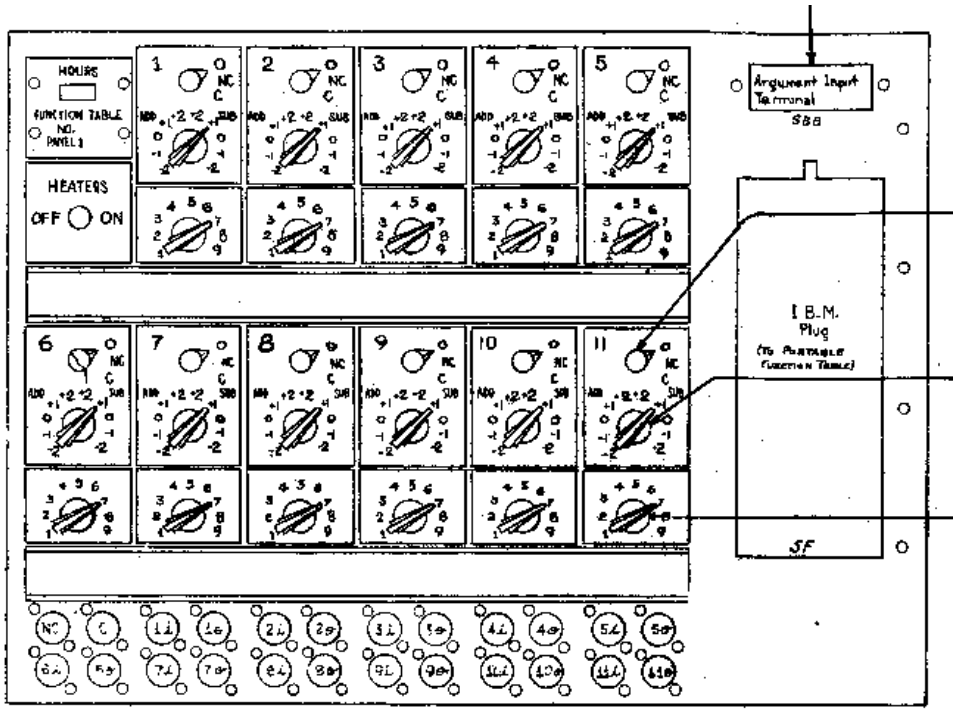

Fig. 0.17 The 11 program controls of a function table

- to transmit either the looked-up number (to add it) or its complement (to subtract it)
- to decide whether or not transmission of the argument to the function table is to be stimulated by the function table or by some other ENIAC component
- to control the number of times (from 1 to 9) a given entry should be transmitted
- given an entry x, to decide which of five entries $x-2, x-1, x, x+1, x+2$ should be transmitted

This last option is particularly interesting: it was used for interpolation of functions and shows that this option was built-in.

Since there were 11 controls for each table, this gives the possibility to manipulate the same table in several different ways during one computation, depending on what needed to be done with the tabular values during computation, what kind of accuracy was required, etc. A nice illustration of the programmability and flexibility of these function tables is the built-in option to split up a function table entry into two columns of varying size. Given a 12-digit entry plus its two signs for a given function table, the default was that the first 6 digits plus sign were sent to an output terminal A, the other 6 digits plus sign to output terminal B. This split-up however, "*into the groups A and B [is] arbitrary [...] and by means of special adaptors [...] those can be regrouped in any manner*" (Burks and Huskey, 1946). This implied that each entry could store either two numbers of varying length or just one number. Further flexibility in the number of columns was possible by using the fact that, from the machine's perspective, the number 9 corresponds also with a negative sign (represented by 9 digit pulses) and 0 with a positive sign (0 digit pulses). As a consequence, one entry could in fact store several columns (e.g. 987054012998 could be interpreted as a column containing the numbers -87, 54, 12, -98). Evidently, more complicated function table programming was required to have this functionality.

This short detour on function tables shows that numerical tables could be set-up in a variety of ways and become programmable to some extent. The function tables, as used in ENIAC were, initially, not permanently wired tables. They had to be wired and switched for every new program ran on the machine. As such they become mere carriers of mathematical tables, *data structures* as they would be called nowadays. However, it is also because of this "flexibility" and "programmability" that these tables could be used not only for different purposes (read: computations) but also in very different ways. One interesting application in this context is the use of the function tables in the permanent rewiring of the ENIAC as a stored-program computer: they were used to store the so-called "order code" developed for this purpose.[52] This rewiring[53] avoided the laborious wiring of the original ENIAC that was required for each new problem to be set-up. Instead a (highly primitive) code could be feeded to the machine by means of punched cards which could then be "interpreted" by means of the function tables. Once this rewiring was complete, one of the function tables became permanently wired up with the rest of the machine and, as such, became an essential part of the interpretation of the programs to be executed on ENIAC. In this way, mathematical tables slowly evolved from tables that store numerical values as numerical values to be used in a computation to tables that store numerical values as symbols that code instructions, the table thus becoming more and more part of an algorithm rather than being used externally by one. Viz., as the physical boundaries between the "operator" and the "operand" become more and more blurred, there can no longer be a strict separation between the "algorithm" and its (tabular) data.

The fact that a mathematical table is used internally within the computer however not only affects our notion of table but also affects the "algorithms" that use such tables. In the context of ENIAC, algorithms which use the table (for instance in an interpolation algorithm) also need to include instructions of how to use the table, when to access it, etc. This was not a trivial task and had a certain influence on the early years of "the art of programming". As explained, ENIAC was "a son-of-a-bitch" to program even though it became more easy to handle once it was rewired. Also its early descendants, like the IAS machine, were not easy to program (though not as hard as the original non-rewired ENIAC). In fact, in these early years, programming a machine often took as long or longer than the execution of the program (Goldstine and von Neumann, 1946):

> It is quite true that in existing machines the time for coding problems is comparable to the solution time, and that every effort must be made to simplify the coding of problems.

This problem of programming, is exactly the reason why people like von Neumann started to develop strategies to make the programming more easy and efficient. One particular problem here was the use of tables: if one had to select for instance the next value from a table this meant that an index in the code should be changed. This process was called *partial substitution* by von Neumann and Goldstine and understood as a fundamental feature of programming since it required (Goldstine and von Neumann, 1946):

> the machine's ability to manipulate its own code. [This is]absolutely necessary for a flexible code. Thus, if a part of the memory is used as a "function table", then "looking up" a value of that function for a value of the variable which is obtained in the course of the computation requires that the machine itself should modify, or rather make up, the reference to the memory in the order which controls this "looking up", and the machine can only make this modification after it has already calculated the vaue of the variable in question. [This] ability of the machine to modify its own orders *is one of the things which makes coding the non-trivial operation we have to view it as*. Therefore this is a quite relevant circumstance in every respect.

A similar point of view can be found in the largely unknown work of Haskell B. Curry, a famous logician who also did relevant work on ENIAC in the context of inverse interpolation. Based on his work with ENIAC, he

[52] See (Bullynck and Mol, 2010) for another example.

[53] Which, according to some, spoiled the original ENIAC

developed a theory of programming which involved, amongst others, certain conditions on this ability of the machine to manipulate its own code – the most important condition is called the *table condition* which allows for a change in the addresses during computation when it concerned looking up values in a table (See (Mol et al, 2013) for more details).

The ENIAC's function tables illustrate that, whereas one can certainly detect the start of the decline in the production of numerical tables as calculating aids for humans, this is not the case from the machine's eye. They indicate very clearly the effect of the transition from human to machine usage on tables: given the programmability of the machine, these tables become almost a kind of general-purpose devices themselves and give rise to a new variety of problems.

0.6 "Mathematical tables and other aids to computation"

The first published description of ENIAC appeared in 1946 in the journal *Mathematical Tables and other Aids to Computation* (MTAC). It would be the first of a sequence of descriptions published in *MTAC* of the new electronic and general-purpose machines being built world-wide. The journal was founded by R.C. Archibald in 1943 as chair of the *National Research Council Committee on the Bibliography of Mathematical Tables and Other Aids to computation.*[54] Its purpose was to report on developments in mathematical tables and other calculating techniques but, very soon, in the late 40s, it also started to play a key role in reporting on work being done in relation to the new computing machines being developed. It reached a relatively broad public, reflecting (Grier, 2001, p. 44):

> a broad spectrum of mathematical interests, [including amongst its subscribers] most of the major computer pioneers, including John von Neumann, Grace Hopper, J. Presper Eckert, and John Mauchly [but also] table-makers Gertrude Blanch and Herbert Salzer, the economist Wassily Leontief; the statistician John Tukey; the electrical engineer V.K. Zworkin; the inventor of the simplex method for linear programming, George Dantzig

In other words, *MTAC* is illustrative of how relevant tools of computation, including tables, had become for different milieux and *MTAC* became a venue for interaction across these different milieux. It was in fact an explicit goal of Archibald to reach a broad public, for instance, by seeking out papers written in a non-specialized language to enhance circulation (Archibald, 1943, p. 1):

> This Quarterly Journal [...] is to serve as a clearing-house for information concerning mathematical tables and other aids to computation. Especially during the past decade have tools for computation been vastly multiplied. These tools, or accounts of them, are to be found in an enormous international range of book, pamphlet, and periodical publication, not only in the fields of Pure Mathematics, Physics, Statistics, Astronomy and Navigation, but also in such fields as Chemistry, Engineering, Geodesy, Geology, Physiology, Economics, and Psychology. An attempt will here be made to guide varied types of inquirers to such material. [...] So far as practicable, the Editor will seek to have both articles and reviews written partly in non-technical language so that scholars in all fields may from such material occasionally glean something of personal advantage

In the late 40s, early 50s *MTAC* also became the preferred medium for publications of members of the Eastern Association for Computing Machinery (EACM) founded in 1947 which soon dropped the "Eastern" in its name. It was only when the ACM decided to found its own journal in 1954 that *MTAC* started to lose much of its readers: "computer scientists" now had their own specialized journal and numerical analysts could turn to journals of the *Society for Industrial and Applied Mathematics* (See (Grier, 2001)). In 1959 the last issue of *MTAC* appeared: the journal was transformed into a new journal called *Mathematics of Computation* (MoC) sponsored by the *American Mathematical Society.* As is stated in the introduction of the first issue of *MoC*, this change in title however (Polachek, 1960, p. 1):

> in no way represents a diminished interest in mathematical tables, which will continue to be a subject of major emphasis, as in the past. It recognizes, however, an increased interest in other areas in the field of *Mathematics of Computation*, which have grown in importance and in which rapid advances are being made in the present era of technological progress.

In other words, the transition from *MTAC* to *Mathematics of computation* indicates how, due to the development of the computer, attention was shifted to methods of computation *in general* rather than to one particular such tool, viz. tables. This *in general* is related to Archibald's ideal of a broad journal but also to the wide range of applications of the computer, rooted in its general-purose character (Polachek, 1960, p. 1):

> High-speed calculators are being used to translate from one language to another, to track satellites, to compose "symphonies," to design nuclear reactors, to forecast weather, to monitor this country's early warning system, to play chess, to simulate

[54] See (Grier, 2001) for more details on the formation of this committee and Archibald's contributions.

