NOTES ON THE SOLUTION OF ALGEBRAIC LINEAR SIMULTANEOUS EQUATIONS

By L. FOX, H. D. HUSKEY, and J. H. WILKINSON
(*The National Physical Laboratory, Teddington, Middlesex*)

[Received 14 October 1947]

SUMMARY

In this paper four methods of solving simultaneous equations and inverting matrices are described from the viewpoint of the practical computer. The first three methods can be performed on ordinary desk calculating machines; the fourth uses Hollerith punched card equipment. The desk methods considered are the method of elimination or 'pivotal condensation', the method of 'orthogonal vectors', and the method of Choleski. The elimination method is, with slight variants, the simple method taught at school, used in such a way that solutions are obtained with the maximum possible speed and accuracy. In the method of orthogonal vectors, applicable only to symmetric matrices, a new set of variables is chosen in such a way that the matrix is transformed to a diagonal form, from which the solution of equations or the inverse of the matrix is immediately obtainable. The Choleski method, applied here again only to symmetric matrices, expresses the square matrix as the product of two triangular matrices, the reciprocation of which is a relatively simple operation. This method is quicker and more accurate than the others and can be used, with slight modifications, in the case of unsymmetric matrices. The procedure used with punched card equipment is essentially a mechanization of the elimination method, with a considerable advantage in speed over the corresponding desk method.

1. Introduction

METHODS for the solution of algebraic linear simultaneous equations fall naturally into two classes. First, there is the indirect approach, in which the unknowns are obtained by successive approximation. Of such methods the procedure of Morris (**1**), and the relaxation method of Southwell (**2**), are best known and the most widely used. Secondly, there is the direct approach, the basic characteristic of which is the fact that any desired accuracy can be obtained in a single application of the process, provided that sufficient figures are retained throughout the computation. The best known method of this class is that known classically as the Gaussian algorithm, or more recently as the method of 'pivotal condensation'. Here the matrix is reduced to a triangular form by successive elimination of variables, and the solution is finally obtained by a process known as back-substitution. Alternatively, by extensions of the elimination process, the matrix can be reduced to a diagonal form, and the back-substitution avoided.

There are striking differences between the two types of method quoted. By the direct method the solution can always be obtained (provided it exists), and the only limit on accuracy is the number of figures to which the computer is prepared or compelled to work. Indirect methods, on the other hand, may never converge to a solution, or may converge so slowly as to be impracticable.

Again, if an indirect method converges reasonably quickly, the number of figures retained in the computation can be increased gradually as the solution improves, reducing to a minimum the labour involved. By the direct method, on the other hand, the precision of the final solution has an upper bound which is completely dependent on the number of figures retained at the first stage of the computation.† Finally, mistakes in the indirect process can merely delay the convergence to an accurate solution, while mistakes in direct methods may completely invalidate the solution obtained, so that careful checks have to be applied throughout.

There is little doubt that indirect methods at their best are quicker and less laborious than direct methods when performed on desk computing machines. They are at their best for equations in which the diagonal terms are large compared with off-diagonal terms, a state of affairs encountered often in survey problems, statistical problems, and in the numerical solution of certain types of ordinary and partial differential equations. Unfortunately there seem to be many practical problems in which the equations are of a type for which indirect methods are generally useless and direct methods must be used.

It is with direct methods that this paper is chiefly concerned. In section 2 we discuss some aspects of the method of pivotal condensation in the light of recent experience in the Mathematics Division, National Physical Laboratory. In section 3 we give the theory and computational details of two methods applicable to symmetric matrices. The first of these seemed to be new, but was subsequently found to be almost identical with the escalator process of Morris (**3**). So different is the theoretical approach that this similarity was only noticed by comparing the numerical results obtained by the two methods when applied to the example given by Morris (**3**). Our presentation is accordingly given here in full detail, with particular emphasis on the computational layout. The second method, due to Choleski, is very powerful but not very well known in this country.

In § 4 a numerical example is solved by each of the methods described in previous sections.

In section 5 we mention, without details, a method in which punched

† Though there are, of course, methods for improving the accuracy of an approximate inverse of a matrix, or the approximate solution of a set of linear equations.

card equipment is adapted for the mechanization of pivotal condensation. This method, several times as fast as the corresponding desk computation, would seem to be of considerable importance in view of the widespread demand for rapid methods of solving equations, calculating determinants, and reciprocating matrices.

The present paper is concerned with computing methods. In a companion paper (4) a mathematical analysis is given of the rounding-off errors inherent in some matrix processes. The main results of this analysis are summarized briefly in § 6. The most striking result is that previous estimates of error have been far too pessimistic, and the practice of retaining large numbers of guarding figures has been quite unnecessary.

2. Pivotal condensation

A. *Description of method*

Though this method is very well known and has often been described, the relevant information is given here for the sake of completeness, as an introduction to § 5, and for the further inclusion of details arising from recent experience in the Mathematics Division.

Briefly, the solution of simultaneous equations requires the computation of a column matrix $\mathbf{x}$, satisfying the equation

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1}$$

where $\mathbf{A}$ is a known square matrix of order n and $\mathbf{b}$ a known column matrix. Closely allied is the problem of the reciprocation of the matrix $\mathbf{A}$, involving the computation of a square matrix $\mathbf{X}$ such that

$$\mathbf{A}\mathbf{X} = \mathbf{I}. \tag{2}$$

Clearly the solution of (2) is equivalent to the solution of n equations of the type (1), in which $\mathbf{b}$ denotes successively the n columns of the unit matrix $\mathbf{I}$, and the n solutions $\mathbf{x}$ then give the n columns of the reciprocal $\mathbf{X}$ of $\mathbf{A}$.

For the solution of equation (1) the process of pivotal condensation or elimination is generally performed in such a way that the matrix $\mathbf{A}$ is transformed into a triangular matrix. The resulting n equations contain n, $n-1$,..., 2, 1 unknowns respectively, and are solved by an obvious process of 'back-substitution'.

For the computation of the reciprocal of a matrix (equation (2)) it is desirable to extend the elimination process, reducing the matrix to a diagonal form. This extended process increases the work of elimination by 50 per cent., at the expense of avoiding the back-substitution. It is therefore not worth while for the solution of one set of simultaneous equations, but undoubtedly pays for the computation of a reciprocal, or when the solution of equation (1) is required for constant **A** and several different **b**.

The simple elimination process proceeds by the following steps:

(i) Write down the matrix **A** (n rows and n columns) side by side with the matrix **b** (n rows and one or more columns, according to the number of equations to be solved).

(ii) Calculate the row sums of the whole matrix (**A**, **b**), and record them in a separate column **s**, say, which will be used to serve as a check on the computation.

(iii) Pick out the largest of the elements a_{ij} of the matrix **A**. Suppose this element, the so-called 'pivot', is a_{kl}. Calculate and record the $(n-1)$ quantities

$$m_i = \frac{-a_{il}}{a_{kl}}, \quad \text{for all } i \neq k.$$

(iv) Multiply the kth row (the 'pivotal' row) of the complete matrix (**A**, **b**, **s**) by the factors m_i, and add to the corresponding ith rows. That is, form

$$\left.\begin{aligned} a_{ij}^{(1)} &= a_{ij}+m_i a_{kj} \\ b_{ij}^{(1)} &= b_{ij}+m_i b_{kj} \\ s_i^{(1)} &= s_i+m_i s_k \end{aligned}\right\} \quad \text{for all } j \text{ and all } i \neq k. \tag{3}$$

The kth row of the new matrix $\mathbf{A}^{(1)}$ is the unchanged pivotal row of **A**, and the lth column of $\mathbf{A}^{(1)}$ has zeros in all rows except the kth.

