ALGORITHM 41
EVALUATION OF DETERMINANT

JOSEF G. SOLOMON

RCA Digital Computation and Simulation Group, Moorestown, New Jersey


```
real procedure   Determinant (A,n);
real array       A; integer n;
comment This procedure evaluates a determinant by triangu-
  larization;
begin real       Product, Factor, Temp; array B[1 : n, 1 : n],
                   C[1 : n, 1 : n];
integer          Count, Sign, i, j, r, y;
                 Sign := 1; Product := 1;
for              i := 1 step 1 until n do for j := 1 step 1 until
                   n do
begin            B[i,j] := A[i,j]; C[i,j] := A[i,j] end;
for              r := 1 step 1 until n-1 do
begin            Count := r-1;
      zerocheck: if B[r,r] ≠ 0 then go to resume;
                 if Count < n-1 then Count := Count + 1
                 else go to zero;
for              y := r step 1 until n do
begin            Temp := B[Count+1,y]; B[Count+1,y] :=
                   B[Count,y]; B[Count,y] := Temp end;
                 Sign := - Sign; go to zerocheck;
           zero: Determinant := 0; go to return;
         resume: for i := r+1 step 1 until n do
begin            Factor := C[i,r] / C[r,r];
                 for j := r+1 step 1 until n do
begin            B[i,j] := B[i,j] - Factor × C [r,j] end end;
for              i := r+1 step 1 until n do
                 for j := r+1 step 1 until n do C[i,j] := B[i,j]
                   end;
for              i := 1 step 1 until n do Product := Product
                   × B[i,i]; Determinant := Sign × Product;
         return: end
```

ALGORITHM 41, REVISION
EVALUATION OF DETERMINANT [Josef G. Solomon, RCA Digital Computation and Simulation Group, Moorestown, N. J.]

BRUCE H. FREED

Dartmouth College, Hanover, N. H.


```
real procedure determinant (a,n);
real array a; integer n; value a,n;
comment This procedure evaluates a determinant by triangu-
  larization;
begin real product, factor, temp;
array b[1:n,1:n];
integer count, ssign, i, j, r, y;
ssign := product := 1;
for i := 1 step 1 until n do
for j := 1 step 1 until n do
b[i,j] := a[i,j];
for r := 1 step 1 until r-1 do
begin count := r-1;
zerocheck: if b[r,r] ≠ 0 then go to resume;
if count < n-1 then count := count +1 else go to zero;
for y := r step 1 until n do
begin temp := b[count+1,y];
  b[count+1,y] := b[count,y];
  b[count,y] := temp end;
ssign := -ssign;
go to zerocheck;
zero: determinant := 0; go to return;
resume: for i := r+1 step 1 until n do
begin factor := b[i,r]/b[r,r];
for j := r+1 step 1 until n do
b[i,j] := b[i,j] - factor × b[r,j] end end;
for i := 1 step 1 until n do
product := product × b[i,i];
determinant := ssign × product;
return: end
```

CERTIFICATION OF ALGORITHM 41
EVALUATION OF DETERMINANT [Josef G. Solomon, RCA Digital Computation and Simulation Group, Moorestown, N. J.]

BRUCE H. FREED

Dartmouth College, Hanover, N. H.

When Algorithm 41 was translated into SCALP for running on the LGP-30, the following corrections were found necessary:

1. In the "*y*" loop after "*B[Count,y] := Temp*" and before the "**end**" insert
   "*Temp := C[Count*+1,y];
   C[Count +1,y] := C[Count,y];
   C[Count,y] := Temp"
2. "Sign" is an ALGOL word when uncapitalized. However, many systems (if not all) do not recognize the difference between small and capital letters. For this reason "*Sign*" was changed to "*ssign*" for the LGP-30 run (and in the revision which follows later).

The following addition might be made in the specification as a concession to efficiency: "**value** *A,n*;".