> the motion of a submarine, or to compute fall-out patterns. This is but a small sample of the many complex tasks which are being performed by these devices [...] Mathematics of Computation will in part be devoted to the exploration of new areas of applications for high- speed computer devices in every field of human endeavor.

Besides, However, this *in general* does not necessarily result in a further integration of different milieux. Rather, we see that a new discipline slowly emerges, viz. computer science, and that older disicplines, like numerical analysis, find new life with the development of the general-purpose computer. Numerical tables as calculating aids seem to steadily move into the background of this process of discipline formation and evolution. Nonetheless, one should not forget that it is not accidental that *MTAC* was an important venue for *both* numerical tables and computing machinery: it indicates how machines and tables were considered to be very strongly connected to each other. As such, this connection, as we can find it in *MTAC* is in need of further research.

0.6.1 The role of tables in MTAC (1943-1959)

From the first issue onward, *MTAC* had a very recognizable structure with regular papers reporting on new tables, computational techniques, bibliographies of tables etc but also regular sections on:

1. Recent mathematical tables
2. Unpublished Mathematical tables
3. Mathematical tables – Errata
4. Mechanical Aids to Computation

As is clear from this general structure, *MTAC* devoted much of its space to tables, but, also to mechanical aids to computation. The section on mechanical aids included reviews of important books and papers. It was renamed to *Automatic computing machinery* in 1947, only one year after ENIAC was revealed to the public and in the same year as the EACM was founded. Before that time, 30 items had been reviewed. The new section no longer consisted of reviews but included several subsections: Technical developments, Discussions, Bibliography and News. The reasons for this change are explained in an introductory note to this section (*MTAC*, vol. 2, nr. 20, 1947, p. 354):

> This new Section deals with matters pertaining to large-scale automatically-sequenced computing machinery. The wartime need for ultra high-speed calculations has caused a development of the field which *may well have a profound effect upon classification and compilation of data and of numerical computation* [m.i.].

The section was finally removed from the journal in 1955 together with the *Recent tables* and *Unpublished tables* section at a time when the journal was already suffering from a decreasing readership. Instead, a new more general section *Reviews and descriptions of tables and books* was introduced. This fusion of the several sections into one general reviewing section can be read as an indication of a steady decline of the sections on Recent and Unpublished tables. However, this does not seem to be the case. First of all, the table errata section does not disappear. In fact, the table errata only disappear in volume 35 of *MoC* in 1980. Secondly, there seems to be no clear decrease in the number of tables being reviewed just before 1955. Fig. 0.18 gives the evolution of the total number of tables described in the two sections on recent and unpublished tables. It is clear from this plot, that the number of tables being reviewed fluctuates but without there being a clear indication of a decrease. This trend does not stop after 1955. The plots of Fig. 0.19 shows the number of reviews related to tables and the number of reviews on other items[55] in the section *Reviews and descriptions of tables and books* starting from issue 49, the first issue of volume 9 (1955) and ending with the first four issues of *Mathematics of Computation*. As is clear from this plot, except for the last issue of *MTAC*, the number of reviews related to tables is much higher than the number of reviews on other items. Starting from issue 65 (vol. 13, 1959) the number of items being reviewed drops significantly and hence also the number of tables being reviewed. However, the first issues of *MoC* reach already the same level of several of the earlier issues and the trend remains that the number of table reviews is significantly larger than the number of other items reviewed. Hence, the tentative conclusion that Fig. 0.19 shows that, at least in the context of *MTAC* and *MoC*, which of course had a specific readership, the apparent decline of tables due to the general-purpose computer was certainly not immediate. However, this does not mean that tables were not affected by the computer (and conversely). We already saw that, in the case of ENIAC, for instance, *internal tables* or *digital tables* start to gain in importance. Also in *MTAC* one can detect such changes. In order to study these we restrict ourselves to the regular papers published in 1954 and 1959. The former was chosen because it is situated several years after the digital computer is introduced, the latter because it is the final year of *MTAC*.

[55] These are often books and conference proceedings on numerical analysis and computer science.

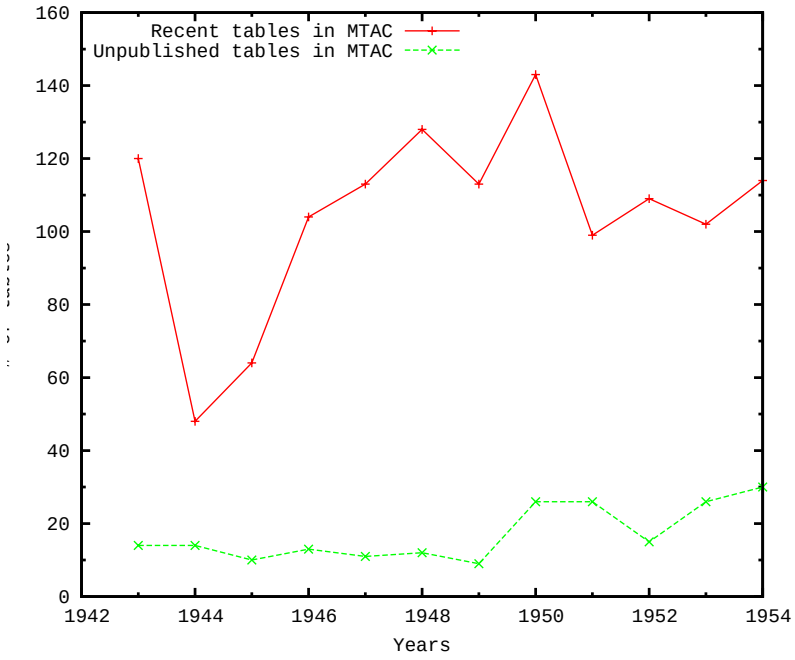

Fig. 0.18 Plot of the evolution of the number of recent and unpublished tables reviewed in *MTAC* from 1943–1955.

0.6.2 The changing role of tables in MTAC

Volume 8 (1954) of *MTAC* contains 17 regular papers, volume 13 (1959) contains 27 regular papers. Table 0.1 gives an overview of the different general topics of all papers. The category *Applied Mathematics* concerns

	1954	1959
Algorithms	7	7
Applied Mathematics	0	1
Bibliographic papers	1	0
Numerical analysis	7	16
"Pure" mathematics	1	3
Total	17	27

Table 0.1 This table gives an overview of the general topics of the regular papers published in volumes 8 and 13 of *MTAC*.

only one paper by Pasta and Ulam on numerical problems in hydrodynamics. The category of *pure mathematics* concerns papers that contain new results or discussions in fields like number theory or group theory. An example is the proof that there are exactly four projective planes of order nine, a result proven with the help of the general-purpose machine SWAC (Hall et al, 1959).[56] The *algorithms* category concerns discussions, generaliza-

[56]This proof is thus one of the first computer-assisted proofs.

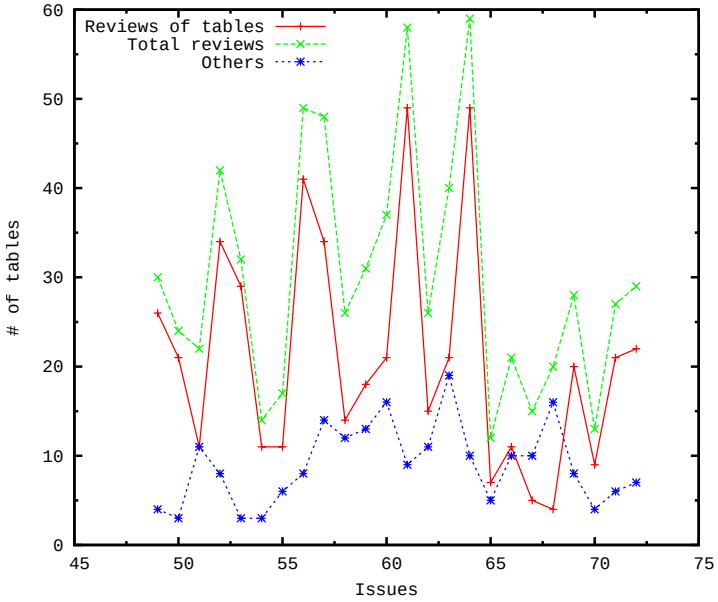

Fig. 0.19 Plots of the evolution of the number of table-related and other items reviewed in the section on *Reviews and descriptions of tables and books* The lefthand plot gives the absolute values the righthand plot the percentages.

tions and/or improvements on existing algorithms or the introduction of new algorithms in different fields of mathematics. A typical example is Daniel Shank's paper *A logarithm algorithms* in which he describes a new algorithm for computing logarithms which "*seems worth recording because of its mathematical beauty and its adaptability to high-speed computing machines*" (Shanks, 1954, p. 60) A major portion of the published papers are those in *numerical analysis*. These frequently involve discussions of algorithms but also include results on, for instance, error estimates (e.g. (Zondek and Sheldon, 1959)) or the computation of tables of coefficients (e.g. (Clenshaw, 1954)).

So how many of these papers are relevant to table-making and use? Table 0.6.2 gives an overview of the number of papers that are still related to numerical tables. It indicates that there is no dramatic decrease in the

	Machine Made		Human Made		Mix
	1954	1959	1954	1959	1959
For machine calculation	5	5	0	0	0
For human calculation	0	0	1	3	0
For human exploration	0	2	0	0	1
For illustration	1	3	0	2	0
As mathematical result	1	0	0	0	0
Total:	8	10	1	5	1

Table 0.2 This table gives an overview of the types of tables discussed in all regular papers published in *MTAC* in 1954 and 1959.

number of papers related to numerical tables between 1954 and 1959. However, Table 0.6.2 also indicates that

most of the numerical tables now involve electronic general-purpose computers.[57] One can identify two more global types of tables in Table : those that are used as a research tool (for exploration, as a mathematical result or as an illustration) and those used for calulation.

0.6.2.1 Tables for the mathematician

There are three kinds of tables here: those used as illustration, those that are in fact a mathematical result and those that are used as exploratory devices. We will not discuss in-depth these first two types of tables because they do not really affect the notion of table in any profound way. Tables used for illustration are usually tables that were computed with the machine to exemplify the algorithms being used. For instance, in one paper an algorithm for numerical integration is discussed. The table produced in this paper gives the results computed by the IBM 650 for several different functions in order to illustrate the algorithm. The one example of a table as a mathematical result concerns the machine computation of the complete character table for the symmetric group of degree 15 and 16 and was computed on the SWAC. The use of the machine is motivated by *the impractability of proceeding by hand computation*" but also by the idea that this kind of computation "*would provide a very good test of the speed and flexibility of such a computer with regard to the handling of purely algebraic problems*" (Bivins et al, 1954, p. 212).

