ALGORITHM 43
CROUT WITH PIVOTING II

HENRY C. THACHER, JR.*
Argonne National Laboratory, Argonne, Illinois

```
real procedure INNERPRODUCT (u,v) index : (k) start : (s)
    finish : (f);
value s, f;  integer k, s, f;  real u, v;
comment  INNERPRODUCT forms the sum of u(k) × v(k) for
   k = s, s+1, . . . , f. If s > f, the value of INNERPRODUCT is
   zero. The substitution of a very accurate inner product proce-
   dure would make CROUT more accurate;
comment  INNERPRODUCT may be declared in the head of
   any block which includes the block in which CROUT is de-
   clared. It may be used independently for forming the inner
   product of vectors;
begin
        real h;
        h := 0;  for k := s step 1 until f do h := h+u × v;
        INNERPRODUCT := h
end     INNERPRODUCT;
procedure CROUT II (A, b, n, y, pivot, det, repeat)
comment  This procedure is a revision of Algorithm 16, Crout
   With Pivoting by George E. Forsythe, Comm. ACM 3, (1960)
   507-8. In addition to modifications to improve the running of
   the program, and to conform to proper usage, it provides for
   the computation of the determinant, det, of the matrix A. The
   solution is obtained by Crout's method with row interchanges,
   as formulated in reference [1], for solving Ay = b and transform-
   ing the augmented matrix [A b] into its triangular decomposi-
   tion LU with all L(k,k) = 1. If A is singular we exit to 'singular,'
   a nonlocal label.  pivot (k) becomes the current row index of
   the pivot element in the k-th column. Thus enough information
   is preserved for the procedure to process a new right-hand
   side without repeating the triangularization, if the boolean pa-
   rameter repeat is true. The accuracy obtainable from CROUT
   would be much increased by calling CROUT with a more accu-
   rate inner product procedure than INNERPRODUCT.
     The contributions of Michael F. Lipp and George E. Forsythe
   by prepublication review and pointing out several errors are
   gratefully acknowledged;
comment  Nonlocal identifiers appearing in this procedure are:
   (1) The nonlocal label 'singular', to which the procedure exits
   if det A=0, and (2) the real procedure 'INNERPRODUCT'
   given above;
        value n;  array A, b, y;  integer n;  integer array
          pivot;  real det;  Boolean repeat;
begin
        integer k, i, j, imax, p;  real TEMP, quot;
        det := 1;  if repeat then go to 6;
        for k := 1 step 1 until n do
1:      begin
          TEMP := 0;
          for i := k step 1 until n do
2:      begin
          A[i,k] := A[i,k] - INNERPRODUCT (A[i,p], A[p,k],
            p, 1, k-1);
          if abs(A[i,k]) > TEMP then
3:      begin
          TEMP := abs(A[i, k]);  imax := i
          end 3
        end 2;
        pivot [k] := imax;
comment  We have found that A[imax, k] is the largest pivot in
   column k. Now we interchange rows k and imax;
        if imax ≠ k then
4:      begin det := - det;  for j := 1 step 1 until n do
5:      begin
          TEMP := A[k,j];  A[k,j] := A[imax, j];  A[imax, j]
            := TEMP
        end 5;
        TEMP := b[k];  b[k] := b[imax]; b[imax] := TEMP
        end 4;
        comment  The row interchange is done. We proceed
          to the elimination;
          if A[k,k] = 0 then go to singular;
          quot := 1.0/A[k,k];
          for i .= k+1 step 1 until n do
          A[i,k] := quot × A[i,k];
          for j := k+1 step 1 until n do
            A[k,j] := A[k,j] - INNERPRODUCT (A[k,p]
              A[p,j], p, 1, k-1);
            b[k] := b[k] - INNERPRODUCT (A[k,p], b[p]
              p, i, k-1)
          end 1;  go to 7;
        comment  The triangular decomposition is now finished,
          and we skip to the back substitution;
6:      begin comment  This section is used when the formal
        parameter repeat is true, indicating that the matrix A
        has previously been decomposed into triangular form by
        CROUT II, with row interchanges specified by pivot,
        and that it is desired to solve the linear system with a
        new vector b, without repeating the triangularization;
          for k := 1 step 1 until n do
          begin
            TEMP := b[pivot[k]];  b[pivot[k]] := b[k];  b[k] :=
              TEMP;  b[k]  :=  b[k]  -  INNERPRODUCT
              (A[k, p], b[p], p, 1, k-1) end;
        end 6;
7:      for k := n step - 1 until 1 do
8:      begin if ¬ repeat then det := A[k,k] × det;
        y[k] := (b[k] - INNERPRODUCT (A[k,p], y[p], p,
          k+1, n)/A[k,k]
        end 8;
end CROUT II;
```

REFERENCE:

(1) J. H. WILKINSON, Theory and practice in linear systems. In John W. Carr III (editor), Application of Advanced Numerical Analysis to Digital Computers, pp. 43–100 (Lectures given at the University of Michigan, Summer 1958, College of Engineering, Engineering Summer Conferences, Ann Arbor, Michigan [1959]).