The following changes might be made to make the Algorithm less wordy:

1. for "*Ssign* := 1; *Product* := 1;"
   put "*Ssign* := *Product* := 1;"
2. for "**begin** *B[i,j]* := *A[i,j]*; *C[i,j]* := *A[i,j]* **end**;"
   put "*B[i,j]* := *C[i,j]* := *A[i,j]*;"
3. for "**begin** *B[i,j]* := *B[i,j]* − *Factor* × *C[r,j]* **end** **end**;"
   put "*B[i,j]* := *B[i,j]* − *Factor* × *C[r,j]* **end**;"

The above corrections and changes were made and the program was run with the correct results, as follows:

$$A = \begin{pmatrix} 10.96597 & 35.10765 & 96.72356 \\ 2.35765 & -84.11256 & .87932 \\ 18.24689 & 22.13579 & 1.11123 \end{pmatrix}$$

Determinant = .1527313$_{10}$06

Hand calculation on a desk calculator gives the value of the determinant for the above matrix as 152,731.3600.

$$A = \begin{pmatrix} 1.0 & 3.0 & 3.0 & 1.0 \\ 1.0 & 4.0 & 6.0 & 4.0 \\ 1.0 & 5.0 & 10.0 & 10.0 \\ 1.0 & 6.0 & 15.0 & 20.0 \end{pmatrix} \qquad \text{Determinant} = .9999999_{10}+00$$

The above matrix, being a finite segment of Pascal's triangle, has determinant equal to 1.000000000.

$$A = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 5.0 & 9.0 & 2.0 \\ 7.0 & 5.0 & 4.0 \end{pmatrix} \qquad \text{Determinant} = .0000000_{10}+00$$

This is, of course, exactly correct.

Finally, one major change can be made which does away with several instructions and reduces variable storage requirements by n^2. This change is the complete removal of matrix C from the program. It is extraneous.

The revised Algorithm was translated into SCALP and run on the LGP-30 with exactly the same results as above.

The revised Algorithm 41 follows.

REMARK ON REVISION OF ALGORITHM 41 EVALUATION OF DETERMINANT [Josef G. Solomon, *Comm. ACM 4* (Apr. 1961), 176; Bruce H. Freed, *Comm. ACM 6* (Sept. 1963), 520]

LEO J. ROTENBERG (Recd 7 Oct. 63)
Box 2400, 362 Memorial Dr., Cambridge, Mass.

While desk-checking the program an error was found. For example, the algorithm as published would have calculated the value zero as the determinant of the matrix

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

The error lies in the search for a nonzero element in the rth column of the matrix b.

Editor's Note. Apparently the best general determinant evaluators in this section are imbedded in the linear equation solvers Algorithm 43 [*Comm. ACM 4* (Apr. 1961), 176, 182; and *6* (Aug. 1963), 445] and Algorithm 135 [*Comm. ACM 5* (Nov. 1962), 553, 557]. They search each column for the largest pivot in absolute value. Algorithm 41 searches only for a nonzero pivot in each column, and will therefore fail for the matrix

$$\begin{bmatrix} 2^{-t} & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

if $t \gg s$, for a machine with s-bit floating point.

It is hoped that soon a good determinant evaluator will be published to take the place of Algorithm 41.—G. E. F.

CERTIFICATION OF: ALGORITHM 41 [F3] EVALUATION OF DETERMINANT [Josef G. Solomon, *Comm. ACM 4* (Apr. 1961), 171] ALGORITHM 269 [F3] DETERMINANT EVALUATION [Jaroslav Pfann and Josef Straka, *Comm. ACM 8* (Nov. 1965), 668]

A. BERGSON (Recd. 4 Jan. 1966 and 4 Apr. 1966)
Computing Lab., Sunderland Technical College, Sunderland, Co. Durham, England

Algorithms 41 and 269 were coded in 803 ALGOL and run on a National-Elliott 803 (with automatic floating-point unit).

The following changes were made:
(i) **value** n; was added to both Algorithms;
(ii) In Algorithm 269, since procedure *EQUILIBRATE* is only called once, it was not written as a procedure, but actually written into the **procedure** *determinant* body.

The following times were recorded for determinants of order N (excluding input and output), using the same driver program and data.

N	T_1 *Algorithm 41* (minutes)	T_2 *Algorithm 269*
10	0.87	0.78
15	2.77	2.18
20	6.47	4.78
25	12.47	8.99
30	21.37	14.98

From a plot of $\ln(T_1)$ against $\ln(N)$ it was found that

$$T_1 = 0.00104N^{2.92}.$$

Similarly,

$$T_2 = 0.00153N^{2.70}.$$

From a plot of T_1 against T_2, it was found that Algorithm 269 was 30.8 percent faster than Algorithm 41, but Algorithm 41 required less storage.