Of more interest are the two papers by Daniel Shanks which contain number-theoretical tables for exploration (Shanks, 1959b,a). As is pointed out in Chapter **??**, explorative tables are an important category in number theory. In this context techniques of "internalization" and "compression" are often used. This development is continued with the rise of the electronic general-purpose computer. We discuss only one table here which is identified as a *statistical table* and shown in Fig. 0.20. It results from a factorization of all numbers n^2+1 from

STATISTICAL TABLE OF PRIMES AND REDUCIBLES

N	$\tau(N)$	$R(N)$	$\delta_R(N)$	$P(N)$	$\pi_-(N)$	$P(N)/\pi_-(N)$
10,000	2898	2898	.28980	841	619	1.3586
20,000	2935	5833	.29165	1559	1136	1.3724
30,000	2930	8763	.29210	2268	1633	1.3889
40,000	2918	11681	.29203	2952	2117	1.3944
50,000	2959	14640	.29280	3613	2583	1.3988
60,000	2912	17552	.29253	4252	3038	1.3996
70,000	2965	20517	.29310	4888	3485	1.4026
80,000	2881	23398	.29248	5513	3933	1.4017
90,000	2947	26345	.29272	6084	4364	1.3941
100,000	2875	29220	.29220	6656	4808	1.3844
110,000	3009	32229	.29299	7239	5247	1.3796
120,000	2934	35163	.29303	7795	5675	1.3736
130,000	2938	38101	.29308	8369	6103	1.3713
140,000	2888	40989	.29278	8944	6531	1.3695
150,000	2983	43972	.29315	9505	6941	1.3694
160,000	2952	46924	.29328	10072	7361	1.3683
170,000	2932	49856	.29327	10658	7770	1.3717
180,000	2981	52837	.29354	11223	8178	1.3723

Fig. 0.20 Shank's statistical table

$n = 1$ to 180,000 using a sieve method. The computation was done in 10 minutes on an IBM 704. However (Shanks, 1959a, p. 78):

> [s]ince this factorization of n^2+1 exceed those in existing published tables, (82 percent of these numbers are greater than a billion), a short summarizing table should be of interest.

Indeed, given the size of the actual factor table, it is replaced by a shorter table which does not give the actual factorization but a statistical summary: $P(N)$ gives the number of primes of the form n^2+1 for $1 \leq 1 \leq N$,

[57] It is not always explicitly stated in the papers whether or not general-purpose computers are involved. For those cases, we have made educated guesses for instance by checking whether an author works at an institution with access to a digital computer or by making a comparison with papers on similar topics where the involvement or non-involvement of computers is clear.

$\pi_{-}(N)$ is the number of primes of the form $4m-1$ for $1 < 4m-1 \leq N$, $R(N)$ is the number of reducible numbers[58] $\leq N$, $r(N) = R(N-10,000)$, and $\sigma_R(N) = R(N)/N$, the mean density of the reducibles. Contrary to the case of the ENIAC computation of the digits if π and e, where the statistical analysis was still done by humans, this is not the case for this table. On the basis of this sttaistical table, Shanks makes several observations and conjectures, like for instance the observation that the results are in perfect agreement with the first Hardy-Littlewood conjecture.

The resulting table is in fact a table computed on the basis of other tables which, in their turn, have been computed by the machine, like for instance the non-condensed factor table. This "table from tables" is not simply introduced for the sake of space but also because it provides the number theorist with new observations that are not necessarily evident from the original tables: the "compressed" table is not just compressed because otherwise hundreds of pages of tables would be required but also because the compression gives new statistical information which would take too much time for a human to compute.[59] Indeed, with the increase in speed of some orders of magnitude also comes an increase of some orders of magnitude in the amount of information made available up to the point that this information becomes "humanly impractical" and the machine which produces the information must be used to "inspect" the information. For this particular computation, the program was in fact rerun with a slight modification to output a table of the largest prime factors in every n^2+1: it was 360 pages long and never published! These observations are very much in agreement with von Neumann's who pointed at a human bottleneck when vast amounts of data would be produced. In fact, it was on this basis that he concluded we would no longer need very long tables of functions (von Neumann, 1966, pp. 38–39):

> [...] let me point out that we will probably not want to produce vast amounts of numerical material with computing machines, for example, enormous tables of functions. The reason for using fast computing machines is not that you want to produce a lot of information. After all, the mere fact that you want some information means that you somehow imagine that you can absorb it, and, therefore, wherever there may be bottlenecks in the automatic arrangement which produces and processes this information, there is a worse bottleneck at the human intellect into which the information ultimately seeps.

As is clear from Shank's example however, the fact that the machine can produce vast amounts of data and, at the same time, compute on/with these data does not necessarily result in the demise of the numerical table, but rather in new tables rooted in tables produced and "used" by the machine.[60]

when the technology had improved enough to allow for more graphical output, this human bottleneck was overcome by using visualizations rather than tables in certain contexts. An early example of this is the paper

0.6.2.2 Tables for calculation

In Table 0.6.2 it is clear that the majority of tables are still tables to be used as calculting aids. Of these, the majority is computed by computers to be used by computers. It is not surprising that there are still a few papers that concern human-made tables to be used as calculating aids for humans: at that time, computer access and time was still very expensive which meant that a large community of researchers was excluded from large-scale digital computation. One interesting example of this set of four papers is a paper describing an improvement on the so-called Aitken-Neville algorithm for recursively generating an n-th order Lagrange interpolation polynomial (Tweedie, 1954). It is interesting to point out that is is exactly this method that is picked up here in the context of digital and electronic computing, whereas earlier, often the method of finite differences was preferred (See Sec. 0.4.3, pp. 30). The tables in Tweedie's paper are instances of applications of the Aitken-Neville algorithm in which each column is computed on the basis of the values computed in the previous column in a manner similar to the method of finite differences. We illustrate the algorithm with an example. Assume we have a set of 4 data points $x_0, \ldots, x_3$ and we want to interpolate at some value of x. We know that the k-th Lagrange interpolating polynomial $P(x)$ for a function f defined at all points in $X = \{x_0, \ldots, x_k\}$ with x_i and x_j distinct numbers in X is given by:

[58] As Shanks explains (Shanks, 1959a, p. 82): A reducible number, r, is a positive integer whose arctangent is a linear combination, with integer coefficients, of the arctangents of smaller positive integers:

$$\tan^{-1} r = \sum_{n-1}^{r-1} a_n \tan^{-1} n$$

[59] For this particular table this would still be doable. However, in a current context in which the "tables" computed by the machine would be millions of pages if printed out, this is no longer possible practically speaking.

[60] It is worth mentioning that, later on, one actually sees that in certain contexts, exploratory tables are being replaced by visualizations. One interesting example in that respect is a paper published in *MoC* where the output are tables, but the results are based on visual patterns that are formed in the tables (Cohen, 1990).

$$P(x) = \frac{x - x_j P_{0,1,\ldots,j-1,j+1,\ldots,k(x)} - (x - x_i) P_{0,1,\ldots,i-1,i+1,\ldots,k}(x)}{x_i - x_j}$$

This interpolates f at $k+1$ distinct numbers $x_0, \ldots, x_k$. Neville's method uses this identity to generate a Neville table. Table 0.3 shows the Neville table for our example of a function with 5 data points. A modification of this

x_0	$P_0(x)$				
		$P_{01}(x)$			
x_1	$P_1(x)$		$P_{012}(x)$		
		$P_{12}(x)$		$P_{0123}(x)$	
x_2	$P_2(x)$		$P_{123}(x)$		$P_{01234}(x)$
		$P_{23}(x)$		$P_{1234}(x)$	
x_3	$P_3(x)$		$P_{234}(x)$		
		$P_{34}(x)$			
x_4	$P_4(x)$				

Table 0.3 Example of a Neville table

method was used by Tweedie to compute several examples by hand (aided by a desk calculator). Nowadays, when access to the machine is no longer a rare luxury, this method is internalized into the machine. In fact it is one of the thousands of standard subroutines in Maple.

Neville's method also has a certain similarity with the method of finite differences discussed in Sec. **??**. However, as is pointed out by Aitken, one of the advantages of the method is that with this approach we need to store less information in the tables. This allows to make tables with higher accuracy with less storage space than with difference methods for interpolation (Aitken, 1932, p. 76):

> At present it is customary [...] to print in addition to the functional values the central differences of even order, often as far as sixth differences, and to advocate the use of Everett's formula of interpolation, the coefficients of which have been computed and tabulated for values of x from 0 to 1 at intervals of 0.001 [The] methods described in this paper provide a simple, expeditious and easily controlled algorithm for interpolation, which dispenses with differences. Since the choice of a smaller interval in x almost always increases the convergence of formulae of interpolation in a decided fashion, the space devoted to printing of differences might with advantage, in our opinion, be devoted to printing the tabular values for smaller intervals.

As we will show, it are exactly these kind of memory concerns that will lead several numerical analysts to discuss different kinds of tables, for instance for interpolation, in connection with their usage by the general-purpose computer from the 50s with its small electronic memory.

The majority of table-related papers however concern tables that are computed by a machine to be used by the machine. This indicated that the use of digital or internal tables in ENIAC is not an end point but rather the start of a new type of numerical tables. The 10 papers we classified in this class are all papers in numerical analysis and concern topics like interpolation, polynomial approximation and numerical integration. As such, these tables certainly fit very well in the longer tradition of table-making in the context of approximation methods. However, the human as the computer and user of the tables is replaced by a machine. The human task is restricted to the development of methods and their implementation on the machine.

It becomes clear from these 10 papers that the motivations and methods for computing (with) such "internal" or digital tables are often rooted in problems and opportunities offered by the general-purpose computer. In Sec. 0.5.2 we already explained that the replacement of tables by algorithms was not only a possibility but also a necessity with the new machines, given the so-called memory bottleneck. By the 50s the electronic memories have been extended already significantly, but the memory problems is still there. As a consequence researchers start to develop new techniques for reducing the size of the tables much in the same spirit as Aitken proposed with his algorithm. Several of the papers involving machine-made tables to be used by the machine have exactly this concern with computer memory. A good example of this comes from a paper on polynomial approximation (Clenshaw, 1954, p. 143):

> The increasing use of high-speed computing machines has revived interest in the approximation of functions of a real variable, particularly by polynomials. An orthodox table of function values at equidistant arguments may require considerable storage space in an electronic machine. In contrast, the coefficients of a polynomial which represent the function to a desired accuracy over a specified range may require very little storage space, and simpler instructions will suffice for the evaluation of the polynomial.