(v) Check the computation by summing the elements of the rows of the new matrix ($\mathbf{A}^{(1)}$, $\mathbf{b}^{(1)}$). These sums should be equal, apart from rounding-off errors in the last figure, to the elements in the corresponding rows of the transformed sum $\mathbf{s}^{(1)}$.

Two courses are now possible. In the simple method, whereby the matrix is reduced to a triangular form, the kth row and the lth column are henceforth left unaltered, and are not written down again. We have in fact effectively eliminated the unknown x_l, leaving $(n-1)$ equations in $(n-1)$ unknowns. The elimination process is then continued, picking out each time the largest element of $\mathbf{A}^{(r)}$ as pivot, until only one row and column, that is, a single term, remains in $\mathbf{A}^{(n-1)}$. The n pivotal rows, containing respectively n, $n-1$,..., 2, 1 unknowns, are then assembled together, and the components of the solution or solutions **x** are obtained successively by an obvious but somewhat tedious and error-provoking process.

Alternatively, after the first elimination process, the kth row and lth column are retained in the new matrix $\mathbf{A}^{(1)}$ and treated in subsequent eliminations like any other row and column. The sequence of pivotal elements is exactly the same as before, the pivot at any stage being the largest element in the part of the matrix excluding previous pivotal rows. At the final stage each row has been used once, and once only, as pivotal row, and in the resulting matrix $\mathbf{A}^{(n-1)}$ all the elements are zero except these pivotal elements, one in each row and column. Simple division of these elements into the corresponding elements of the final matrix $\mathbf{b}^{(n-1)}$ gives immediately the solution or solutions $\mathbf{x}$ of the equations. This extension of the Gauss algorithm is usually attributed to Jordan.

A slightly different process for avoiding the back-substitution is favoured by some writers (**5**). For the solution of equation (1), the first step would be to write down the coefficients in the scheme

$$\begin{array}{c|c} \mathbf{A} & \mathbf{b} \\ \hline -\mathbf{I}_n & \mathbf{O} \end{array},$$

where $\mathbf{I}_n$ is the unit matrix of order n. Pivots are chosen from $\mathbf{A}$ in exactly the same sequence as before, and the rows of $-\mathbf{I}_n$ as well as of $\mathbf{A}$ are included in the elimination process, the simple one used for the reduction of $\mathbf{A}$ to a triangular form. Clearly at each stage one more row of $-\mathbf{I}_n$ is involved, and one more row of $\mathbf{A}$ is omitted so that, as in the previous method, the number of rows to be written down is always the same. The number of columns on the left of the vertical line is reduced by one at each stage. When the last elimination has been performed, nothing remains above or to the left of the vertical line, and the solution to the equations, or the reciprocal matrix (for $\mathbf{b} = \mathbf{I}_n$) is given in the bottom right-hand corner.†

An example performed by each of the three methods is given in § 4.

B. *Special points*

A mathematical analysis of questions concerning the error of the reciprocal matrix is given in a companion paper (**4**), summarized in § 6. The points discussed in this section are prompted by recent practical experience in the Mathematics Division of the N.P.L.

1. The choice of the largest coefficient as pivot reduces to a minimum the rounding-off error by ensuring in the simple elimination process that the factors m_i are always less than unity. In the extended processes some of the factors may be greater than unity. For the first method this can occur in the factors corresponding to previous pivotal rows, and for the second method in the factors corresponding to the rows of $-\mathbf{I}_n$. This feature is unavoidable, and occurs in a disguised form in the 'substituting-back' process.

† Because of the horizontal line separating $(\mathbf{A}, \mathbf{b})$ from $(-\mathbf{I}_n, \mathbf{O})$ we call this, colloquially, the 'below-the-line' method.

2. Since only a finite number of figures can be retained in the computation, even if the initial coefficients of the matrix $\mathbf{A}$ are known exactly, the solution obtained is only an approximation. This approximation $\mathbf{x}_0$ should always be checked by computing the residuals (from equation (1))

$$\boldsymbol{\delta} = \mathbf{b} - \mathbf{A}\mathbf{x}_0.$$

A more accurate solution $\mathbf{x}_0 + \mathbf{x}_1$ can then be obtained by finding $\mathbf{x}_1$ from

$$\mathbf{A}\mathbf{x}_1 = \boldsymbol{\delta}.$$

Most of the elimination has already been performed in the computation of $\mathbf{x}_0$, so little extra effort is required.

Somewhat similarly, a more accurate inverse can be obtained in the form $\mathbf{X}_0 + \mathbf{X}_1$, where $\mathbf{X}_0$ is an approximate solution of (2), and

$$\mathbf{X}_1 = \mathbf{X}_0(\mathbf{I} - \mathbf{A}\mathbf{X}_0).$$

This second solution is worth obtaining not only when improvement is required, but also in those cases when accuracy of a given order is required, and there is doubt about the number of correct figures in the first approximation, of which the residuals do not always give a reliable indication.

3. Very often the coefficients and right-hand sides of the equations are obtained from physical considerations and experiments, and are therefore known to a limited degree of accuracy. Consequently any solution, for which the residuals are less than the 'tolerance' of the original system, is as good as any other, and it is pointless to attempt to obtain more accurate solutions. For example, if the coefficients are known exactly, whilst the right-hand sides have an error which may be as much as 5 units in, say, the fourth decimal place, *any* solution is 'accurate' for which the residuals are less than 5 units in the fourth decimal.

4. It seems to be customary, in solving equations by elimination, to keep many more extra figures than are given in the original coefficients. In our experience this is quite unnecessary.

To fix ideas, consider the solution by simple elimination of a set of equations, working throughout with, say, six decimal places. Originally all the coefficients are less than unity and at least one in every row lies between 0·1 and 1·0. The elimination process is performed until the final pivot is reached. This pivot is written down to six decimal places, of which the last one or two figures may be incorrect, due to accumulation of rounding-off errors. (In spite of much that has been written to the contrary, it is our experience that this error is small.)

The significance of the last pivot is this, that with well-conditioned equations its value is not very much less, and may even be greater, than the first pivot. Thus it may contain as many as six *significant* figures, resulting in solutions correct almost to six significant figures. With ill-conditioned equations, this last pivot may be much smaller than the original—possibly the first four figures after the decimal point are all zeros. Consequently only a few *significant* figures are available in the final pivot, and results are certainly not obtainable to more figures.

This does not, however, mean that more accurate solutions could be obtained by taking originally more than six places. For if the original coefficients are known accurately only to six decimals, the last pivot cannot be obtained to better than six decimals, and if four significant figures were lost in the elimination, only two remain and more are meaningless.

If, furthermore, the original coefficients were known only to four decimals, and in the working to six decimals the first four figures in the final pivot were zero, then the original equations have effectively no solution, being linearly dependent. A solution obtained with the help of more figures would, as far as the physical problem is concerned, be meaningless and even misleading.

Our conclusion is then as follows. If the original coefficients are correct to n places of decimals, computations should be carried out with perhaps one extra figure for every ten equations, these extra figures being the so-called 'guarding figures' for the accumulation of rounding-off errors. Then if the first n decimals of the final pivot contains m significant figures, no answer to better than m significant figures can be obtained from the given equations to the physical problem.

