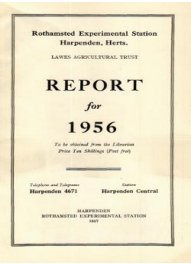
This work is licensed under a Creative Commons Attribution 4.0 International License.

Thank you for using eradoc, a platform to publish electronic copies of the Rothamsted Documents. Your requested document has been scanned from original documents. If you find this document is not readible, or you suspect there are some problems, please let us know and we will correct that.

ROTHAMSTED RESEARCH

Report for 1956

Full Table of Content


Rothamsted Experimental Station
Harpenden, Herts.
LAWES AGRICULTURAL TRUST
REPORT
for
1956

The Electronic Computer at Rothamsted

M. J. R. Healy

M. J. R. Healy (1957) *The Electronic Computer at Rothamsted ;* Report For 1956, pp 229 - 235 **-**
DOI: https://doi.org/10.23637/ERADOC-1-117

This work is licensed under a Creative Commons Attribution 4.0 International License.

THE ELECTRONIC COMPUTER AT ROTHAMSTED

By

M. J. R. Healy

Since a certain amount of mystery still surrounds electronic computers—typified by the practice of referring to them as " Giant Brains "—it may be well to start by considering in outline just what these computers do and how they do it. In brief, they are capable merely of performing the simplest arithmetical operations, such as addition, subtraction and multiplication; their essential feature is that they carry out these operations extremely fast and completely automatically.

Any computer can be divided into a number of parts. To begin with, there will be some kind of input and output devices to allow for communication with the outside world. By comparison with the speed of carrying out arithmetical operations, these devices are usually rather slow, and, to enable the machine to work at full speed, the numbers it uses as data and as intermediate results will be held in some kind of store into and out of which they can be moved at " electronic " speeds. There will be an arithmetic unit or " function box " which, on being supplied with two numbers, will produce their sum, difference or product, and some kind of control mechanism that will call up the correct numbers successively from the store and set the arithmetic unit to perform the correct functions on them.

To enable a computer to carry out a given piece of arithmetic, the latter must be broken down into a sequence of additions, subtractions and multiplications. Together these form a sequence of instructions to the machine, which is called a programme and is also held in the store. Even a simple sum, however, requires a large number of instructions, and various devices are resorted to in order to get round this difficulty. The most important of these is the machine's rudimentary power of discrimination; ordinarily, the machine obeys the instructions in a fixed order, but at given points it is able to inspect some current result and to obey one or other of two alternative instructions according as this result is zero or not (or is negative or not). Thus we might construct a sequence of instructions to carry out one cycle of an iterative approximation scheme. At the end of the sequence the machine inspects the difference between the approximation just obtained and the one obtained in the previous cycle; if this difference is not zero (to the order of accuracy required) it must return to an earlier instruction and repeat the iterative cycle, while if the difference is zero it can proceed to a different stage of the calculations.

Many calculations can be carried out by means of a repeated cycle of instructions provided that a few of these instructions can be modified slightly at each passage through the cycle. Thus to

https://doi.org/10.23637/ERADOC-1-117

This work is licensed under a Creative Commons Attribution 4.0 International License.

form the sum of a series of n numbers held in the store, the sequence of instructions must carry out the following operations :

Set total to zero

→ Pick up x_i from store ($i = 1, 2, \ldots, n$)
add x_i to total
test for end of series
No — Yes → Finish

At each passage through this cycle the " pick-up " instruction must refer to a different number in the store. An ingenious device enables the modification to be combined with a count which also provides the basis for the final discrimination. The numbers x_i are held in locations in the store which are numbered consecutively, and the " pick-up " order is so written as to refer to the first of these locations. Before it is actually obeyed, however, a count number i (starting at n) is automatically added to it in such a way as to make it refer to the ith location. At the end of each cycle, the count is reduced by one and then inspected to see whether it has reached zero; this provides the necessary discrimination, which enables the machine to leave this part of the computation and to proceed to the next.