As is clear from this quote, the interest in polynomial approximation has increased with the rise of the electronic computer. Indeed, rather than using tables of functions and direct methods, polynomial approximation is

considered as more suitable for electronic computers, even more so given the memory problem: indeed, this approach allows to replace tables of functions by tables of coefficients resulting in an alternative way for computing functions. Clenshaw gives, as an example, an approximation to $\sin \frac{1}{2}\pi x$ in the range $-1 \leq x \leq 1$:

$$x \sum_{n=0}^{5} A_n T_n^*(x^2)$$

The machine can then look up the coefficients in the table to determine the approximation:

$$\begin{aligned}\sin \tfrac{1}{2}\pi x \doteq\ & x\{1.276278962 - 0.285261569(2x^2 - 1) \\ & +0.009118016(8x^4 - 8x^2 + 1) - 0.000136587(32x^6 - 48x^4 + 18x^2 - 1) \\ & +0.000001185(128x^8 - 256x^6 + 160x^4 - 32x^2 + 1) \\ & -0.000000007(512x^{10} - 1280x^8 + 1120x^6 - 400x^4 + 50x^2 - 1)\} \\ =\ & 1.570796326x - 0.645964102x^3 + 0.079692704x^5 - 0.004681984x^7 \\ & +0.000160640x^9 - 0.000003584x^{11}\end{aligned}$$

A similar attitude is found in a paper describing a method of approximation using dynamic programming to deal with problems of maximizing integrals coming from the context of the calculus of variations. In the introduction to the paper, the authors explain (Bellman and Dreyfus, 1959, p. 247):

> In this note we wish to indicate some ways in which the theory of approximation can be used to increase the range of present day computers [...] *What we wish to do is to trade additional computing time, which is expensive, for additional memory capacity, which does not exist.* [...] In terms of the capacities of modern computers, we have an extremely efficient algorithm if $N = 1$, a scalar problem, and a feasible algorithm if $N = 2$. If $N = 3$ or more, we face fast memory difficulties if we attempt to proceed in a routine fashion. The reason for this is the following. To store a function of N variables in the usual way, we tabulate the values of the function at a set of lattice points within the domain of interest. If there are M different possible values of c_1, of c_2, and so on, the total number of grid points will be M^N. For $M = 100$, and $N = 3$, this yields a quantity outside of present capabilities.

One of the "tricks" the authors use to tackle this problem is again the replacement of tables of functions by tables of coefficients which are then used to approximate values of the functions (Bellman and Dreyfus, 1959, pp. 248–249). Just as in the case of the ENIAC, this replacement results in a trade-off between storage and computation, viz. between tables and algorithms. As the authors themselves point out, this trade-off results in making solvable previously unsolvable problems:

> We see that [...] problems involving four and five state variables which are completely untouchable by direct methods are within the scope of the method we have outlined. [W]e have a way of attacking previously impregnable problems

These two examples indicate that, even though tables as human aids to computation *do* start to disappear, as is witnessed by Table 0.6.2 and our observations from Sec. 0.5.2, this does not mean that tables as calculating aids in general disappear.

The ENIAC, as one of the first general-purpose electronic computers, and *MTAC*, as the leading journal on mathematical tables, are both important cases to help understand the mutliple transformations of numerical tables into the digital era. Unsurprisingly, tables that were used before as computing aids for humans start to disappear already from the early years onward and are being replaced by more and more advanced algorithms. These are more adapted to the speed of the new machines and do not require as much storage space. However, this certainly does not mean that the computer is the death knell to the numerical table. Rather than disappearing, the numerical table as calculating aid gets slowly internalized into the machine and, as such, requires the development of new techniques and technologies, both for the production and the use of tables by the machine. Important for all this is that the decision for using or not using a table rather than an algorithm depends on several technological, practical and mathematical conditions. Indeed, as becomes clear from the different examples discussed, the trade-offs between algorithms and tables depends on advances in programming technology, the computation speed gained or lost, the memory that is or is not available or the mere availability of algorithms to replace a given table.

0.7 Computer science

In the previous section we showed how the replacement of tables as a means to human computation by algorithms was often not only a possibility but also a necessity in the early years of digital computing due to the memory

bottleneck. However, as technology advanced, this memory bottleneck was slowly resolved and, as such, opened up the possibility of storing more and more data and programs inside of the machine. The significance of this steady process of internalization cannot be underestimated. Without it, we would not be able to do most of what we do nowadays with a computer, wether it is the writing of some paper in LaTeX, programming some mathematical problem or looking something up on the web. It also affects the history of the mathematical table since it allows for more extensive storage of tables inside of the computer. Today, when the significance of data storage and processing has become a major research topic in itself, one can find a rich variety of such internalized tables. One particularly interesting type of such tables are those that have a certain permanency either on the hard- or software level. These tables break with a fundamental feature of older tables: they are not intended for human use and are thus usually hidden behind several so-called levels of abstraction. Nonetheless they need to be "made" somehow and the computer needs to be told how it should use the tables. This requires one particular type of knowledge, more particularly, knowledge that would nowadays be identified as computer science knowledge. Indeed, out of the slow convergence and ultimately interaction accross different milieux (engineers, mathematicians, physicists) we see that the field of "computer science"[61] is slowly shaped as a discipline. It is within this new milieux that one develops (programming) techniques which often rely on internalized tables to be used as calculating aids by the computer.

We identify two classes of such internalized tables: tables that are used *within* an algorithm and tables that are used *as* an algorithm. Even though these two classes *cannot* be strictly separated, the classification allows to highlight some important aspects of these internalized tables. Rather than giving a multitude of examples for each class, we will instead identify and discuss one illustrative example for each class.

0.7.1 Tables in Algorithms

Nowadays, so-called look-up tables, or LUTs for short, have become a common tool in computer science. In the literature one can find several examples. Some common examples are color LUTs (CLUTs), used to convert a set of colors to a new set of colors, for instance to change the colors of a picture to warmer colors, and LUTs to compute certain common functions like logarithms or square roots. This last class of LUTs is often implemented on the hardware level.

One illustrative and (in)famous example of the use of this last class of LUTs concerns the so-called Pentium FDIV bug. This bug was discovered independently by Intel and by Th. Niceley, a number theorist who, during his work on prime gaps, discovered some inconsistencies in his results. It took him four months to find out that there was not a bug on the software level (his programs, his compiler etc) but one at the hardware level. It turned out that a LUT that was implemented on the floating-point divide unit was missing five entries. This bug resulted in quite some fuss and finally serious financial losses for Intel (Coe and et al, 1995, p. 18):

> It started with an obscure defect in the floating-point unit of Intel Corporation's flagship Pentium microprocessor. It ultimately led to a deluge of traffic on the Internet, to hundreds of reports in the mass media, to bad jokes on late-night television, to a $475 million fourth-quarter debit on Intel's balance sheet, and to a surge in sales of jewelry made from recycled silicon.

So why did Intel decide to use a table rather than algorithm for its hardware divisions? The reason is: efficiency. Indeed, in their previous design (the 486DX chip) a divisor algorithm was used which generates a quotient only at one clock cycle whereas the newer design relies on the so-called SRT algorithm which generates two quotients during one clock cycle. Hence, contrary to the early days of digital computing, nowadays, due to advances in hardware design, the use of a table is sometimes preferred over an algorithm for the sake of speed. The choice for a table-based algorithm or an algorithm without tables then really depends, on the one hand, on the local context in which a given program is going to be used, and, on the other hand, on technological conditions, viz. advances in algorithm and hardware design. In the case of the Pentium bug the needed hardware was available, resulting in high-speed access to a table at the hardware level, as well as the table-based algorithm for division. This algorithm, know as the SRT algorithm,[62] was developed from the "machine's point of view". Indeed, it would not make much sense from a human point of view to use it for ordinary division. This is clarified by the general idea behind the algorithm itself: the SRT algorithm is in fact rooted in the (human) algorithm for long division. As we all know, during long division we need to make educated guesses as to what would be the next digit of the quotient. It is exactly this educated guess which is replaced in the SRT algorithm by a table look-up (Coe and et al, 1995, p. 21–22):

[61] There has been a lot of discussion both within history and "computer science" proper on the disciplinary formation of "computer science". We therefore use the terminology "computer science" here without assuming any particular definition of the discipline or subscribing to any particular position on the scientiifc nature of "computer science". Viz. we use it here as a glossed term.

[62] S, R and T refer to the names of the people who developed the algorithm in the 50s, Sweeney, Robertson and Tocher

> What would you choose for the next digit in the quotient? The correct choice, when multiplied by the divisor, must produce a value close to, but less than, the remainder. When we do this by hand, we do not use precise algorithms for determining the digits; we use a combination of trial and error, experience, pattern matching, and luck [...] The Pentium would do the same computation using radix 4 [...] The quotient digits are obtained by table look-up in a two-dimensional array.

In other words, the use of this table-based algorithm is rooted in the fact that the educated guesses we make during long division are simply not that straightforward, if possible at all, to program as algorithms. Table look-ups are much more straightforward from the machine's point of view. Combined with the possibility to implement this table on the hardware level in a way that allows for efficient table access, the use of the SRT algorithm in Intel's chip is indeed a good example of how tables are used in algorithms nowadays. But how exactly was this LUT implemented on the hardware level?

For this particular case, the floating-point divide unit relies on a hardware device known as Programmable Logic Arrays (PLA) for the implementation of the LUT. PLAs are just one example of what is known as programmable logic devices which are nowadays standard components in computer hardware. A PLA consists of a number of prefabricated and interconnected programmable logical gates.

So what does a PLA look like? To start with, one has a number of Boolean variables which are connected to a wire and a wire with a NOT gate. Let us assume we have three variables, then we get the circuit of Fig. 0.21. These wires of the variable x_i and its negation $\overline{x_i}$ are then connected to a sequence of AND gates. If we have

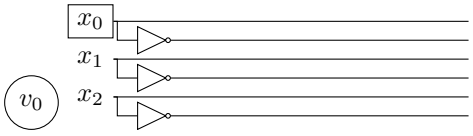

Fig. 0.21 The NOT circuitry of a PLA

four AND gates we get the circuit of Fig. 0.22. Note that a dot indicates that there is a connection between perpendicular wires. The final step is to connect the outputs of the AND gates to a sequence of OR gates. If we

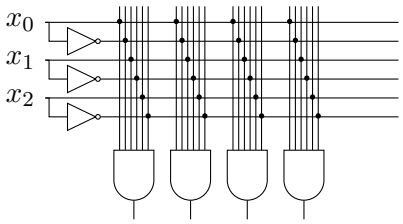

Fig. 0.22 The NOT and AND circuitry of a PLA

have four, the final PLA circuit will look like that shown in Fig. 0.23. Now, as shown in Fig. 0.23, in its initial

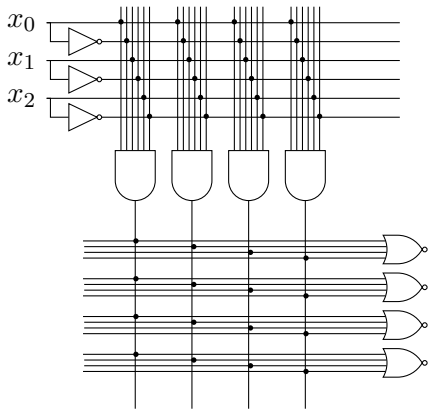

Fig. 0.23 A complete PLA circuit

state, a PLA has all of its wires interconnected so we cannot really do much with it. In order to "program" it we need to "destroy" connections. In that way, the circuit can be used to implement a truth table like the one shown in table 0.4: In order to implement table 0.4 we should destroy the right connections in Fig. 0.23 so that

x_1	x_2	x_3	F_1	F_2	F_3	F_4
1	0	0	1	0	0	1
0	1	1	0	1	1	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1

Table 0.4 An example of a truth table

we get Fig. 0.24.