These conclusions have been borne out by experiment on a set of eighteen ill-conditioned equations, whose coefficients were given to five decimals. Several guarding figures were added, and the final pivot had two significant figures in its first five decimals. The coefficients were then rounded off to four decimals, and the elimination process repeated, only one significant figure being left in the final pivot. The two solutions agreed generally to only one significant figure, bearing out the above conclusions.

3. Two methods applicable to symmetric matrices

One of the disadvantages of the elimination method, even in the most favourable case of a symmetric matrix, is that quite a substantial part of the total time is taken in recording the numbers used at the various stages. In the two methods of this section far fewer numbers are recorded, and the saving in time is considerable. Improved accuracy would also be expected, since for every recorded figure one rounding off is performed. Though they apply only to symmetric matrices, any matrix can be brought to this form by multiplication by its transpose.

The first method, which we have called 'the method of orthogonal vectors', requires a word of explanation. As stated above, this seemed to us to be new, but in fact is almost identical with the escalator process of Morris (3). So different is the theoretical approach, however, that this similarity was only noticed by comparing actual numerical results. This new approach is expressible very concisely in vector and matrix notation, and permits of a very convenient computational layout. There is also an important computational difference which is explained in the next section.

A. *The method of orthogonal vectors*

In the equation
$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1}$$
the solution $\mathbf{x}$ is a column matrix or vector. The present method consists in expressing $\mathbf{x}$ as a linear combination of n column matrices or vectors $\mathbf{x}_r$ in the form
$$\mathbf{x} = \sum_1^n \alpha_r \mathbf{x}_r. \tag{4}$$
These vectors are chosen in a special way to satisfy the 'orthogonality' condition
$$\mathbf{x}_s' \mathbf{A}\mathbf{x}_r = \mathbf{x}_r' \mathbf{A}\mathbf{x}_s = 0 \quad (r \neq s), \tag{5}$$
where $\mathbf{A}$ is necessarily symmetric. Then from (1) and (4) we have
$$\sum_1^n \alpha_r \mathbf{A}\mathbf{x}_r = \mathbf{b}. \tag{6}$$
Premultiplication by $\mathbf{x}_s'$ for all $s = 1,\ldots,n$, then gives the n equations typified by
$$\sum_{r=1}^n \alpha_r \mathbf{x}_s' \mathbf{A}\mathbf{x}_r = \mathbf{x}_s' \mathbf{b}. \tag{7}$$
Equations (7) constitute a set of simultaneous equations for the quantities α_r. By equation (5), all terms in (7) vanish for $s \neq r$, that is, the matrix of coefficients is diagonal. We have in fact
$$\alpha_r = \frac{\mathbf{x}_r' \mathbf{b}}{\mathbf{x}_r' \mathbf{A}\mathbf{x}_r}, \tag{8}$$
and the final solution from (4) is
$$\mathbf{x} = \sum_1^n \frac{\mathbf{x}_r' \mathbf{b}}{\mathbf{x}_r' \mathbf{A}\mathbf{x}_r} \mathbf{x}_r. \tag{9}$$

The orthogonal vectors are obtained as follows. Starting with the n coordinate vectors
$$\left.\begin{array}{l} \mathbf{y}_1 = \{1, 0, 0, \ldots, 0\} \\ \mathbf{y}_2 = \{0, 1, 0, \ldots, 0\} \\ \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\ \mathbf{y}_n = \{0, 0, 0, \ldots, 1\} \end{array}\right\}, \tag{10}$$
we calculate in turn the vectors $\mathbf{x}_r$, starting with $\mathbf{x}_1 = \mathbf{y}_1$. The vector $\mathbf{x}_2$ is given by
$$\mathbf{x}_2 = \mathbf{y}_2 + \lambda_2 \mathbf{x}_1, \tag{11}$$

in which, since $\mathbf{x}_2$ is to be orthogonal to $\mathbf{x}_1$, we use equation (5) and obtain

$$\lambda_2 = -\frac{\mathbf{x}_1'\mathbf{A}\mathbf{y}_2}{\mathbf{x}_1'\mathbf{A}\mathbf{x}_1} = -\frac{\mathbf{y}_2'\mathbf{A}\mathbf{x}_1}{\mathbf{x}_1'\mathbf{A}\mathbf{x}_1}. \tag{12}$$

Similarly $$\mathbf{x}_3 = \mathbf{y}_3+\lambda_3\mathbf{x}_1+\mu_3\mathbf{x}_2,$$

in which, making $\mathbf{x}_3$ orthogonal to both $\mathbf{x}_1$ and $\mathbf{x}_2$, we find (since $\mathbf{x}_1$ and $\mathbf{x}_2$ are already orthogonal)

$$\lambda_3 = -\frac{\mathbf{y}_3'\mathbf{A}\mathbf{x}_1}{\mathbf{x}_1'\mathbf{A}\mathbf{x}_1}, \qquad \mu_3 = -\frac{\mathbf{y}_3'\mathbf{A}\mathbf{x}_2}{\mathbf{x}_2'\mathbf{A}\mathbf{x}_2}. \tag{13}$$

Finally $$\mathbf{x}_n = \mathbf{y}_n+\lambda_n\mathbf{x}_1+\mu_n\mathbf{x}_2+\dots$$

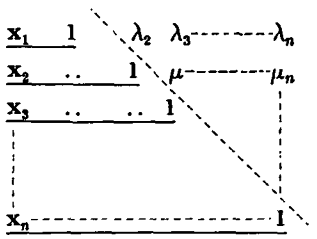
and $$\lambda_n = -\frac{\mathbf{y}_n'\mathbf{A}\mathbf{x}_1}{\mathbf{x}_1'\mathbf{A}\mathbf{x}_1}, \qquad \mu_n = -\frac{\mathbf{y}_n'\mathbf{A}\mathbf{x}_2}{\mathbf{x}_2'\mathbf{A}\mathbf{x}_2}, \text{ etc.} \tag{14}$$

These orthogonal vectors, and hence the coefficients on the left of equation (7), are independent of the constant term $\mathbf{b}$. The extension of this method to the evaluation of the reciprocal of $\mathbf{A}$ is therefore fairly easy, involving merely the computation from equation (7) of n sets of values of α for the n columns $\mathbf{b}$ of the unit matrix. The remaining calculation follows trivially from equation (4).

In practice, we regard this method as a device whereby the original matrix is transformed to an almost diagonal form. It is clearly not essential for equation (5) to hold: given any vectors $\mathbf{x}_r$ we can, in theory, set up equations (7), solve for α, and obtain the required solution from equation (4). In practice the orthogonality condition is used because it gives the simplest set of equations (7). Since we work with a finite number of figures, equations like (5) will not be satisfied exactly, and the off-diagonal terms in equation (7) may differ from zero in the number of places to which we are working. We therefore accept and record these terms, and solve the resulting equations (7) by iteration, which in this case is very rapid. For large numbers of equations, particularly when they are ill-conditioned, this treatment adds greatly to the accuracy attainable. Here we differ from the escalator process which ignores the off-diagonal terms and effectively uses equation (8) for the computation of the α's.

