

A COMPACT NON-ITERATIVE POISSON SOLVER

by

O. Buneman

Prepared under

AEC Contract AT(04-3) 326 PA 20

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, in the extent that such employee or contractor of the Commission, or employee of such contractor invents, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

SUIPR Report No. 294

May 1969

Institute for Plasma Research  
Stanford University  
Stanford, California

RESTRICTION OF THIS DOCUMENT IS UNLIMITED

Fig

## A Compact Non-Iterative Poisson Solver\*

O. Buneman

Electrical Engineering Department

Stanford University

Subroutine "XYPOIS" converts a two-dimensional array of M by N charges Q into the corresponding array of potentials. It solves Poisson's equation in the finite-difference form\*\*:

$$\frac{2\phi - \phi_{\text{above}} - \phi_{\text{below}}}{S} + 2\phi - \phi_{\text{left}} - \phi_{\text{right}} = 2Q$$

where S is the square of the cell ratio,

$$S = (\Delta Y / \Delta X)^2$$

The same array is used for the final potentials as for the initial charges.

More precisely, one prescribes (M-1) by (N-1) internal charges, and arbitrary boundary potentials "Q" are prescribed on the periphery of the rectangular charge domain. The peripheral Q-values will be left untouched and after calling XYPOIS one has an (M+1) by (N+1) array of potentials from the original charges and 2(M-1)+2(N-1) boundary values. M and N must be powers of 2.

The double array  $Q_{mn}$ , with m from 0 to M and n from 0 to N for

\* Work supported by AEC Contract AT(04-3) 326 PA 20

\*\*The factor 2 in front on the charge was chosen for convenience. The correct scaling of the charge per unit length depends on the user's preference for units.

potentials but m from 1 to M-1 and n from 1 to N-1 for charges, is listed as a single array with the index  $I = m+n(M+1)$ . The corner elements, among them the element  $m = 0, n = 0$  with  $I = 0$ , are never required - which is fortunate, or, one might say, fortunante! The total Q-array to be declared in the main program should be  $(M+1) \cdot (N+1) - 2$ . The numbering of elements is illustrated in figure 1.

The principle of XYPOIS is cyclic reduction in two directions:  
The columns  $\vec{\Phi}$  are related to  $\vec{Q}$  by:

$$\vec{A} \vec{\Phi} = 2\vec{Q} + \vec{\Phi}_{\text{left}} + \vec{\Phi}_{\text{right}}$$

where  $\vec{A}$  is tri-diagonal, with  $2 + 2/S$  in the diagonal,  $-1/S$  on each side.  
On left-multiplying with A one obtains:

$$A' \vec{\Phi} = A'(Q' - Q_r - Q_l) + 2Q' + \vec{\Phi}_{ll} + \vec{\Phi}_{rr}$$

where vector and matrix arrows have been omitted, 'rr' stands for the subscript two to the right, 'll' for the subscript two to the left and where

$$A' = A^2 - 2 = (A - \sqrt{2})(A + \sqrt{2}) \quad \dots \text{factors } A - 2\cos(\text{odd } \pi/2)$$

$$Q' = 2A^{-1}Q + Q_r + Q_l \text{ (branch '26' and DO loops '27', '22', IPHASE = 3)}$$

This has thrown out odd potential columns. On doing it again one throws out twice-odd potential columns:

$$A'' \vec{\Phi} = A''(Q'' - Q'_{ll} - Q'_{rr}) + 2Q'' + \vec{\Phi}_{llll} + \vec{\Phi}_{rrrr}$$

with

$$A'' = (A - \sqrt{2+\sqrt{2}})(A - \sqrt{2-\sqrt{2}})(A + \sqrt{2-\sqrt{2}})(A + \sqrt{2+\sqrt{2}}) \dots \text{factors } A - 2\cos\frac{\text{odd}\pi}{4}$$

and

$$Q'' = Q' - Q'_r - Q'_1 + Q'_{rr} + Q'_{11} + A'^{-1} (2Q' + Q'_{rr} + Q'_{11} - Q'_r - Q'_1 - Q'_{rrr} - Q'_{111})$$

(branch '28' and DO loops '29', '22', IPHASE = 4)

This throw-out or "MODE 2" operation is continued till the center column is related to the two given lateral columns. Subsequent filling-in is done in "MODE 1" by going through branches '24' and '20', following DO loops '25', '22' and '23', '22' respectively (IPHASE = 2 and IPHASE = 1).

The matrix inversions are by means of "cyclic reduction" and the reader will recognize the scheme above as the principle of cyclic reduction of G. H. Golub (private communication 1965) extended from scalars to vectors. An outer loop can be built around XYPOIS which uses the same principle at the next level and solves Poisson's equation in three dimensions. See Varga (1962), p. 195 on cyclic reduction.