Now, if we represent numbers as binary numbers then a PLA circuit can be used to "look-up" a number, viz. given a binary number p that is sent as an input to the PLA its output can again be interpreted as a binary number. For instance, in our example, there are four possible input numbers (100, 011,001, 010) that result in one of four possible outputs. By varying the number of variables, AND gates and OR gates we can increase and decrease the number of table entries. Note that, in principle, such circuits can represent any n-dimensional table. Indeed, given m different variables with $n \leq m$, the m variables can be split up into n groups.

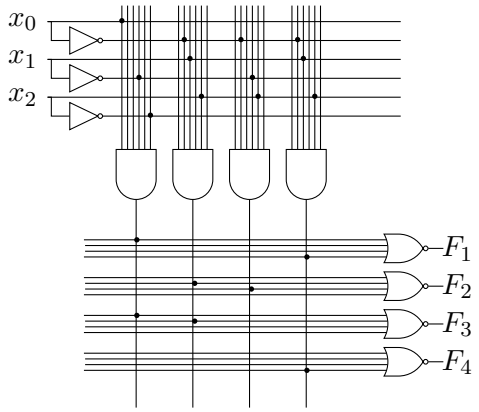

Fig. 0.24 An example of a PLA circuit

A PLA is just one example of how tables can be represented on the hardware level, viz. how internalized tables are physically represented in a machine. However, it shows how the representation of a "table" from the human point of view differs from the representation of a table from the machine's point of view. Indeed, whereas the human will "connect" the right columns and rows, the electrical signals in a machine will follow the right path of preprogrammed connections between wires to "look-up" the right value. What happened in the Pentium bug is that some values were not correctly downloaded on the PLA used as a LUT, viz. some connections were not destroyed correctly. The result was that miscomputations could occur.

0.7.2 Tables as Algorithms

In Sec. 0.15 we discussed the ENIAC function tables. Originally these tables were understood as the hardware or machine version of mathematical tables as human aids to computation. Later on these function tables were used in quite a different manner: they were no longer exclusively used to look-up numerical values to compute mathematical functions. Rather they were used to store and translate the order code of ENIAC, viz. they were used as a kind of programming device. In that sense, the ENIAC function tables were not only used *in* but also *as* a kind of algorithms.

But in what sense precisely can such tables be understood as algorithms? From a certain point of view, tables are always some kind of look-up device. Often they involve the usage of a function f of n variables, where n refers also to the dimension of the table and $f(m_1, m_2, \ldots, m_n)$ returns the tabular value indexed by $m_1, m_2, \ldots, m_n$ (See Dominique's chapter). Hence, when utilized in a computer, it seems that tables must always be tables used *in* algorithms rather than *as* algorithms. Still, from another point of view, some tables seem to be more

"algorithmic" than others (see e.g. Chapters ?? and ??, but also Sec. 0.14, p. 36). Also within computer science one can find many instances of tables which, even though essentially speaking they are still a kind of look-up table, have features that make them more algorithmic. In order to highlight such tables, we will focus in the remainder of this section on compilers.

A compiler is the basic technology underpinning human-machine communication. It is the device which translates our commands, expressed in a language which is (assumed to be) easier to use for humans, to machine language. Since the machine does not really "understand" but merely manipulates 0s and 1s one essential technique of compilers is that they are capable to interpret operations and/or instructions as data. As such, the interchangeability between operators and operand lies at the core of compiling technology and it is not surprising that tables play an important role during compiling.

A compiler works in two stages: the *stage of analysis* and the *stage of synthesis*.[63] In the first stage of analysis, the source program – the instructions coming from the user of the machine – is broken up in its constituent parts and a grammatical structure is imposed on them. It is also during this stage that the *symbol table* is constructed: it collects information from the source program (for instance, about the different variable names used in the source program). During synthesis, the machine constructs the target program, viz. the instructions in machine language, from the intermediate representation constructed during analysis.

Each of these two stages is further subdivided into several subphases. Fig. 0.25 gives an overview of a typical decomposition into phases.[64] In the remainder we will focus on the analysis phase, more specifically on the construction of the so-called symbol table and on the use of parsing tables during syntactic analysis.

0.7.2.1 The symbol table

The first type of table we want to discuss here is the symbol table which is constructed during the analysis phase. Without going too much into the details, one could say that the symbol table maintains a record about the various variable names used in some source program and their attributes. These so-called attributes can give information (Aho et al, 1986, p. 11):

> about the storage allocated for a name, its type [e.g., whether it is a number, a character, etc], its scope (where in the program its value may be used), and in the case of procedure names, such things as the number and types of its arguments, the method of passing each argument (for example, by value or by reference), and the type returned.

Hence, the symbol table is an essential part of the compiler.

By way of a small example, let us look at the kind of information that is stored in the symbol table during lexical analysis. Lexical analysis is the first phase of the analysis. It reads as an input the source program as a character stream and groups these characters in so-called lexemes. For each lexeme, a token is produced as an output:

$$\langle \texttt{token} - \texttt{name}, \texttt{attribute} - \texttt{value} \rangle$$

The token-name is an abstract symbol used during syntax analysis. Examples are: `number, if, then, id`. The attribute value is optional. It is only used when a symbol table entry needs to be created. In that case, it becomes a pointer to the table entry for that particular token-name. Let us look at an example:

$$\texttt{position} = \texttt{initial} + \texttt{rate} * \texttt{60} \tag{0.2}$$

The character stream of this program will be regrouped by the lexical analyzer into the following lexemes `position, =, initial, +, rate, 60` which, in their turn, will be mapped into the following token stream:

$$\langle \texttt{id}, 1 \rangle \ \langle = \rangle \ \langle \texttt{id}, 2 \rangle \ \langle + \rangle \ \langle \texttt{id}, 3 \rangle \ \langle * \rangle \ \langle \texttt{60} \rangle$$

The lexemes `position, initial, rate` are so-called identifiers (token-name **id**). Identifiers are the most important example of tokens that need attributes and need to be stored in the symbol table because a great deal of information is associated with them like their lexeme, their type, the location at which they were first found etc. which is needed during synthesis. The different attributes are tabulated in the symbol table and an entry is, usually, created for each identifier during lexical analysis. The token-attribute of a given identifier can then be used as an indirect reference to its table entry (see below).

[63] Many of the examples and much of the explanations that follow come from the so-called dragon book, viz. the classic book on compilers: (Aho et al, 1986)

[64] Note that this is but one possible decomposition. As noted in (Aho et al, 1986, p. 10), *[i]n practice several phases may be grouped together [...] and the intermediate representations between the grouped phases need not be constructed explicitly.*

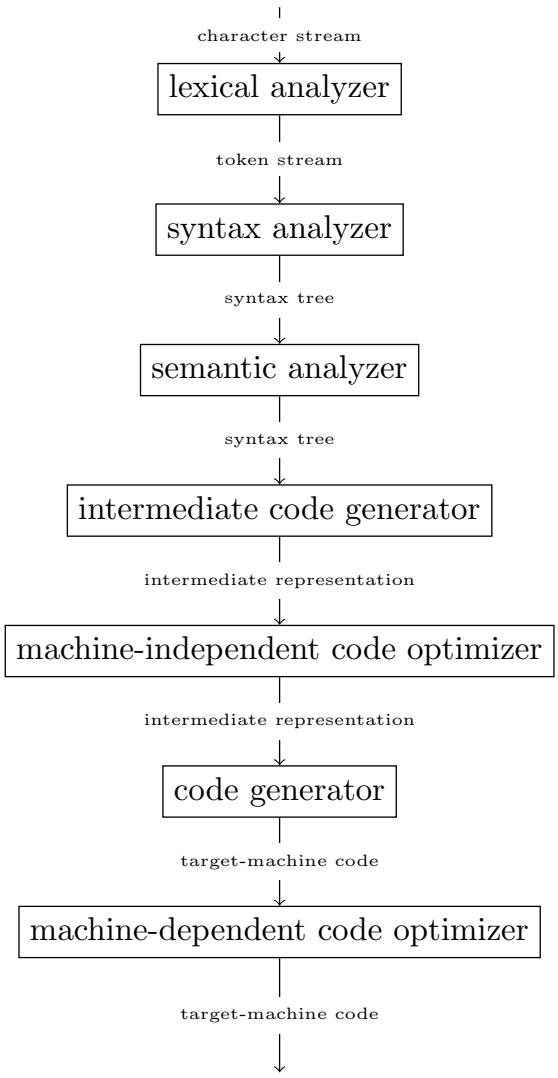

Fig. 0.25 The several phases of a compiler (Aho et al, 1986)

Explained in this way, a symbol table containing, amongst others, information about the identifiers used in a source program, is just another example of a look-up table used by some algorithm, in this case, the compiler. However, since time is really of the essence when compiling, it is very important that these tables are organized as a particular data structure which "*allows us to find the record for each identifier quickly and to store or retrieve data from that record quickly*" (Aho et al, 1986, p. 11).[65] The usual data structure for symbol tables are hash tables.

So how does a hash table function? We start out with a table consisting of two columns and m rows. The first column is a numbered list from 1 to m, the second is a list of empty pointers, viz. a data type which, rather than directly referring to some value, refers to the address of that value. So how do we tabulate? This is done by using the hash function h, usually the function $x \bmod m$ where x is a numerical value representing our names. In order to compute x for a given token-name, one usually converts the character string of a given lexeme l (for instance, `rate`) to a numerical value. A simple technique is to add up the numerical values of the characters (their ASCII codes) of a lexeme. In the case of `rate` we get: $x = 114 + 97 + 116 + 101 = 428$. Given $h(x)$ we insert the token name for the lexeme and its attributes, including the lexeme, at location $h(x)$, or, more precisely, we use the pointer at location $h(x)$ to refer to a location where x is then stored together with

[65] Data structures, which are fundamental to computer engineering, are a kind of method to organize your data in a way that it becomes efficiently to use them. What kind of data structure you are going to use, depends on the kind of data you want to organize, the way they are generated and the way they need to be used. These structures sometimes also give you the method for searching, storing and deleting information from your data set.

an empty pointer. If at some location y there is already an entry, we simply link the new entry to that previous entry using its pointer. A hash table is thus a 2-column table, where the first column contains the hash keys as an index from 1 to m and the second a linked list of the different entries associated with that hash key.