Certain sequences of instructions are likely to be required in several different places during the course of a complete programme —for example, those for evaluating square-roots and for organizing the input and output of information. Such a sequence of instructions is known as a *sub-routine*; a sub-routine, once written, can be incorporated in any programme without the programmer having to investigate its detailed mode of operation. The usual technique is to supply the sub-routine, whenever it is entered, with a " link " instruction, which in the simplest case is merely the first instruction that is to be obeyed after the sub-routine has finished its task. This " link " is automatically stored by the sub-routine itself and ultimately referred to in order to get back into the main programme.

The N.R.D.C.–Elliott 401 Computer

The computer used in the Statistics Department has been described elsewhere (Lipton, 1955). In brief, it is a binary machine (working in the scale of 2 rather than the scale of 10 used in ordinary arithmetic) with a store containing 2,944 locations of 32 binary digits each (this is equivalent to between nine and ten decimals, and represents the accuracy to which normal arithmetic is carried out). The store is a magnetically coated disc rotating at over 4,500 r.p.m. The disc is divided into 23 tracks, each with 128 locations. Reading a number from the store takes 0·1 msecs. and this is also the time taken to carry out an addition or subtraction; multiplication takes 3·1 msecs. In addition to the main store, there are five registers, each of 32 binary digits capacity, which have immediate access— a number or instruction in the main store is only accessible once per

This work is licensed under a Creative Commons Attribution 4.0 International License.

revolution of the disc, so that in an unlucky case up to 12·8 msecs. may be wasted in waiting for the required store-location to arrive at the reading–writing station. For this reason, consecutive instructions are not placed in adjacent locations in the store, but are spread out at the will of the programmer so that, as far as possible, each order becomes available for reading as soon as the previous order has been completed. To enable this to be done, each instruction specifies the location from which the next instruction is to be drawn.

Input to the machine is by 5-hole teleprinter punched tape. This is read by a tape-reader capable of operation at up to 200 rows per second, which is comparable with the computing speed of the machine. Very recently, a punched-card input has been added. Output is either to an electric typewriter operating at 8 characters per second or to a tape-punch, which is about three times as fast; output tape can be printed out on standard teleprinter equipment away from the machine.

The arithmetic and control units of the machine are built up of some 250 plug-in units of about a dozen different types. These carry out logical operations, and the arithmetical operations are built up by connections between the plates. This packaged construction is of immense help in the day-to-day maintenance of the machine and in the tracing of faults.

Programming and coding

The first step in organizing a piece of computation for the machine is to decide on the numerical methods to be employed. These may be quite different from those which would be adopted if the same problem was to be tackled on a desk machine. A good human computer is capable of taking advantage of peculiarities in the data, and of taking special action to retain accuracy at awkward places, but he is easily bored and inevitably subject to error in copying out intermediate results. The machine, on the other hand, will happily allow rounding off and cancellation errors to swamp its results, unless the programmer has taken steps to prevent this, but will carry out repetitive tasks rapidly and accurately.

The arithmetic process has next to be broken down into a number of logical steps, each of which can conveniently be considered in isolation. This is particularly important with a complex programme through which the machine may have to steer several different paths according to the precise nature of the data. Frequently, different parts of a large programme may perform practically identical operations on different intermediate results. These can conveniently be programmed as formal sub-routines.

The final stage consists in writing the actual instructions which enable the machine to perform the required operations. On the Rothamsted computer, each instruction specifies first a source of information, which may be the store or an immediate access register; next a function, which specifies the way in which the number from the source is to be combined with the contents of a special register called the accumulator; and next the destination to which the result of the previous instruction is to be sent. A further number is used to indicate discrimination or order modification as described earlier.

This work is licensed under a Creative Commons Attribution 4.0 International License.

In addition, there is a timing number (in the form of a store address) which specifies the moment at which the order is to be obeyed; if the order contains a reference to the store, this timing address will specify the actual location required. Finally, the order contains the address in which the next instruction to be obeyed can be found. As an example, an order might read

1·52 6030 1·30

1·52, location 52 on track 1, is the address of the next order. Source 6 is the main store; the number from location 1·30 is to be added (function 0) into the accumulator, whose previous contents are to be sent to destination 3, one of the immediate access registers.