In the computations it is convenient to write column matrices like $\mathbf{x}_r$ and $\mathbf{A}\mathbf{x}_r$ in rows. The matrix $\mathbf{X}$ whose rows are the vectors $\mathbf{x}_r$, is then seen to be a lower triangular matrix, with units in the principal diagonal. The numbers λ, μ, etc., then fit into place in the upper half of the matrix, λ_2 to λ_n occupying the places 2 to n in row 1, μ_3 to μ_n occupying places 3 to n in row 2, etc. In this scheme the various numbers are ideally situated for the computation of successive $\mathbf{x}_r$.

The orthogonality condition (5) implies that the matrix whose rows are the column matrices $\mathbf{Ax}_r$ is an upper triangular matrix. The effective vanishing of terms below the principal diagonal is one check on the computation. A more powerful check on $\mathbf{Ax}_r$, the summation check, is


$\underline{\mathbf{x}}_1$ 1 λ_2 λ_3 λ_n
$\underline{\mathbf{x}}_2$ .. 1 μ μ_n
$\underline{\mathbf{x}}_3$ 1
$\underline{\mathbf{x}}_n$ 1

The vectors are underlined to prevent confusion with the coefficients λ, μ, etc.

obtained by calculating $\mathbf{sx}_r$, where $\mathbf{s}$ is the row formed by the sum of the columns of $\mathbf{A}$. This quantity should be effectively equal to the sum of the elements of $\mathbf{Ax}_r$.

The computations are performed in the following order, starting at the point where the rth vector $\mathbf{x}_r$ has just been formed.

1. Calculate $\mathbf{x}'_r\mathbf{Ax}_s$ for all $s < r$. This is a good check on the accuracy of $\mathbf{x}_r$, and these quantities are needed in any case, being some of the (practically zero) coefficients of equation (7).

2. Calculate $\mathbf{Ax}_r$, checking that all elements less than the rth are practically zero, and carrying out also the summation check.

3. Calculate $\mathbf{x}'_r\mathbf{Ax}_r$ (another coefficient in (7)) and the quantities of the type λ of equation (14). (Note that $\mathbf{y}'_s\mathbf{Ax}_r$ is merely the sth element of $\mathbf{Ax}_r$.)

4. Calculate $\mathbf{x}'_s\mathbf{Ax}_r$ for all $s < r$. These values should also be very small and very nearly equal to the corresponding terms $\mathbf{x}'_r\mathbf{Ax}_s$. A set of $r \times r$ coefficients in the top left-hand corner of the matrix given by equation (7) has now been computed.

5. Calculate the vector $\mathbf{x}_{r+1}$.

6. Proceed in this way, and in this order, until all the vectors $\mathbf{x}_r$ and all the coefficients $\mathbf{x}'_r\mathbf{Ax}_s$ have been obtained.

7. Calculate $\mathbf{x}'_s\mathbf{b}$ for all s.

8. Solve equation (7) for α by relaxation or iteration.

9. Calculate the final solution $\mathbf{x}$ from equation (4).

10. Calculate the residuals
$$\boldsymbol{\delta} = \mathbf{b} - \mathbf{Ax},$$
and obtain if required a more accurate solution by repeating the process from step (7) onwards with $\boldsymbol{\delta}$ in place of $\mathbf{b}$.

When the matrix is ill-conditioned, the diagonal terms $\mathbf{x}'_r\mathbf{Ax}_r$ in the equations given by (7) suffer, as did the 'pivots' in the elimination method, from loss of significant figures. In experiments with a quite ill-conditioned set of eighteen equations, however, a more accurate solution was obtained than by the elimination process, working to the same number of decimals, with a saving of time of about one-third. This increased accuracy is due to some extent to the inclusion of the small but non-zero off-diagonal terms in the equation (7), and the saving of time to the considerable decrease in the number of recorded quantities.

An example is given in § 4.

B. *Choleski's method*

Another method, due to Choleski, for the reciprocation of a symmetric matrix was pointed out to us recently by John Todd. This method is so simple in theory and straightforward in practice that its neglect in this country is surprising. The method consists simply in expressing the symmetric matrix **A** in the form

$$\mathbf{A} = \mathbf{L}\mathbf{L}', \tag{15}$$

where **L** is a lower triangular matrix and **L**′, its transpose, an upper triangular matrix. Then

$$\mathbf{A}^{-1} = \mathbf{L}'^{-1}\mathbf{L}^{-1} \tag{16}$$

and the problem is solved.

Methods for the reciprocation of a triangular matrix are given in *Elementary Matrices* (**6**). The problem merely involves the back-substitution process mentioned in § 2, for equations whose right-hand sides take successively the columns of the unit matrix. Further, $\mathbf{L}'^{-1}$ is the transpose of $\mathbf{L}^{-1}$, so that only one such process is necessary.

The computation of **L** is also simple. Consider the case of a third-order matrix. We have to find the elements α, corresponding to known elements a, for which the following matrix equation holds:

$$\begin{pmatrix} \alpha_{11} & 0 & 0 \\ \alpha_{21} & \alpha_{22} & 0 \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{pmatrix} \begin{pmatrix} \alpha_{11} & \alpha_{21} & \alpha_{31} \\ 0 & \alpha_{22} & \alpha_{32} \\ 0 & 0 & \alpha_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{pmatrix}. \tag{17}$$

Computationally, it is better to consider the matrix product as composed of the scalar products of rows of the left-hand matrix. If (r, s) denotes the scalar product of the rth and sth rows, we have the six independent equations:

$$\left.\begin{aligned} (1,1) &= a_{11} \\ (2,1) = (1,2) &= a_{12} \\ (3,1) = (1,3) &= a_{13} \end{aligned}\right\} \quad \text{giving successively } \alpha_{11}, \alpha_{21}, \alpha_{31},$$

$$\left.\begin{aligned} (2,2) &= a_{22} \\ (3,2) = (2,3) &= a_{23} \end{aligned}\right\} \quad \text{giving successively } \alpha_{22}, \alpha_{32},$$

and $$(3,3) = a_{33} \quad \text{giving } \alpha_{33}.$$

Thus only the matrix $\mathbf{L}$ need be recorded, and its elements are obtained column by column. A good check is provided by writing down an extra row whose elements are the sums of columns so far obtained. If this row is denoted by $\mathbf{s}$, and the corresponding row obtained from the matrix $\mathbf{A}$ denoted by Σ, the identity

$$n\text{th row } \mathbf{s}' = \Sigma_n, \tag{18}$$

where Σ_n is the nth element of Σ, holds for all n. As soon as a column is completed, and the element $\mathbf{s}_r$ is obtained, application of the identity (18) for $n = r$ gives a perfect check on the computation so far performed.

As pointed out by A. M. Turing, it is unnecessary to compute the reciprocal of $\mathbf{L}$ if the solution is desired to only one set of simultaneous equations. For equations (1) and (15) imply that

$$\mathbf{LL'x} = \mathbf{b}. \tag{19}$$

Thus writing $\mathbf{L'x} = \mathbf{y}$, we have

$$\left.\begin{aligned} \mathbf{Ly} &= \mathbf{b} \\ \mathbf{L'x} &= \mathbf{y} \end{aligned}\right\}, \tag{20}$$

and two processes of back-substitution give the solution.

Though this method applies only to a symmetric matrix, Turing (4) has shown that it can be extended, without the device of normalization, to the unsymmetric case in a form convenient for computation.