In order to restrict the space needed by the hash table, usually pointers to the lexemes are used rather than the lexemes themselves. Viz. rather than storing the lexeme names in the hash table, a reference to another location where the original lexeme is stored is used. It is also this lexeme pointer which can then be used as the token attribute. Fig. 0.26 gives a partial representation of a hash table representation of a symbol table. Hash

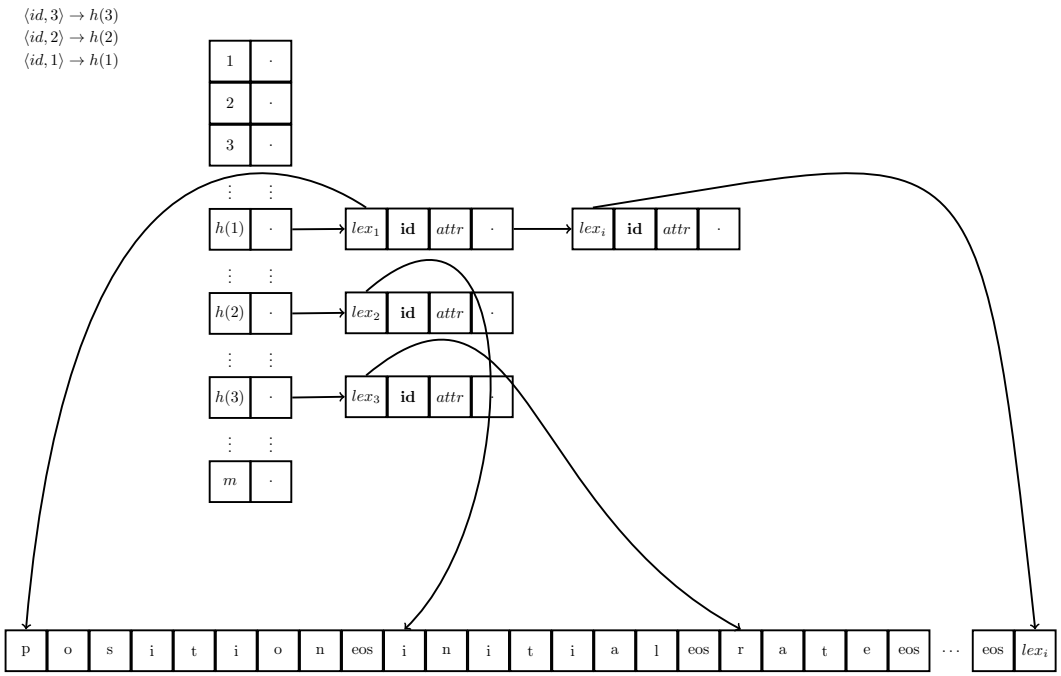

Fig. 0.26 The hash table representation of a symbol table. lex_1, lex_2, lex_3 are pointers to lexemes, $h(1), h(2)$ and $h(3)$ are the result of applying h to the different numerical values associated with the lexemes `position,initial, rate`

tables, even though they are extremely simple data structures, are chosen over, for instance, lists because they usually realize an efficient search: indeed, on the average it takes $\mathcal{O}(1)$ time to find a given item in a hash table which means that, on the average, it always takes the same number of steps to look something up in the table. The worst case however is $\mathcal{O}(m)$. This occurs when all data would be stored in the same linked list caused by choosing the wrong hash table size and function. Indeed, one such instance could occur if for some reason all the numerical values associated with our data would be odd numbers and our divisor of the hash function would be 2. This difference between worst case and average behavior shows how important it is to carefully chose the right size for the hash table as well as a good modulo function. To this end, several experiments were conducted as described in (Aho et al, 1986). On their basis, several recommendations are made.

The hash table is both a storage device and a kind of algorithm. As a storage device it is used to look-up information to be used by the compiler. As an algorithm, it contains pointers ordering and storing this information in such a way as to allow for an efficient search. Although the "ordering" of information is quite arbitrary from the human point of view, it is not from the machine's. Indeed, the position of a given value is *not* arbitrary. It is determined by a well-chosen hash function which results in a particular structure of positions which defines the efficiency of searching and inserting values in the hash table. Changing the hash table or that structure affects the efficiency of the table.

0.7.2.2 The parser table

The second phase of analysis of a compiler is called parsing or syntax analysis. During this phase, the parser takes the token stream produced by the lexical analyzer as an input and verifies whether its grammatical structure is acceptable. Usually, it also generates a syntax tree. Parsers often rely on so-called parser tables which are also a type of algorithmic table but in a different manner than the hash table.

In order to define a parser one first needs to define a grammar. Without going too much into the details, a (context-free) grammar is defined by a set of production rules, for instance:

$$
\begin{array}{ll}
(1)\ E \to & E + T \\
(2)\ E \to & T \\
(3)\ T \to & T * F \\
(4)\ T \to & F \\
(5)\ F \to & (E) \\
(6)\ F \to & \mathbf{id}
\end{array}
$$

This grammar accepts all kinds of basic arithmetic expressions like $\mathbf{id} + \mathbf{id}$ or $\mathbf{id} * \mathbf{id} + \mathbf{id}$, viz. all these kinds of basic arithmetic expressions can be generated by the grammar in a finite number of steps. This can be shown by constructing parser trees. Fig. 0.27 shows a parser tree for $\mathbf{id} * \mathbf{id} + \mathbf{id}$ using the above grammar. The role

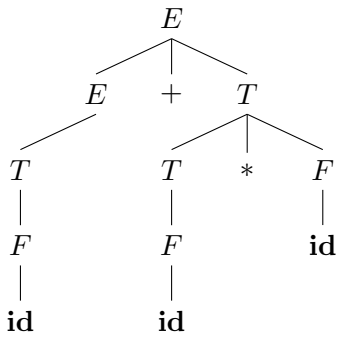

Fig. 0.27 A parse tree for the expression $\mathbf{id} * \mathbf{id} + \mathbf{id}$

of a parser is then to decide whether or not a certain sentence can be generated by a given grammar that corresponds, roughly speaking, to the syntactical structure of our programming language. For instance, given the grammar defined above, what we want our parser to do is to accept expressions like "$\mathbf{id} * \mathbf{id} + \mathbf{id}$", because it can be generated by this grammar, but reject expressions like "$* + \mathbf{id}$", because they cannot be generated by this grammar.

There are mainly three types of parsers, but we will focus on only one of them, viz. so-called LR-parsers. Such parsers work bottom-up. Roughly speaking this means it constructs a parsing tree in reverse order, starting from the bottom leaves up to the top, viz. they construct a grammar derivation in reverse order. As such, its basic operation is that of reducing a string like $\mathbf{id} * \mathbf{id} + \mathbf{id}$ to the start symbol.

So how does this work? To start, we need an empty stack and an input buffer containing the string to be parsed. The parser scans the input from left to right and shifts zero or more input symbols on the stack, possibly followed by a reduction. The parser keeps repeating this cycle of shifting and reducing until it detects an error – which means that the string cannot be produced by the grammar – or the stack contains the symbol \$ and the input buffer is empty, which means that the parse was completed successfully. A reduction occurs only if the top of the stack contains a substring which is equal to the righthand side of a production rule of the grammar.For instance, using the above example, if the top of the stack contains at some point $E + T$ then we can reduce this to E. The substring that is reduced is also called the handle.

In order for an LR parser to work properly, it also needs to keep track of states and uses so-called GOTO operations. The reason for this is that the parser needs to be able to decide when to reduce and when to shift. States are stored on the stack and it then depends on the state s_i on top of the stack, combined with the leftmost symbol of the input string a_i what the next operation – a shift, reduce, accept or error – will be. Accordingly, states on the stack need to be removed (popped) or shifted (pushed) on the stack depending on the operation performed. This works as follows. If the action performed at step i is a shift, the state s_j associated with s_i is shifted to the stack and a_i is removed from the input string. If the next operation on s_i and a_i is a reduction, then the parser executes the proper reduction. Depending on the length r of the string reduced (viz., the handle), r states are popped from the stack and the new state associated with the next symbol on the stack is pushed on the stack.

So how to program such LR parser? There are different options, but the one used is so-called table-driven: the program of an LR parser, including its different operations, are stored in a so-called parsing table. It consists

State	Action						Goto		
	id	+	∗	(	)	$	*E*	*T*	*F*
0	s5			s4			1	2	3
1		s6				acc			
2		r2	s7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6			
6	s5			s4				9	3
7	s5			s4					10
8		s6			s11				
9		r1	s7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			

Table 0.5 Example of a parser table for an LR parser

of two main columns: an ACTION column and a GOTO column. Table 0.5 gives an example of a parsing table for the grammar from page 56. In this table sx means that a shift should be executed and the new state s_{i+1} is x; rx means that a reduction should be performed using the production rule numbered x. In order to understand how this table works we apply it to our example $\mathbf{id} * \mathbf{id} + \mathbf{id}$ resulting in Table 0.6. Since, in the

	Stack	Symbol	Input	Action
(1)	0		$\mathbf{id} * \mathbf{id} + \mathbf{id}\$$	shift
(2)	0 5	**id**	$* \mathbf{id} + \mathbf{id}\$$	reduce by $F \to \mathbf{id}$
(3)	0 3	F	$* \mathbf{id} + \mathbf{id}\$$	reduce by $T \to F$
(4)	0 2	T	$* \mathbf{id} + \mathbf{id}\$$	shift
(5)	0 2 7	$T*$	$\mathbf{id} + \mathbf{id}\$$	shift
(6)	0 2 7 5	$T * \mathbf{id}$	$+ \mathbf{id}\$$	reduce by $F \to \mathbf{id}$
(7)	0 2 7 10	$T * F$	$+ \mathbf{id}\$$	reduce by $T \to T * F$
(8)	0 2	T	$+ \mathbf{id}\$$	reduce by $E \to T$
(9)	0 1	E	$+ \mathbf{id}\$$	shift
(10)	0 1 6	$E+$	$\mathbf{id}\$$	shift
(11)	0 1 6 5	$E + \mathbf{id}$	$\$$	reduce by $F \to \mathbf{id}$
(12)	0 1 6 3	$E + F$	$\$$	reduce by $T \to F$
(13)	0 1 6 9	$E + T$	$\$$	reduce by $E \to E + T$
(14)	0 1	E	$\$$	accept

Table 0.6 Computation of an LR parser on $\mathbf{id} * \mathbf{id} + \mathbf{id}$

initial state 0, the leftmost symbol is **id**, we need to shift the state 5 to the stack. We are now in state 5 with $*$ as the leftmost symbol. If we look at table 0.5, this means that we need to reduce **id** to F using rule (6) of or grammar. We can now pop state 5 which means our new state is 0. Combined with F, following Table 0.5, this means that we now need to shift the state 3 to our stack. Etc.