Our algorithm is so arranged that one never multiplies by a matrix A or A' etc. One only divides. Otherwise one would run into very large numbers. (A precaution against such large numbers has been built into the inner cyclic reduction loop for scalars, which does employ multiplications - see the IF statement following DO loop 21. The multiplier limitation should be raised if one desires higher precision than the 22-bit single precision of an IBM 360.) Some of the additions in the formation of Q'' could be eliminated if one kept separate potential and charge arrays, however in the present algorithm we overwrite all quantities labeled Q, Q', Q'' etc. and eventually place  $\Phi$ -s in the same register.

The number of multiplications is  $2MN \log_2 N$  approximately and several times that many additions. Typically it is like going over the entire

grid with the direct Poisson operator about 10 times. One could compare this with 10 iterations in a relaxation scheme and ask how near to machine accuracy one gets with zero a priori information on the charge.

The diagonal elements for cyclic reduction within each factor of  $A'$ ,  $A''$  etc. are purely algebraic. That they can be expressed as cosines of multiples of  $\pi/N$  is coincidental, but it shows that XYPOIS bears some hidden relationship to the method of Hockney (1967) employing the Fourier analysis of Cooley-Tukey (1965). The quantities  $2\cos L\pi/N$  are generated in the statements up to statement "4" in XYPOIS and stored under the name TWOCOS, an array which must be allowed the length  $N$ . They could be obtained from a cosine function instead but should always be prestored for speed. In repeated uses of XYPOIS the TWOCOS generator can be taken outside the subroutine. The register RECIP must be  $M/2$  long and merely stores some recurring coefficients of the inner cyclic reduction loop. The register P is also for temporary storage, namely of that array of length  $M$  which is currently processed through this inner loop.

The statements beginning with 'I = M + 1' just before '5' and ending in the following 'IF' are concerned with transferring the given boundary potentials at the top and bottom edges into adjacent surface charges. This enables one to run the inner cyclic reduction with zero terminal elements (note  $P(M) = 0$  while  $P(0)$  is not referred to): equation 1 shows the equivalence of a top edge potential  $\phi$  with a charge just below of value  $\phi/2S$  and zero top edge potential. If, as in many applications, the top and bottom "plates" are kept at zero potential, the statements 'I = M+1' through '5' and down to the following 'IF' may

be omitted. At the lateral boundaries the transfer is unnecessary.

The core of the subroutine begins with the statement 'MODE = 2'. Timing tests should begin there.

The ratio S is redundant in many applications and it pays, for the sake of speed, to erase multiplications by S whenever one uses solely  $S = 1$ . It is a useful parameter when one wants a potential array print-out without distortion on standard IBM printers which have a cell ratio of  $3/5$  or  $S = 0.36$ .

Such potential printouts (contour maps) have been found extremely instructive for monitoring any calculation which involves potential distributions. An illustrative "MAIN" program is therefore shown below which, after setting boundary potentials and six internal charges, calls XYPOIS, determines potential maximum and minimum (DO loop 45), and then prints out the potential, discretized into the 32 levels indicated by the character declaration following the DIMENSION statement. Alternate levels are left blank to give visual contour display. The procedure for introducing boundary potentials, and the numbering, are exemplified here: the left and right boundaries have the potential rising linearly to a maximum of 128.0 (DO loop 40) then going linearly and more gently down to 32.0 at the top (DO loop 41). Across the top edge the potential is made to climb parabolically up to 64.0 in the middle (DO loop 43), while the potential is kept zero at the bottom. The internal charges form a hexagon around the center  $m = 80$ ,  $n = 20$  after zeroing the charge elsewhere (DO loop 42). The user might like to try out this MAIN program together with the subroutine just as shown, and

reproduce the contour pattern shown in Fig. 5.

The subroutine solves the difference equation to machine accuracy as has been tested by injecting random charges and boundary potentials, then differencing the output potentials as in our first equation and making the comparison with the original charges.

#### REFERENCES

- 1 R. W. Hockney, Proceedings of the American Physical Society Topical Conference on Numerical Simulation of Plasma, Los Alamos Scientific Laboratory, 1968 p. D6-1.
- 2 J. W. Cooley and J. W. Tukey, Math. Comp. 19, 297, 1965.
- 3 R. S. Varga, "Matrix Iterative Analysis", Prentice-Hall, N. J., 1962, p. 195.

	M+M1	.	.	M+NF*M1	
MF					MF+N*M1
			$l = m + n*M1$		
.					.
2					2+N*M1
1					1+N*M1
	M1	.	.	NF*M1	

Fig. 1a. Numbering of mesh-points,  
 general case; definitions:  
 $M1=M+1$ ,  $MF=M-1$ ,  $NF=N-1$ .

(128)	257	386	.	8255	
127	256	385	.	8254	8383
			$l = m + n*129$		
.	.	.	.	.	.
2	131	260	.	8129	8258
1	130	259	.	8128	8257
	129	258	.	8127	(8256)

Fig. 1b. Numbering of mesh-points,  
 case  $M=128$ ,  $N=64$ .