For symmetric matrices, this method is undoubtedly the quickest, and probably the most accurate, fewer figures being recorded than in any of the other processes described. It has also a rather important application in the problem of calculating the roots λ and the modes $\mathbf{x}$ of the matrix equation

$$(\mathbf{A}\lambda-\mathbf{B})\mathbf{x} = 0, \tag{21}$$

where $\mathbf{A}$ and $\mathbf{B}$ are both positive definite, symmetric matrices. By any method the case

$$(\mathbf{I}\lambda-\mathbf{C})\mathbf{x} = 0 \tag{22}$$

is very much more easy to deal with, and equation (21) can be transformed into (22) by the Choleski device. We have

$$(\mathbf{A}\lambda-\mathbf{B})\mathbf{x} = 0,$$

whence

$$(\lambda\mathbf{I}-\mathbf{A}^{-1}\mathbf{B})\mathbf{x} = 0. \tag{23}$$

Now write $\mathbf{A} = \mathbf{LL'}$, $\mathbf{A}^{-1} = \mathbf{L'}^{-1}\mathbf{L}^{-1}$. Equation (23) can then be written

$$(\lambda\mathbf{I}-\mathbf{L'}^{-1}\mathbf{L}^{-1}\mathbf{BL'}^{-1}\mathbf{L'})\mathbf{x} = 0. \tag{24}$$

Premultiplication by $\mathbf{L}'$ then gives the equation

$$(\lambda\mathbf{I}-\mathbf{L}^{-1}\mathbf{B}\mathbf{L}'^{-1})\mathbf{L}'\mathbf{x} = 0. \tag{25}$$

Since $\mathbf{L}^{-1}\mathbf{B}\mathbf{L}'^{-1}$ is symmetric, equation (25) is in the required form (22), the variable $\mathbf{x}$ having been replaced by $\mathbf{L}'\mathbf{x}$.

4. Numerical example

The example solved in this section is the one chosen by Morris (**3**) as an illustration of his escalator method. The complete working is given here for each of the three methods of elimination described in § 2, and for the vector and Choleski methods of § 3. The inverse of the matrix is also obtained by Choleski's method. The computational layouts should be self-explanatory, but the following points may be noted:

1. The original coefficients were given to six decimals. We have worked throughout with eight decimals.

2. In the elimination methods the pivots (underlined) are always on the principal diagonal, so that a good deal of symmetry is maintained. This is always the case with a positive definite, symmetric matrix.

3. The three elimination methods give almost exactly the same solution. If the substituting-back process is to be avoided, it is our opinion that the Jordan method is preferable to the 'below-the-line' method.

4. A check on any substituting-back process, particularly desirable in the Choleski method, is obtainable by a method analogous to the various summation checks mentioned previously. Thus if the solution $\mathbf{y}$ has been obtained to the equation

$$\mathbf{L}\mathbf{y} = \mathbf{b},$$

which occurs in the Choleski method, a good check is furnished by recording the row $\mathbf{s}$, whose elements are the sums of the columns of $\mathbf{L}$, and forming the product $\mathbf{s}\mathbf{y}$. This should then be equal to the sum of the elements of $\mathbf{b}$. All sums recorded in the computations are required for some check of this kind.

5. In the elimination method the last pivot has lost three of the original six significant figures. Thus in accordance with earlier remarks, if the coefficients are known accurately to only six figures, three figures only are reliable in the answers.

6. If the original coefficients are exact, answers can be obtained to any degree of accuracy, and the question arises as to which of the methods has given the best result. Conclusive evidence is not furnished by one example. In particular, it is difficult to be equally 'fair' to all methods—this is not necessarily achieved by retaining the same number of decimals throughout. However, the accurate result has been obtained, and the

errors in the approximate solutions are given, for what they are worth, in the following table, with the results given by Morris included for comparison.

Error in 5th decimal of	*Elimination*	*Vector*	*Choleski*	*Morris*
x_1	35	13	2	23
x_2	19	6	1	12
x_3	68	25	4	47
x_4	42	13	3	25
x_5	44	14	2	28
x_6	25	8	1	16

Our experience is that Choleski's method is quicker and more accurate than the vector method, for symmetric matrices. Both are much superior to pivotal condensation, which is less accurate and involves much more work. For unsymmetrical matrices the vector and Choleski methods cannot be used in their present form, but it is likely that the modified Choleski method given by Turing (4) is still the best both for speed and accuracy. The usefulness of the elimination process is that it can be 'mechanized' and adapted to Hollerith equipment, as described in the next section.

7. In one particular the above table is unfair to the vector method. As already stated, we are prepared to accept non-zero off-diagonal terms in equations (7), that is, we do not insist that quantities like $\mathbf{x}_r'\mathbf{A}\mathbf{x}_s$ should be exactly zero to the number of figures retained. We do, however, want to know these quantities accurately—they are subject to rounding-off errors, as is seen in the computation, where $\mathbf{x}_r'\mathbf{A}\mathbf{x}_s$ is not always equal to $\mathbf{x}_s'\mathbf{A}\mathbf{x}_r$. This accuracy can be achieved by the simple process of recording the $\mathbf{A}\mathbf{x}_r$ alone to one or more extra figures (depending on the magnitude of the vectors $\mathbf{x}_r$), an almost trivial increase in the amount of work. If one extra figure is kept in the present example, the errors, given in the following table, are seen to be very little worse than those of the Choleski method.

Error in 5th decimal of	*Vector method*
x_1	3
x_2	2
x_3	9
x_4	4
x_5	5
x_6	2

I. ELIMINATION AND BACK SUBSTITUTION

	m_i	(1)	(2)	(3)	(4)	(5)	(6)	b	s
1	−·66475184	·53999900	·52328600	·43578500	·36224200	·27647200	·18469100	·12367900	2·44615400
2		·52328600	·78719000	·36224200	·52565100	·18469100	·28026900	·04844800	2·71177700
3	−·46017099	·43578500	·36224200	·38814100	·29730400	·26397400	·16793600	·12495000	2·04033200
4	−·66775620	·36224200	·52565100	·29730400	·43767700	·16793600	·26324600	·04730400	2·10136000
5	−·23462061	·27647200	·18469100	·26397400	·16793600	·20157800	·11492100	·10647000	1·31604200
6	−·35603730	·18469100	·28026900	·16793600	·26324600	·11492100	·19406500	·03783100	1·24295900
1	−·88049650	·19214367		·19498396	·01281453	·15369832	− ·00161834	·09147310	·64349524
3		·19498396		·22144774	·05541466	·17898456	·03896434	·10265564	·79245090
4	−·25023809	·01281453		·05541466	·08667029	·04460744	·07609464	·01495255	·29055411
5	−·80824740	·15369832		·17898456	·04460744	·15824568	·04916412	·09510310	·67980322
6	−·17595276	− ·00161834		·03896434	·07609464	·04916412	·09427878	·02058170	·27746524
1	+·49417850	·02046098			− ·03597789	− ·00389696	− ·03592630	·00108517	−·05425500
4		− ·03597789			·07280343	− ·00018131	·06634428	−·01073480	·09225271
5	+·00249040	− ·00389696			− ·00018131	·01358187	·01767129	·01213195	·03930684
6	−·91127959	− ·03592630			·06634428	·01767129	·08742290	·00251916	·13803133
1	+·11646259	·00268148				− ·00398656	− ·00314038	−·00422023	−·00866569
5	−·66147606	− ·00398656				·01358142	·01783651	·01210521	·03953658
6		− ·00314038				·01783651	·02696471	·01230248	·05396332
1		·00231574				− ·00190927		−·00278745	−·00238098
5	+·82447511	− ·00190927				·00178300		·00396741	·00384114
5						·00020885		·00166923	·00187808
x		+5·38590	−2·81316	−11·59164	+6·36445	+7·99248	−4·20333		