A parsing table is quite different from a hash table: rather than providing an algorithmic structure for storing data in an efficient manner, a parsing table contains the different possible actions of an algorithm which are accessed by means of a function f over two inputs x and y, viz., the state and the leftmost character of the input string. As such, it is the algorithm itself rather than the data it relies on that is structured by and contained in a tabular format. With this table-driven method of programming (also known as data-driven programming), the "data" stored in the table are not just a description of the state of an object (e.g. the parser) but "*actually defines the control flow of the program*" (Raymond, 2003). Indeed, the table and its content determines how the algorithm should proceed given some initial data. As such, parsing tables are a kind of programming structure. Their significance cannot be underestimated since they lie at the foundation of compiler design. Also, as indicated, the table-driven approach is certainly not restricted to compilers: it is a general programming technique which is sometimes more useful than other approaches. In the particular case of the parsing table, the table structure allows for a simple and short algorithm which replaces what would be a more "bulky" program of **if** statements.

The fact that internalized tables can be look-up tables, data structures and programming structures is rooted in one of the very foundations of computer science: the interchangeability between data and programs. It is in fact this interchangeability which makes the differentiation between tables-as-algorithms and tables-in-algorithms hard to make, since programs are in fact data inside of a computer (or vice versa). What we have seen throughout this section is that the use of tables in the computer era can certainly not be restricted to tables as data for

human use only, but that there is in fact a rich history and variety of different tables that certainly do not play a marginal role in the history of computer science. Within this context, we often saw a type of trade-off at work between tables and algorithms, the choice for using an algorithm or a table depending on the available knowledge and technology and the application for which one is programming. The fact that one has the choice is rooted in the fundamental interchangeability between data and programs which perhaps lies at the core of the reciprocity between tables and calculating machinery.

0.8 Discussion

The history of computing machinery is intricately connected with that of numerical tables. This historically developed relation between machines and tables is a complex one and cannot be captured in a one-dimensional history. Our approach here has been to study this complex relation from the perspective of the history of the machine and of the embedding of computing machines in new technical systems, involving new relations between academics and practitioners and beyond, between science and industry. From this perspective, changes in the dynamical relation between tables and machines are partially explained as being rooted in technological changes. The most fundamental such change, which is of course rooted in a more continuous history, occurs in the late 40s when the first electronic general-purpose computers are being developed. Before that time, one sees that the machines we discuss are developed and/or changed *as* table makers. However, when it becomes clear that the electronic computer can compute values of the older tables on-the-fly, this order was being reversed. Indeed, rather than that the computer undermined the need for numerical tables as calculating aids, new tables were required that could be used by the computer. Hence, today we see that tables are developed and calculated in function of computers and their changing technology. This change, from machines being built in function of tables to tables being computed in function of the machine, was reflected here in a shift of attention for the evolving machines to new types of machine tables such as LUTs or hash tables. However, even if, from one perspective, we can observe a kind of order relation between tables and machines, the relation is really a reciprocal one: even today, tables and computing machinery still go hand-in-hand.

Throughout the history of numerical tables, we see often that there is a clear differentiation between the mathematics required to compute tables and the actual computation. It is the latter which is/was delegated to human or non-human computers. This division of labor was one of the leading philosophical principles for Babbage, a principle which was so interpreted that it also reflected a hierarchical view on society where mathematics and mathematicians were considered as rightful to rule all other sciences and other kinfs of workers. However, it is also typical of this history that more and more sophisticated techniques were developed to delegate as much as possible to computers, not only to relieve the mathematician, but also to reduce the risk of error, to satisfy the ever-higher needs for more precise or longer tables, etc. These techniques are not restricted to mathematics but also involve, amongst others, advanced engineering skills. Hence, in order to mechanize the process of table-making, it is required that one overcomes a strict division of labor but that different milieux start to collaborate in such a way that the right techniques can be developed and matched. For instance, it was only when Vannevar Bush was able to resolve the problem of developing a mechanical device which could implement the feedback required in Lord Kelvin's equations (See pp. 22) that a differential analyzer could be built and, conversely, it was only when the method of finite differences was largely developed that one could start to think of building a difference engine to mechanize table making. It is this increased collaboration between different milieux – engineers, mathematicians, physicists, etc – that is in fact a conditio *sine qua non* for the development of the (first) electronic general-purpose computer(s). This is today still reflected in the field of computer science which cannot be reduced to either mathematics, physics or engineering but requires skills from each of these disciplines. Moreover, the general-purpose character of the computer has made possible its use within any discipline. Notwithstanding this plural nature of computer science, one *can* observe the changes in disciplinary fields. In this respect, the work of the renowed computer scientist Peter Denning is worth mentioning. Together with several other computer scientists, he identified three major paradigms within computer science, each being associated with its own practices: engineering, mathematics and science. They made an appeal "*on everyone to accept the three and not try to make any one of them more important than the other*" (Denning and Freeman, 2009, pp. 28–29). Hence, this appeal was rooted in a concern about a decreasing interaction accross these three paradigms and hence, a concern with a fragmented identity of the field.[66]

[66] Today, Denning went a step further and proposes to characterize computer science as one fundamentally new paradigm concerned with information-processing, alongside the physical, life and social sciences (Denning and Freeman, 2009).

Looking back into the rearview mirror of history, one can see that, even though Babbage's interpreation of the principle of division of labor as reflecting a hierarchical view on *human* society, did not suffice to force the mechanization of tables, his idea of a strict separation between work which can in principle be done by a machine, viz. repetitive work which does not require any deep skills, and work that requires the work of a mathematician, does find some resonance within the history we have been sketching: Indeed, as we have shown, a major part of the history of numerical tables and machines concerns the developments of techniques to internalize as much as possible inside of a "brainless" machine which merely follows instructions. The internalization of the addition process and the automation of the printing process in Babbage's machines; the internalization of the feedback process in the differential analyzer; the automation of part of the programming process in the development of compilers are all major steps not only in the history of the numerical table but in a wider history of algorithmisation and mechanisation of what used to be human skills. Today, the class of tables that are being used by the computer as calculating aids are in fact no longer visible to the user of the program that relies on these tables. This hiding is not in and by itself a problem. But it can become one, if one delegates trust over what happens behind the user interface to political and private establishments. For mathematics, this problem is already quite apparent in the only rarely porblematized use of licenced software like Maple and Mathematica. One can only hope that by writing histories of computing machinery, at least some of what also risks at becoming a hidden history, comes to the fore again.

References

Aho AV, Lam MS, Sethi R, Ullman JD (1986) Compilers. Principles, techniques and tools, second edition, 2007 edn. Pearson Education

Aitken A (1932) On interpolation by iteration of proportional parts, without the use of differences. Proceedings of the Edinburgh Mathematical Society, ser 2 3:56–76

Alt F (1972) Archeology of Computers – Reminiscences, 1945-1947. Communications of the ACM 15(7):693–694

Anonymous (1821) Inaugural address of the society, explanatory of their views and objects (circulated prior to their first public meeting). Memoirs of the Astronomical Society 1:285–294

Arbogast LFA (1800) Du Calcul des Dérivations. Airy Papers, Strasbourg

Archibald RC (1943) Introductory. Mathematical tables and other aids to Computation 1(1):1–2

Ashworth W (1821) The calculating eye : Baily, herschel, babbage and the business of astronomy. The British Journal for the History of Science 1:409–441

Aubin D (2009) Observatory mathematics in the nineteenth century. In: Oxford Handbook for the History of Mathematics, Oxford, pp 273–298

Babbage C (1813) Preface. vol 1, Longmans and Green, London, also in Babbage (1989), vol. 1, pp. 44-68

Babbage C (1815) An essay on the calculus of functions, part i. vol 105, London, pp 489–523, also in Babbage (1989), vol. 1, pp. 93-193

Babbage C (1816) An essay on the calculus of functions, part ii. vol 106, London, pp 179–256, also in Babbage (1989), vol. 1, pp. 93-193

Babbage C (1823) On the theoretical principles of the machinery for calculating tables. In: Edinburgh Philosophical Journal, vol 8, London, p 1128, also in Babbage (1989), vol. 2, pp. 38-43

Babbage C (1832) On the economy of machinery and manufactures. David Knight, London, also in Babbage (1989), vol. 8

Babbage C (1857) On tables of the constants of nature and arts. The Annual Report of the Board of Regents of the Smithsonian Institution for 1856 pp 289–302, also in Babbage (1989), vol. 4, pp. ???–???

Babbage C (1989) The Works of Charles Babbage, vol 11 volumes. William Pickering, London

Baily F (1823) On mr babbages new machine for calculating and printing mathematical and astronomical tables. In: Edinburgh Philosohical Journal, vol 16, London, pp 46, col. 409–422, also in Babbage (1989), vol. 2, pp. 44-68

Bellman R, Dreyfus S (1959) Functional approximations and dynamic programming. Mathematical tables and other aids to Computation 13(68):247–251

Bivins R, Metropolis N, Stein P, Wells M (1954) Characters of the symmetric groups of degree 15 and 16. Mathematical Tables and Other Aids to Computation 8(48):212–216

Bonneuil C, Joly PB (2013) Sciences, techniques et sociétés. La Découverte, Paris

Bromley AG (1987) The evolution of babbage's calculating engines. Annals of the History of Computing 9:113–136

Bromley AG (2006) The functional algorithms and control of charles babbages analytical engine circa 1838. Technical Report, december 1980 167

Brown E (912) Bulletin of the American Mathematical Society pp 365–368

Bullynck M, Mol LD (2010) Setting-up early computer programs: D. H. Lehmer's ENIAC computation. Archive for Mathematical Logic 49(2):123–146

Burks AW, Burks AR (1981) The ENIAC: First General-Purpose Electronic Computer. IEEE Annals for the history of computing 3(4):310–399