This " machine language ", though fairly complex, is sufficiently logical to be quickly mastered, and once the necessary arithmetic has been decided upon, the process of writing the appropriate orders is straightforward, if time-consuming. In practice, the logical section of the orders is written first, the addresses being filled in afterwards.

Statistical uses of the computer

The jobs handled by the computer fall into two main categories —routine tasks which, although individually simple, arise in sufficient volume to justify the writing of programmes, and more elaborate work which is too complex or time-consuming to be a practical proposition on desk machines.

In the first category the most important class of work is the analysis of the results of field experiments. These are readily handled by human computers—a fact which has contributed largely to the development of modern statistical methods in experimentation—but the volume of work involved has become considerable in recent years. Programmes are available for analysing randomized blocks (up to 126 plots), Latin squares (up to 10×10), randomized blocks with split plots (up to 128 sub-plots), 3^3 single-replicate factorial experiments and 2^n factorial experiments, which together account for some 80 per cent of all experiments reaching the department; other programmes are being developed. As an example of the speed of operation the actual analysis of a single variate takes about 15–30 seconds; input and output bring the total time up to about 2 minutes. Using these programmes, experiments involving a total of 2,500 analyses have been handled over the past two years.

A very important feature of these programmes is that they do all the work involved in an analysis. To begin with, the figures actually received from the field are very seldom those that are to be analysed; at least some conversion, say from lb./plot to cwt./acre, will be required, and often more elaborate initial computation will be called for. We use a general input routine for experimental designs which takes care of all this preliminary data-handling, places the required figures in the correct locations in the store and carries out certain checks to guard against punching errors. On the output side, the programmes are arranged to print their results in the customary form, leaving the minimum of clerical work to be done before the results can be dispatched to the experimenter.

This work is licensed under a Creative Commons Attribution 4.0 International License.

Another source of routine work is probit analysis, which arises in insecticide research and other fields. Statistically, the problem is one of weighted regression, and the computations, though straightforward, are laborious and time-consuming. One of our programmes fits a single probit line to up to 16 observed points, printing out the equation to the probit line, the ED50 and its fiducial limits; natural mortality and heterogeneity in the data can be allowed for, and the same programme can without modification use the logit or angular transformations in place of probits. A further programme fits a bivariate probit regression, or probit plane. This useful technique has hitherto been neglected in practice because of the tedium of the computations involved.

Intermediate between the routine tasks and the large-scale computational problems are the programmes dealing with multiple regression. The first stage in regression calculations is the computation of sums of squares and products, and this can easily outweigh the more sophisticated calculations that follow it. We have several programmes for this purpose that can at will provide weighted or unweighted means, sums of squares and products, variances and covariances, standard deviations and correlation coefficients. These will shortly be adapted to punched-card input, since much multivariate material is most easily made available on punched cards.

The next step in multiple-regression calculations involves the inversion of a matrix and the calculation from the inverse of the regression coefficients and their standard errors. One programme will handle up to 24 variables, and inverts a 10×10 matrix in about 8 minutes. Another, given the sums of squares and products of up to 35 variables, enables the operator to select any one of these as a dependent variable and any selection of up to 9 others as independent variates. At each stage, the operator can add a new independent variate to his regression equation, replace the last independent variate added by a different one or delete one of the earlier independent variates. In each case, the machine will recalculate the regression and print out the residual mean square with its degrees of freedom and the regression coefficients with their standard errors.

Other programmes, which, though scarcely in routine use, are general in their application, have been devised to handle various matrix operations. As an example, one programme will evaluate all the latent roots and vectors of a symmetric matrix, an 8×8 matrix taking about 5 minutes. This problem arises in general discriminant analysis and has been used in a recent study by this technique of the teeth of fossil anthropoids.