II. ELIMINATION (JORDAN)

	m_i	1	2	3	4	5	6	b	s
1	− ·66475184	+ ·53999900	+ ·52328600	+ ·43578500	+ ·36224200	+ ·27647200	+ ·18469100	+ ·12367900	+ 2·44615400
2		+ ·52328600	+ ·78719000	+ ·36224200	+ ·52565100	+ ·18469100	+ ·28026900	+ ·04844800	+ 2·71177700
3	− ·46017099	+ ·43578500	+ ·36224200	+ ·38814100	+ ·29730400	+ ·26397400	+ ·16793600	+ ·12495000	+ 2·04033200
4	− ·66775620	+ ·36224200	+ ·52565100	+ ·29730400	+ ·43767700	+ ·16793600	+ ·26324600	+ ·04730400	+ 2·10136000
5	− ·23462061	+ ·27647200	+ ·18469100	+ ·26397400	+ ·16793600	+ ·20157800	+ ·11492100	+ ·10647000	+ 1·31604200
6	− ·35603730	+ ·18469100	+ ·28026900	+ ·16793600	+ ·26324600	+ ·11492100	+ ·19406500	+ ·03783100	+ 1·24295900
1	− ·88049650	+ ·19214367	0	+ ·19498396	+ ·01281453	+ ·15369832	− ·00161834	+ ·09147310	+ ·64349524
2	− 1·63579001	+ ·52328600	+ ·78719000	+ ·36224200	+ ·52565100	+ ·18469100	+ ·28026900	+ ·04844800	+ 2·71177700
3		+ ·19498396	0	+ ·22144774	+ ·05541466	+ ·17898456	+ ·03896434	+ ·10265564	+ ·79245090
4	− ·25023809	+ ·01281453	0	+ ·05541466	+ ·08667029	+ ·04460744	+ ·07609464	+ ·01495255	+ ·29055411
5	− ·80824740	+ ·15369832	0	+ ·17898456	+ ·04460744	+ ·15824568	+ ·04916412	+ ·09510310	+ ·67980322
6	− ·17595276	− ·00161834	0	+ ·03896434	+ ·07609464	+ ·04916412	+ ·09427878	+ ·02058170	+ ·27746524
1	+ ·49417850	+ ·02046098	0	0	− ·03597789	− ·00389696	− ·03592630	+ ·00108517	− ·05425500
2	− 5·97505159	+ ·20433319	+ ·78719000	0	+ ·43500425	− ·10809016	+ ·21653152	− ·11947507	+ 1·41549373
3	− ·76115452	+ ·19498396	0	+ ·22144774	+ ·05541466	+ ·17898456	+ ·03896434	+ ·10265564	+ ·79245090
4		− ·03597789	0	0	+ ·07280343	− ·00018131	+ ·06634428	− ·01073580	+ ·09225271
5	+ ·00249040	− ·00389696	0	0	− ·00018131	+ ·01358187	+ ·01767129	+ ·01213195	+ ·03930684
6	− ·91127959	− ·03592630	0	0	+ ·06634428	+ ·01767129	+ ·08742290	+ ·00251916	+ ·13803133
1	+ ·11646259	+ ·00268148	0	0	0	− ·00398656	− ·00314038	− ·00422023	− ·00866569
2	+ 6·67090356	+ ·41930294	+ ·78719000	0	0	− ·10700682	− ·17987898	− ·05532811	+ ·86427903
3	+ ·42774093	+ ·22236869	0	+ ·22144774	0	+ ·17912256	− ·01153391	+ ·11082724	+ ·72223233
4	− 2·46041140	− ·03597789	0	0	+ ·07280343	− ·00018131	+ ·06634428	− ·01073580	+ ·09225271
5	− ·66147606	− ·00398656	0	0	0	+ ·01358142	+ ·01783651	+ ·01210521	+ ·03953658
6		− ·00314038	0	0	0	+ ·01783651	+ ·02696471	+ ·01230248	+ ·05396332

II. ELIMINATION (JORDAN) (*cont.*)

	m_i	1	2	3	4	5	6	b	s
1		+ ·00231574	0	0	0	− ·00190927	0	− ·00278745	− ·00238098
2	−172·0200757	+ ·39835377	+ ·78719000	0	0	+ ·01197882	0	+ ·02674055	+1·22426313
3	−95·4448340	+ ·22102542	0	+·22144774	0	+ ·18675197	0	+ ·11608951	+ ·74531464
4	+12·19966836	− ·02825126	0	0	+ ·07280343	− ·04406646	0	− ·04100496	− ·04051925
5	+ ·82447511	− ·00190927	0	0	0	+ ·00178300	0	+ ·00396741	+ ·00384114
6	+1·35610215	− ·00314038	0	0	0	+ ·01783651	+ ·02696471	+ ·01230248	+ ·05396332
1	+9·14182428	+ ·00231574	0	0	0	− ·00190927	0	− ·00278745	− ·00238098
2	−1629·933397	0	+ ·78719000	0	0	+ ·34041159	0	+ ·50623791	+1·63383950
3	−1766·731769	0	0	+·22144774	0	+ ·36898193	0	+ ·38213721	+ ·97256688
4	+322·522959	0	0	0	+ ·07280343	− ·06735892	0	− ·07501093	− ·06956642
5		0	0	0	0	+ ·00020885	0	+ ·00166923	+ ·00187808
6	−73·0061767	0	0	0	0	+ ·01524734	+ ·02696471	+ ·00852241	+ ·05073446
1		+ ·00231574	0	0	0	0	0	+ ·01247236	+ ·01478810
2		0	+ ·78719000	0	0	0	0	−2·21449581	−1·42730581
3		0	0	+·22144774	0	0	0	−2·56694446	−2·34549672
4		0	0	0	+ ·07280343	0	0	+ ·46335407	+ ·53615750
5		0	0	0	0	+ ·00020885	0	+ ·00166923	+ ·00187808
6		0	0	0	0	0	+ ·02696471	− ·11334169	− ·08637698
x		+5·38591	−2·81314	+11·59165	+6·36445	+7·99248	−4·20333		