* Work supported by the U. S. Atomic Energy Commission.

CERTIFICATION OF ALGORITHM 43
CROUT II (Henry C. Thacher, Jr., *Comm. ACM*, 1960)

HENRY C. THACHER, JR.*
Argonne National Laboratory, Argonne, Illinois

CROUT II was coded by hand for the Royal Precision LGP-30 computer, using a 28-bit mantisa floating point interpretive system (24.2 modified).

The program was tested against the linear system:

$$A = \begin{pmatrix} 12.1719 & 27.3941 & 1.9827 & 7.3757 \\ 8.1163 & 23.3385 & 9.8397 & 4.9474 \\ 3.0706 & 13.5434 & 15.5973 & 7.5172 \\ 3.0581 & 3.1510 & 6.9841 & 13.1984 \end{pmatrix} \quad b = \begin{pmatrix} 6.6355 \\ 6.1304 \\ 4.6921 \\ 2.5393 \end{pmatrix}$$

with the following results:

$$A' = \begin{pmatrix} 12.171900 & 27.394100 & 1.9827000 & 7.3756999 \\ 0.25226957 & 6.6327021 & 15.097125 & 5.6565352 \\ 0.25124262 & -0.56260107 & 14.979620 & 14.527683 \\ 0.66680633 & 0.76468695 & -0.20207132 & -1.3606142 \end{pmatrix}$$

$$b' = \begin{pmatrix} 6.6354999 \\ 3.0181653 \\ 2.5702026 \\ -0.082780734 \end{pmatrix} \quad pivot = \begin{pmatrix} 1 \\ 3 \\ 4 \\ 4 \end{pmatrix} \quad y = \begin{pmatrix} 0.15929120 \\ 0.14691771 \\ 0.11257482 \\ 0.060840712 \end{pmatrix}$$

det = −1645.4499. All elements of Ab − y were less than 10^{-7} in magnitude. Identical results were obtained with the same b, and repeat **true**. With the same b and the last row vector of A replaced by (19.1927, 33.4409, 25.1298, 5.2811), i.e. A 4, j = A 1 j, + 2A 2, j − 3A 3, j, the results were:

det = $0.10924352 \times 10^{-3}$,

y = $(0.29214425 \times 10^{8}, -0.12131172 \times 10^{8}, 0.72411923 \times 10^{7}, -0.51018392 \times 10^{7})$

Failure to recognize this singular matrix is due to roundoff, either in the data input or in the calculation.

* Work supported by the U.S. Atomic Energy Commission.

CERTIFICATION OF ALGORITHM 43
CROUT II [Henry Thacher, Jr., *Comm. ACM* (1960), 176]

C. DOMINGO AND F. RODRIGUEZ-GIL
Universidad Central, Caracas, Venezuela

CROUT II was coded in PUC-R2 and tested in the IBM-1620. Two types of INNERPRODUCT subroutines were used. The first one finds the scalar product in fixed-point arithmetic to increase accuracy, using an accumulator of 32 digits. The second one uses ordinary floating-point with eight significative figures.

Using a unit matrix as right-hand side, a 6 × 6 segment of Hilbert matrix was inverted. The inverse was inverted again.

The maximum difference between this result and the original segment of Hilbert matrix was:

Using fixed-point INNERPRODUC.........	8.2426×10^{-4}
(Value of determinant.....................	$4.7737088 \times 10^{-18}$)
Using floating-point INNERPRODUC......	3.014016×10^{-2}
(Value of determinant....................	$4.4950721 \times 10^{-18}$)

Two typographical errors were observed in the algorithm:

The statement:

b[*k*] := *g*[*k*] − INNERPRODUCT (*A*[*k*,*p*], *b*[*p*], *p*,*i*,*k*−1)

should be:

b[*k*] := *b*[*k*] − INNERPRODUCT (*A*[*k*,*p*], *b*[*p*], 1,*k*−1)

The statement:

y[*k*] := (*b*[*k*] − INNERPRODUCT (*A*[*k*,*p*], *y*[*p*], *p*,*k*+1,*n*)/*A*[*k*,*k*]

should be:

y[*k*] := (*b*[*k*] − INNERPRODUCT (*A*[*k*,*p*], *y*[*p*], *p*,*k*+1,*n*))/*A*[*k*,*k*]

Storage may be saved eliminating the array *y* and using instead the array *b*, in which the solution is formed.

A previous certification of this algorithm [*Comm. ACM 4*, 4 (Apr. 1961), 182] was tested again with the same results. Two errors were detected in the certification: The row that must replace the last row of *A* in order to obtain a singular matrix must be:

19,1927 33.4409 −251298 −5.2811