A number of special problems involving heavy computations have been tackled on the computer, and some of these may be briefly mentioned. A large body of data from the Danish pig progeny testing stations is being analysed for Dr. J. W. B. King (Animal Breeding Research Organization, Edinburgh). Several different measurements are taken on each pig, and it is required to break down the observed variances and covariances into genetically meaningful components. The estimation of linkage values from human pedigree data is being undertaken for Dr. J. H. Renwick (Galton Laboratory).

This work is licensed under a Creative Commons Attribution 4.0 International License.

It has been suggested that tsetse flies could be eradicated from an area by the release of large numbers of sterilized male flies. Practically nothing was known, however, about the numbers of flies which would be required, or about the length of time over which release should be continued. By setting up a mathematical model of the tsetse population and following it through many generations on the computer, both these points were investigated.

The feeding of dairy cattle is regulated on an empirical basis, and it is likely that the full efficiency of the cow is not being realized. K. L. Blaxter and H. Ruben (Hannah Dairy Research Institute) have approached this problem by setting up a rather elaborate mathematical model of the cow's metabolism. Fitting this model to observed data is an extremely arduous task, and is now being programmed for the computer.

There are many numerical problems which, although simple in themselves, are too time-consuming to be practical on desk machines. A typical example is the estimation of sampling errors for different patterns of sampling from completely known material. We have programmed this problem for the simple case when the sampling units form a linear sequence in space or time. The programme has been applied to data on human dietary constituents.

An obvious application of the computer is to the construction of mathematical tables, and several of the new tables in the fifth edition of *Statistical tables for biological, agricultural and medical research* were computed in this way. A large-scale project was the computation of tables of a generalized Beta distribution in collaboration with G. Foster (London School of Economics).

It may have been realized that the main difficulty in employing an electronic computer lies in the writing of programmes. The general outlines of the code of instructions for the 401 machine have been given above, and it will have been seen that programming is complicated by the " optimum access " feature—each instruction has to carry the address of the next instruction to be obeyed, and the relative placing of the instructions can have a large effect on the speed with which a programme will be carried out. In addition to this the relations to each other and to the location in which the instruction is stored of the two locations mentioned in each instruction (the timing address and the address of the next instruction) sometimes form part of the logical content of the instruction. For this reason, there are various conditions on the parities of the various locations that have to be borne in mind, and also certain delays which, if not imposed by the programmer, will cause the machine to " waste a revolution " or occasionally to perform the wrong operation. These conditions are straightforward but rather numerous, and form a potent source of errors when a programme is being written. To meet this problem, we have developed an automatic programming routine which enables this part of the programming load to be borne by the computer itself; the programmer uses a simplified coding to express the logical and arithmetic content of his instructions, and the routine fills the appropriate timing, at the same time indicating possible inconsistencies that it has detected in the simplified code. A register of occupied store locations is printed out, and an order tape punched ready to feed into the machine.

This work is licensed under a Creative Commons Attribution 4.0 International License.

This routine has been found to produce programmes whose timing is as good as or better than the average human programmer and which contain far fewer errors when they are tested for the first time.

A field of application which we propose to develop in the immediate future is the application of the computer to sample-survey analysis. Apart from its speed of operation, there are several points at which the computer is expected to contribute. To begin with, any large batch of numerical data almost inevitably contains a few gross errors. These can have a disproportionate effect on the results, especially on estimates of error; yet their discovery and elimination is usually beyond the capacity of the unaided human investigator. We have made some progress using standard punched-card equipment, but much work remains to be done. A second task for which the computer is well suited is the preliminary computation of indices and other quantities on which the actual analysis will be carried out. Then, the data on one sampling unit may specify that x tons of some mixed fertilizer was applied to a field of y acres; what is required is the amounts of N, P and K applied, in cwt./acre. The full utilization of this stage will require machinery for punching from tape on to cards. Lastly, electronic computation should lead to improvements in the summary tables of results and to greater efficiency in the methods of collecting information, since the fact that some analytical technique requires elaborate numerical work need no longer be a barrier to its use.

References

Lipton, S. (1955). A note on the electronic computer at Rothamsted. *Math. Tab., Wash.* **9**, 69.