Burks AW, Huskey HD (1946) ENIAC operating manual. Tech. rep., Moore School of Electrical Engineering, University of Pennsylvania
Bush V (1931) The differential analyzer. A new machine for solving differential equations. Journal of The Franklin Institute 212:447–488
C Babbage A (1843) Addition to the memoir of m. menabrea on the analytical engine. vol 23, London, pp 23–291, also in Babbage (1989), vol. 3, pp. 83–88
Campbell-Kelly M, Croarken M, Flood R, Robson E (2003) The history of mathematical tables. From Sumer to Spreadsheets. Oxford University Press
Cartwright D (1999) Tides, A Scientific History. Cambridge University Press, Cambridge
Clenshaw C (1954) Polynomial approximations to elementary functions. Mathematical Tabels and Other Aids to Computation 8(47):143–147
Coe T, et al (1995) Computational aspects of the Pentium affair. IEEE Computational science and engineering 2:18–31
Cohen GL (1990) On an integer's infinitary divisor's. Mathematics of Computation 54(189):395–411
Colebrooke H (1825 (1822)) Address on presenting the gold medal of the astronomical society to charles babbage. In: Edinburgh Philosophical Journal, vol 1, London, pp 509–512, also in Babbage (1989), vol. 2, pp. 57-60
Comrie L (1925) The application of calculating machines to astronomical computing. Popular Astronomy 3:243–246
Comrie L (1928a) On the application of the brunsviga-dupla calculating machine to double summation with finite differences. Monthly Notices of the Royal Astronomical Society LXXXVIII(5):444–459
Comrie L (1928b) On the construction of tables by interpolation. Monthly Notices of the Royal Astronomical Society LXXXVIII(6):506–523
Comrie L (1932a) The nautical almanac office burroughs machine. Monthly Notices of the Royal Astronomical Society 92:523–541
Comrie L (1932b) Monthly Notices of the Royal Astronomical Society 92:694–707
Comrie L (1946a) The application of commercial calculating machines to scientific computation. Mathematical Tables and Other Aids to Computation 2(6):149–159
Comrie L (1946b) Babbage's dream comes true. Nature 159:567–568
Corporation R (1955) A million Random Digits with 100,000 Normal deviates, republished in 2001 by rand corporation edn. The Free Press
Davis M (2001) Engines of Logic. Mathematicians and the origins of the computer. W.W. Norton and Company, New York, London, published in hardcover as "The Universal Computer. The Road from Leibniz to Turing".
Denning PJ, Freeman PA (2009) The profession of it. computing's paradigm. Communications of the ACM 52(12):28–30
Dubbey JM (1978) The Mathematical Work of Charles Babbage. Cambridge University Press, London
Durand-Richard MJ (1990) Revue d'Histoire des Sciences XLIII(2-3):129–180
Durand-Richard MJ (1996) L'Ecole Algébrique Anglaise : les conditions conceptuelles et institutionnelles d'un calcul symbolique comme fondement de la connaissance. In: C Goldstein JR J Gray (ed) L'Europe mathématique - Mythes, histoires, identités, Editions de la Maison des sciences de l'homme, Paris, pp 445–498
Durand-Richard MJ (2000) Logic versus algebra : English debates and boole's mediation. In: Gasser J (ed) Anthology on Boole, Kluwer Academic Publishers, Dordrecht, pp 139–166
Durand-Richard MJ (2010) Planimeters and integraphs in the 19th century, before the differential analyzer. Nuncius XXIV(6):101–124
Durand-Richard MJ (2011) Le regard français de charles babbage (1791-1871) sur le déclin de la science en angleterre. Paris, pp 287–304
Durand-Richard MJ (2012) 8(2):101–124
Durand-Richard MJ (2014a) Paris, pp ???–???
Durand-Richard MJ (2014b) Historiographie du calcul graphique. ???, Paris, pp ???–???
Eckert W (1940) Punched card methods and scientific computation. Thomas J. Watson Astronomical Computing Bureau, Columbia University, New York
Frankel S, Metropolis N (1947) Calculations in the liquid-drop model of fission. Physical Review 72(10):914–925
Fritz W (1994) ENIAC – a problem solver. IEEE Annals for the history of computing 16(1):25–45
FRPhilips, Jackson J, Know-Shaw H (1928) Meeting of the Royal Astronomical Society, friday, 1928 march 9. The Observatory, A Monthly review of Astronomy LL(647):105–118
Gille B (1978) Prolégomènes une histoire des techniques. In: Gille B (ed) Histoire des Techniques, Gallimard, La Pléiade, Paris

Goldstine A (1946) Report on the ENIAC, technical report I. Tech. rep., Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia
Goldstine H, Goldstine A (1946) The electronic numerical integrator and computer (ENIAC). Mathematical Tabels and Other Aids to Computation 2(15):97–110
Goldstine H, von Neumann J (1946) On the principles of large scale computing machines. In: Taub A (ed) John von Neumann, Collected Works, 1961-63, Pergamon Press, vol 5, pp 1–32, lecture Manuscript, 1946
Grattan-Guinness I (1992) Charles babbage as an algorithmic thinker. Annals of the History of Computing 14(3):34–48
Grattan-Guinness I (2003) The computation factory : de prony project for making tables in the 1790s. In: Campbell-Kelly et al (2003), pp 105–121
Grier D (2003) Table making for the relief of labour. In: Campbell-Kelly et al (2003), pp 265–292
Grier DA (2001) The rise and fall of the committee on mathematical tables and other aids to computation. IEEE Annals of the History of Computing 23(2):38–49
Grier DA (2005) When Computers where Human. Princeton University Press, Princeton and Oxford
Haigh T, Priestley M, Rope C (2014) Reconsidering the stored-program concept. IEEE Annals for the history of computing 36(1):4–17
Hall M, Swift J, Killgrove R (1959) On projective planes of order nine. Mathematical Tables and Other Aids to Computation 13(68):233–246
Hartree DR (1935a) The differential analyser. Nature 135:940
Hartree DR (1935b) The differential analyser. National Archives for the History of Computing, Manchester University D1:1–7
Hartree DR (1946) The application of the differential analyser to the evaluation of solutions of partial differential equations. In: Canadian Mathematical Congress, Montréal, pp 327–337
Herschel J (1813a) On equations of differences, and their application to the determination of functions from given conditions. Memoirs of the Analytical Society 1:65–114
Herschel J (1813b) On trigonometrical series, particularly those whose terms are multiplied by the tangents, cotangents, secants, &c., of quantities in arithmetical progression, together with some singular transformations. Memoirs of the Analytical Society 1:33–64
Herschel J (1820) Collection of Examples of the Applications of the Calculus of Finite Differences. Deighton and Sons, Cambridge
Hyman A (1983) Charles Babbage, Pioneer of the Computer. Princeton University Press, New Jersey
Lacroix S (1816) Translated by G. Peacock, J.F.W. Herschel and C. Babbage, with appendix and notes. Deighton and Sons, Cambridge
cois Lacroix SF (1802) Traité des différences et des Séries, faisant suite au Traité du Calcul différentiel et du Calcul intégral, Duprat, Paris
Lardner D (1834) Babbage's calculating engine. vol 59, Edinburgh, pp 263–327
Lehmer DH (1949) On the converse of fermat's theorem ii. American Mathematical Monthly 56(5):300–309
Lehmer DH (1951) Mathematical methods in large scale computing units. In: Proceedings of Second Symposium on Large-Scale Digital Calculating Machinery, 1949, Harvard University Press, Cambridge, Massachussetts, pp 141–146
Lehmer DH (1966) Mechanized mathematics. Bulletin of the American Mathematical Society 72:739–750
Ligonnière R (1987) Préhistoire et Histoire des Ordinateurs. Laffont, Paris
Lovelace A (1843) Sketch of the analytical engine invented by charles babbage. Scientific Memoirs 3:666–7311, also in Babbage (1989), vol. 3, pp. 89–170
Lubet JP (2014) Le calcul aux différences et ses multiples facettes. Editions du CNRS, Paris, pp ??–??
McCartney S (1999) ENIAC. The Triumphs and Tragedies of the World's first Computer. Walker and Co., New York
Menabrea L (1842) Notions sur la machine analytique de m. charles babbage. In: Bibliothèque universelle de Genève, vol 41, Longmans and Green, Genève, also in Babbage (1989), pp. 62-82
Metropolis N (1987) The beginning of the monte carlo method. Los Alamos Science (Special Issue, Stanislaw Ulam 1909-1984) 15:125–130
Metropolis N, Ulam S (1949) The monte carlo method. Journal of the American Statistical Society 44:335–341
Mol LD, Carlé M, Bullynck M (2013) Haskell before Haskell. An alternative lesson in practical logics of the ENIAC. Journal of Logic and Computation p doi: 10.1093/logcom/exs072
Napper R (2000) The Manchester Mark 1 computers. In: Rojas R, Hashagen U (eds) The first computers: History and Architectures, MIT Press, pp 356–377

von Neumann J (1966) The General and Logical Theory of Automata. University of Illinois Press, Urbana, London
Norbert A (2003) Table making in astronomy. In: Campbell-Kelly et al (2003), pp 177–207
Observatory UN (1965) The American Ephemeris and Nautical Almanac for 1967. US Government Printing Office, Washington
Panteki M (1992) C.N.A.A., London
Polachek H (1960) Mathematics of computation. Mathematics of Computation 14(69):1–2
Polachek H (1997) IEEE Annals of the history of computing 19(2)
Raymond ES (2003) The art of Unix programming. Addison-Wesley
Reitwiesner GW (1950) An ENIAC determination of π and e to more than 2000 decimal places. Mathematical Tables and Other Aids to Computation 4(29):11–15
Sadler D (2008) A Personal History of H. M. Nautical Almanac Office, 30 October 1930-18 February 1972. United Kingdom Hydrographic Office, Sidford, Devon
Ségal J (2004) Editions Syllepse, Paris
Shanks D (1954) A logarithm algorithm. Mathematical Tables and Other Aids to Computation 8(46):60–64
Shanks D (1959a) Quadratic residues and the distribution of primes. Mathematical Tables and Other Aids to Computation 13(68):272–284
Shanks D (1959b) A sieve method for factoring numbers of the form $n^2 + 1$. Mathematical Tabels and Other Aids to Computation 13(66):78–86
Slavyanov S, Lay W, Seeger A (2000) Oxford University Press, Oxford
Smith A (1776) An Enquiry into the Nature and Causes of the Wealth of Nations. W. Strahan and T. Cadell, London
Thackray JMJ (1981) Gentlemen of Science, Early Years of the British Association for the Advancement of Science. Clarendon Press, Oxford
Thomson W (1876a) Mechanical integration of linear differential equations of the second order with variable coefficients. Proceedings of the Royal Society 24:269–271
Thomson W (1881) The tidal gauge, tidal harmonic analyser, and tidal predictor, with discussion. In: Minutes of the Proceedings of the Insitution of Civil Enginners, London, vol 65, pp 2–72
Tropp HS (1973) Ida rhodes, interview, march 21, 1973. Archives Center, National Museum of American History, Computer Oral HIstory Collection, 1969-1973
Tweedie M (1954) A modification of the Aitken-Neville iterative procedure for polynomial interpolation. Mathematical Tabels and Other Aids to Computation 8(15):13–16
Wilkins G (2003) The making of astronomical tables in h. m. nautical almanach office. In: Campbell-Kelly et al (2003), pp 295–320
Williams M (2003) Difference engines : from müller to comrie. In: Campbell-Kelly et al (2003), Oxford, pp 123–142
Wormesley DHJ (1937) A method for the numerical mechanics or mechanical solution of certain type of partial differential equations. Proceedings of the Royal Society A 161(2):454–366
Zondek B, Sheldon J (1959) On the error propogation in Adam's extrapolation method. Mathematical Tabels and Other Aids to Computation 13(65):52–55