III. ELIMINATION ('BELOW THE LINE')

	m_i	1	2	3	4	5	6	b	s
1	− ·66475184	+ ·53999900	+·52328600	+·43578500	+·36224200	+ ·27647200	+·18469100	+ ·12367900	+2·44615400
2		+ ·52328600	+·78719000	+·36224200	+·52565100	+ ·18469100	+·28026900	+ ·04844800	+2·71177700
3	− ·46017099	+ ·43578500	+·36224200	+·38814100	+·29730400	+ ·26397400	+·16793600	+ ·12495000	+2·04033200
4	− ·66775620	+ ·36224200	+·52565100	+·29730400	+·43767700	+ ·16793600	+·26324600	+ ·04730400	+2·10136000
5	− ·23462061	+ ·27647200	+·18469100	+·26397400	+·16793600	+ ·20157800	+·11492100	+ ·10647000	+1·31604200
6	− ·35603730	+ ·18469100	+·28026900	+·16793600	+·26324600	+ ·11492100	+·19406500	+ ·03783100	+1·24295900
2	+1·27034134	0	−1	0	0	0	0	0	−1
1	− ·88049650	+ ·19214367		+·19498396	+·01281453	+ ·15369832	−·00161834	+ ·09147310	+ ·64349524
3		+ ·19498396		+·22144774	+·05541466	+ ·17898456	+·03896434	+ ·10265564	+ ·79245090
4	− ·25023809	+ ·01281453		+·05541466	+·08667029	+ ·04460744	+·07609464	+ ·01495255	+ ·29055411
5	− ·80824740	+ ·15369832		+·17898456	+·04460744	+ ·15824568	+·04916412	+ ·09510310	+ ·67980322
6	− ·17595276	− ·00161834		+·03896434	+·07609464	+ ·04916412	+·09427878	+ ·02058170	+ ·27746524
2	−2·07801168	+ ·66475184		+·46017099	+·66775620	+ ·23462061	+·35603730	+ ·06154550	+2·44488244
3	+4·51573812	0		−1	0	0	0	0	−1
1	+ ·49417850	+ ·02046098			−·03597789	− ·00389696	−·03592630	+ ·00108517	− ·05425500
4		− ·03597789			+·07280343	− ·00018131	+·06634428	− ·01073580	+ ·09225271
5	+ ·00249040	− ·00389696			−·00018131	+ ·01358187	+·01767129	+ ·01213195	+ ·03930684
6	− ·91127959	− ·03592630			+·06634428	+ ·01767129	+·08742290	+ ·00251916	+ ·13803133
2	−7·59035515	+ ·25957289			+·55260389	− ·13731140	+·27506895	− ·15177412	+ ·79816021
3	−3·43717446	+ ·88049650			+·25023809	+ ·80824740	+·17595276	+ ·46356599	+2·57850074
4	+13·73561658	0			−1	0	0	0	−1

III. ELIMINATION ('BELOW THE LINE') (*cont.*)

	m_i	1	2	3	4	5	6	b	s	
1	+ ·11646259	+ ·00268148				− ·00398656	−·00314038	− ·00422023	− ·00866569	
5	− ·66147606	− ·00398656				+ ·01358142	+·01783651	+ ·01210521	+ ·03953658	
6		− ·00314038				+ ·01783651	+·02696471	+ ·01230248	+ ·05396332	
2	+8·47432440	+ ·53265785				− ·13593519	−·22850770	− ·07028559	+ ·09792937	
3	+1·93156537	+1·00415878				+ ·80887059	−·05208410	+ ·50046681	+2·26141208	
4	−33·79526759	− ·49417850				− ·00249040	+·91127959	− ·14746283	+ ·26714786	
6	+37·08550917	0				0	−1	0	−1	
1		+ ·00231574				− ·00190927		− ·00278745	− ·00238098	
2	+ ·82447511	− ·00190927				+ ·00178300		+ ·00396741	+ ·00384114	
2	−218·524208	+ ·50604525				+ ·01521718		+ ·03396962	+ ·55523205	
3	−431·003882	+ ·99809293				+ ·84332298		+ ·52422985	+2·36564576	
4	+167·569986	− ·38804852				− ·60528003		− ·56322843	−1·55655698	
6	+50·2917383	− ·11646259				+ ·66147606		+ ·45624373	+1·00125720	
1	+431·827407	−1				0		0	−1	
5						+ ·00020885		+ ·00166923	+ ·00187808	
2	−2070·571654					+ ·43243889		− ·64309492	+1·07553381	
3	−7978·097965					+1·66622576		+1·72563162	+3·39185738	
4	+4430·052095					− ·92521638		−1·03032139	−1·95553777	
6	−2707·472109					+ ·56545555		+ ·31605802	+ ·88151357	
1	+3947·690256					− ·82447511		−1·20369731	−2·02817242	
5	+4788·125449					−1		0	−1	
2								−2·81317	−2·81317	$=x_2$
3								−11·59165	−11·59165	$=x_3$
4								+6·36445	+6·36445	$=x_4$
6								−4·20334	−4·20334	$=x_6$
1								+5·38591	+5·38591	$=x_1$
5								+7·99248	+7·99248	$=x_5$

IV. VECTOR METHOD

	A						b
	+ ·53999900	+ ·52328600	+ ·43578500	+ ·36224200	+ ·27647200	+ ·18469100	+ ·12367900
	+ ·52328600	+ ·78719000	+ ·36224200	+ ·52565100	+ ·18469100	+ ·28026900	+ ·04844800
	+ ·43578500	+ ·36224200	+ ·38814100	+ ·29730400	+ ·26397400	+ ·16793600	+ ·12495000
	+ ·36224200	+ ·52565100	+ ·29730400	+ ·43767700	+ ·16793600	+ ·26324600	+ ·04730400
	+ ·27647200	+ ·18469100	+ ·26397400	+ ·16793600	+ ·20157800	+ ·11492100	+ ·10647000
	+ ·18469100	+ ·28026900	+ ·16793600	+ ·26324600	+ ·11492100	+ ·19406500	+ ·03783100
s	+2·32247500	+2·66332900	+1·91538200	+2·05405600	+1·20957200	+1·20512800	
x_1	+1·00000000	− ·96904994	− ·80701075	− ·67081976	− ·51198613	− ·34202100	
x_2	− ·96904994	+1·00000000	+ ·21440731	− ·62342227	+ ·29712339	− ·36163619	
x_3	−1·01478214	+ ·21440731	+1·00000000	−1·79847338	− ·97594513	−1·72196807	
x_4	+1·75836622	−1·00902811	−1·79847338	+1·00000000	+ ·73718963	− ·33252874	
x_5	+1·48670750	− ·65597144	−2·30176106	+ ·73718963	+1·00000000	−1·72024388	
x_6	−1·38636181	+ ·73312297	+2·83566639	−1·60067469	−1·72024388	+1·00000000	
							Row Sums
Ax_1	+ ·53999900	+ ·52328600	+ ·43578500	+ ·36224200	+ ·27647200	+ ·18469100	+2·32247500
Ax_2	0	+ ·28009973	− ·06005543	+ ·17462041	− ·08322418	+ ·10129420	+ ·41273473
Ax_3	0	0	+ ·02358150	+ ·04241070	+ ·02301425	+ ·04060659	+ ·12961304
Ax_4	−1	−1	+1	+ ·00954113	− ·00703360	+ ·00317269	+ ·00568021
Ax_5	−1	0	+1	+3	+ ·00765458	+ ·01316771	+ ·02082232
Ax_6	+1	0	−1	−6	−3	+ ·00063521	+ ·00063512

IV. VECTOR METHOD (*cont.*)

	A						$\mathbf{x}_s'\mathbf{b}$
$x_1' A x_r$	+ ·53999900	0	0	−1	−1	+1	+ ·12367900
$x_2' A x_r$	0	+ ·28009973	0	0	+1	−1	− ·07140313
$x_3' A x_r$	0	0	+ ·02358150	+2	+2	−2	+ ·00983037
$x_4' A x_r$	−1	0	0	+ ·00954110	−1	−2	− ·00882767
$x_5' A x_r$	−1	0	0	−1	+ ·00765456	−4	+ ·00583097
$x_6' A x_r$	+1	0	0	−1	−4	+ ·00063532	− ·00267067

α	x
+ ·22903568	+5·38638
− ·25492056	−2·81341
+ ·41686575	−11·59257
− ·92523338	+6·36496
+ ·76174135	+7·99301
−4·20363135	−4·20363