FIG. 1.

```

SUBROUTINE XYPOIS(M,N,S,Q,P,TWOCOS,RECIP)
DIMENSION Q(1),P(1),TWOCOS(1),RECIP(1)
TWOCOS(N/2)=0.
LO=N/2.
1  L=LO/2
   TWOCOS(L)=SQRT(2.+TWOCOS(LO))
   LO=L
2  TWOCOS(N-L)=-TWOCOS(L)
   L=L+2*LO
   IF((2*L/N)*(2*LO-3)) 4,3,1
3  TWOCOS(L)=(TWOCOS(L+LO)+TWOCOS(L-LO))/TWOCOS(LO)
   GO TO 2
4  LO=N/2
   JU=(N-1)*(M+1)
   IU=M-1
   I=M+1
5  Q(I+1)=Q(I+1)+.5*Q(I)/S
   I=I+M+1
   Q(I-2)=Q(I-2)+.5*Q(I-1)/S
   IF(I.LE.JU) GO TO 5
   P(M)=0.
   ID=1

```

FIG. 2.

```

15  MODE=2
    LI=2*LO
    I PHASE=2*MODE-LI/N
    JD=(M+1)*N/LI
    JH=(M+1)*(N/(2*LI))
    JT=JD+JH
    JI=2*JD
    JO=JD*MODE
    DO 11 J=JO, JU, JI
        JI=J+1
        JIU=J+IU
        GO TO (20,24,26,28), I PHASE
28  DO 29 I=J1, JIU
        PI=Q(I)-Q(I+JT)-Q(I-JT)
        Q(I)=Q(I)-Q(I+JH)-Q(I-JH)+Q(I+JD)+Q(I-JD)
29  P(I-J)=PI+Q(I)
        GO TO 10
26  DO 27 I=J1, JIU
        P(I-J)=2.*Q(I)
27  Q(I)=Q(I+JD)+Q(I-JD)
        GO TO 10
24  DO 25 I=J1, JIU
        P(I-J)=2.*Q(I)+Q(I+JD)+Q(I-JD)
25  Q(I)=Q(I)-Q(I+JH)-Q(I-JH)
        GO TO 10
20  DO 23 I=J1, JIU
        P(I-J)=2.*Q(I)+Q(I+JD)+Q(I-JD)
23  Q(I)=0
10  DO 22 L=LO, N, LI
        A=2.+S*(2.-TWO COS(L))
19  RECIP(ID)=1./A
        II=2*ID
        DO 21 I=II, IU, II
21  P(I)=P(I)*A+P(I+ID)+P(I-ID)
        A=A*A-2.
        ID=II
        IF((A.LT.1.E8).AND.(ID.LT.M/2)) GO TO 19
        A= S/A
        DO 18 I=II, IU, II
18  P(I)=P(I)*A
16  ID=II/2
        A=RECIP(ID)
        P(ID)=(S*P(ID)+P(II))*A
        IO=ID+II
        DO 17 I=IO, IU, II
17  P(I)=(S*P(I)+P(I+ID)+P(I-ID))*A
        II=ID
        IF(ID.GT.1) GO TO 16
22  CONTINUE
        DO 11 I=J1, JIU
11  Q(I)=Q(I)+P(I-J)
12  GO TO (14,13,12,12), I PHASE
12  LO=LO/2
    IF(LO.EQ.1) MODE=1
    GO TO 15
13  LO=2*LO
    IF(LO.LT.N) GO TO 15
14  RETURN
    END

```

FIG. 3.

```

DIMENSION Q(8383), P(128), TWOCOS(64), RECIP(64)
LOGICAL*1 KP(129), STAR/'*'/, LIT(32)/'0','1','2','3',
&' '4' '5' '6' '7' '8' '9' 'A' 'B',
&' 'C' 'D' 'E' 'F' /
DO 40 I=1,32
  Q(I)=4.*I
40  Q(I+8256)=Q(I)
  DO 41 I=33,127
    Q(I)=160.-I
41  Q(I+8256)=Q(I)
42  DO 42 I=128,8256
    Q(I)=0.
  J=0
  DO 43 I=128,8255,129
    Q(I)=64.-(32.-J)**2/32.
43  J=J+1
  Q(78+20*129)=-20.
  Q(79+21*129)=-20.
  Q(81+21*129)=-20.
  Q(82+20*129)=-20.
  Q(81+19*129)=-20.
  Q(79+19*129)=-20.
  CALL XYPOIS(128,64,.36,Q,P,TWOCOS,RECIP)
  PMAX=0.
  PMIN=0.
  DO 45 I=1,8383
    IF(Q(I).GT.PMAX) PMAX=Q(I)
    IF(Q(I).LT.PMIN) PMIN=Q(I)
45  CONTINUE
  WRITE(6,47)PMIN,PMAX
47  FORMAT(1H0,2F20.6)
  SC=31.99999/(PMAX-PMIN)
  J=-1
44  KP(1)=STAR
  KP(129)=STAR
  IO=2
  IU=128
49  DO 48 I=10,IU
48  KP(I)=LIT(1+IFIX(SC*(Q(I+J)-PMIN)))
  WRITE(6,50)KP
  J=J+129
  IO=1
  IU=129
50  IF(J-8255) 49,44,51
  FORMAT(1H ,129A1)
51  STOP
  END

```

FIG. 4.