V. CHOLESKI

	A						b
	+ ·53999900	+ ·52328600	+ ·43578500	+ ·36224200	+ ·27647200	+ ·18469100	+ ·12367900
	+ ·52328600	+ ·78719000	+ ·36224200	+ ·52565100	+ ·18469100	+ ·28026900	+ ·04844800
	+ ·43578500	+ ·36224200	+ ·38814100	+ ·29730400	+ ·26397400	+ ·16793600	+ ·12495000
	+ ·36224200	+ ·52565100	+ ·29730400	+ ·43767700	+ ·16793600	+ ·26324600	+ ·04730400
	+ ·27647200	+ ·18469100	+ ·26397400	+ ·16793600	+ ·20157800	+ ·11492100	+ ·10647000
	+ ·18469100	+ ·28026900	+ ·16793600	+ ·26324600	+ ·11492100	+ ·19406500	+ ·03783100
Σ	+2·32247500	+2·66332900	+1·91538200	+2·05405600	+1·20957200	+1·20512800	+ ·48868200
	L						*Row sums*
	+ ·73484624						+ ·73484624
	+ ·71210271	+ ·52924449					+1·24134720
	+ ·59302882	− ·11347389	+ ·15356267				+ ·63311760
	+ ·49294938	+ ·32994280	+ ·27617847	+ ·09767861			+1·19674926
	+ ·37623109	− ·15725091	+ ·14986877	− ·07200762	+ ·08749041		+ ·38433174
	+ ·25133285	+ ·19139396	+ ·26443011	+ ·03248065	+ ·15050428	+ ·02520588	+ ·91534773
s	+3·16049109	+ ·77985645	+ ·84404002	+ ·05815164	+ ·23799469	+ ·02520588	

y [Ly = b]	**x [L′x = y]**
+·16830596	$-4·20354 = x_6$
−·13491521	$+7·99285 = x_5$
+·06401531	$+6·36480 = x_4$
−·09037462	$-11·59228 = x_3$
+·06664695	$-2·81334 = x_2$
−·10595387	$+5·38623 = x_1$
Sum −·03227548	

V. CHOLESKI (*cont.*)

Computation of $\mathbf{A}^{-1}$

$\mathbf{L}'^{-1}$

+1·36082890	−1·83100621	−6·60826068	+18·00155507	+16·99280531	−55·00091709
	+1·88948590	+1·39622029	−10·33008406	− 7·49763555	+29·08508695
		+6·51199930	−18·41215803	−26·30872348	+112·49937136
			+10·23765592	+ 8·42594334	−63·50368463
				+11·42982414	−68·24746657
					+39·67328258

$\mathbf{A}^{-1} = \mathbf{L}'^{-1}\mathbf{L}^{-1}$

+3686·78585	−1925·75615	−7009·10808	+3820·23503	+3947·89803	−2182·06693
−1925·75615	+1014·38705	+3668·59854	−2015·94069	−2070·68016	+1153·90088
−7009·10808	+3668·59854	+13729·67118	−7554·29775	−7978·50117	+4463·21935
+3820·23503	−2015·94069	−7554·29775	+4208·52408	+4430·27264	−2519·39963
+3947·89803	−2070·68016	−7978·50117	+4430·27264	+4788·35757	−2707·60103
−2182·06693	+1153·90088	+4463·21935	−2519·39963	−2707·60103	+1573·96935

5. Solution by Hollerith equipment

The present Hollerith equipment appears to be somewhat unsuitable for the computation of scalar products, an essential part of methods such as that of Choleski, and for this reason the elimination methods are more likely than others to be readily adaptable.

Several such methods for the theoretical use of punched card methods for solving simultaneous equations have been suggested by various writers (refs. 8 and 9). As far as we can discover, however, these methods have had only limited success in practice. The theoretical mechanization of the elimination method is fairly straightforward, but difficulties peculiar to digital machines have to be overcome before the method can be said to be practicable. These difficulties arise through the restricted capacity of the Hollerith multiplying punch, and the consequent need for great care in fixing the position of the decimal point. This trouble is particularly acute in the case of ill-conditioned equations.

The methods by which these difficulties were overcome are too lengthy to receive detailed treatment here. A full account, together with a detailed description of the complete process of solution, can be obtained from the authors at the National Physical Laboratory. The Hollerith equipment used consisted of a card punch, an eight by eight decimal multiplying punch, a rolling total tabulator, a reproducer summary punch, and a collator. A special feature of the method is the fact that tabulator and multiplying punch are working simultaneously throughout a large part of the work, effecting a considerable saving in time.

Several sets of eighteen very ill-conditioned equations have been solved by this method, with a time-saving factor of about eight, compared with computation on a desk machine.

6. Note on error analysis

An analysis of the magnitude of the errors arising from rounding off has recently been carried out by A. M. Turing (4), for some methods of solving linear equations and inverting matrices. In all those cases where a conveniently simple error estimate can be given this error does not exceed n (order of matrix) times the error which might be caused by altering the coefficients in the matrix to be inverted by the maximum rounding-off error. This in effect means that the errors only become large when the inverse of the matrix has large coefficients. In the case of the Jordan method, for example, the maximum error E in the inverse is given by

$$|E| = \epsilon |M|^2 n^3/3,$$

where ϵ is the rounding-off error in each calculation, and M is the largest coefficient in the inverse.

The statistical error is very much less than this.

The proportionality to the *square* of the inverse coefficients appears natural when we consider the case of finding the reciprocal of a scalar:

$$\frac{1}{x}-\frac{1}{x+\epsilon}=\frac{\epsilon}{x(x+\epsilon)}\simeq\left(\frac{1}{x}\right)^2\epsilon.$$

Hotelling (7) has implied that the building-up error in the elimination methods is so large that these methods can be used only with extreme caution. Turing's analysis shows that this estimate can be reached only in the most exceptional circumstances, and in our practical experience on matrices of orders up to the twentieth, some of them very ill-conditioned, the errors were in fact quite small.

The work described here has been carried out as part of the Research programme of the National Physical Laboratory and this paper is published by permission of the Director of the Laboratory.

REFERENCES

1. J. Morris, 'A successive approximation process for solving simultaneous linear equations', *A.R.C.R. & M.* No. 1711 (1936).
2. R. V. Southwell, *Relaxation Methods in Engineering Science* (Oxford, 1940).
3. J. Morris, 'An escalator process for the solution of linear simultaneous equations', *Phil. Mag.* Series 7, **37** (1946), 106.
4. A. M. Turing, 'Rounding-off errors in matrix processes', *Quart. J. Mech. and Applied Math., in the press.*
5. A. C. Aitken, 'Studies in practical mathematics. 1. The evaluation with applications of a certain triple product matrix', *Proc. Roy. Soc. Edin.* **57** (1936–7).
6. Frazer, Duncan, and Collar, *Elementary Matrices* (Cambridge Univ. Press).
7. H. Hotelling, 'Some new methods in matrix calculation', *Ann. Math. Stat.* **14**, No. 1 (March 1943).
8. R. J. Schmidt and R. A. Fairthorne, 'Some comments on the use of Hollerith machines for scientific calculations', *R.A.E. Report* A.D. 3153 (September 1940).
9. H. O. Hartley, 'The application of some commercial calculating machines to certain statistical calculations', Supplement to *Journal Roy. Stat. Soc.* **8**, No. 2 (1946).