E

10 JAN. 1984

TRI-PP 83-95
C1

Paper presented at the EPS Conference on
Computing in Accelerator Design and Operation
Berlin, September 20-23, 1983

TRI-PP-83-95
Sep 1983

A Fortran Program (RELAX3D)
to Solve the 3 Dimensional Poisson (Laplace) Equation

H. Houtman*, C.J. Kost
TRIUMF, University of British Columbia
Vancouver, B.C., Canada

1) Introduction

RELAX3D is an efficient, user friendly, interactive FORTRAN program which solves the Poisson (Laplace) equation $\nabla^2 \phi = \rho$ for a general 3 dimensional geometry consisting of Dirichlet and Neumann boundaries approximated to lie on a regular 3 dimensional mesh. The finite difference equations at these nodes are solved using a successive point-iterative over-relaxation method ^{1,2)}. A menu of commands, supplemented by an extensive HELP facility controls the dynamic loading of the subroutine describing the problem case, the iterations to converge to a solution, and the contour plotting of any desired slices etc.

Basic parts of the program were developed in 1973 ³⁾ based on work by others^{4,5)} with the idea in mind that appropriate applications would be found for it at some future date when computers could run it in a practical manner. Fortunately, our view into the crystal ball turned out to be accurate and the program has found many problems to solve both at TRIUMF and elsewhere.

2) Method

Finite difference methods, dating back to Gauss ⁶⁾ can be used to obtain numerical solutions, to any desired accuracy, for all static electric and magnetic fields in uniform media and time-varying fields where eddy currents can be neglected. We will restrict ourselves to problems given in TABLE 1. The partial differential equation for the field, being replaced by a set of finite difference equations at discrete points in a mesh occupying the volume of interest, results in a large number of equations connecting the potential ϕ at each point with those of adjacent points and is solved by an iterative procedure known as relaxation. The basis of the iteration procedure is that the potential ϕ at each grid point (that is allowed to vary) on each iteration is replaced using the finite - difference equation. The over-relaxation method accelerates the convergence by replacing the old ϕ by

*Current address Physics Department, University of British Columbia,
Vancouver, B.C., Canada.

CERN LIBRARIES, GENEVA

CM-P00067270

$$\phi_{\text{old}} + R * (\phi_0 - \phi_{\text{old}}) \text{ where usually } 1 < R < 2$$

The patterns of points used to form the finite difference equations are commonly called "templates", "molecules" or "stencils".

Let the uniform grid spacing along the x, y, and z axes be h_x , h_y , and h_z respectively where in general these may all be different.

FIG.1

The 3 dimensional molecule is shown in Fig. 1 for the Poisson equation $\nabla^2 \phi = \rho$ resulting in the finite difference equation

$$\phi_0 = W_x(\phi_1 + \phi_2) + W_y(\phi_3 + \phi_4) + W_z(\phi_5 + \phi_6) - \rho/C$$

where $C = \frac{2}{h_x^2} + \frac{2}{h_y^2} + \frac{2}{h_z^2}$, $W_x = \frac{1}{Ch_x^2}$, $W_y = \frac{1}{Ch_y^2}$, $W_z = \frac{1}{Ch_z^2}$

and $\rho = \rho(x_0, y_0, z_0)$

For cylindrical co-ordinates with rotational symmetry

$$\phi_0 = W_x(\phi_1 + \phi_2) + W_y(\phi_3 + \phi_4) + \frac{W_r}{R}(\phi_4 - \phi_3) - \rho/C$$

where $C = \frac{2}{h_x^2} + \frac{2}{h_y^2}$, $W_x = \frac{1}{Ch_x^2}$, $W_y = \frac{1}{Ch_y^2}$

$$W_r = \frac{1}{2Ch_y^2} = \frac{W_y}{2}$$

and axis of symmetry is along x axis and radius R is along y axis, while for R = 0 (on axis)

$$\phi_0 = W_x(\phi_1 + \phi_2) + W_y\phi_4 - \rho/C$$

$$C = \frac{2}{h_x^2} + \frac{4}{h_y^2}, \quad W_x = \frac{1}{Ch_x^2}, \quad W_y = \frac{4}{Ch_y^2}$$

$$\rho = \rho(x_0, y_0) = \rho(z, R)$$

3. Boundaries

RELAX3D handles three types of boundaries. The first type is a DIRICHLET boundary, that is a boundary (usually metal) on which the potential ϕ is known (and hence fixed). These boundary points are always specified as negative in the user written subroutine BND (describing the user problem geometry) and consequently will not be allowed to change during the relaxation procedure.

The second type is a NEUMANN boundary. All or some of the points on the faces of the 3 dimensional box (cylinder) can be of this type (default). The charge density, permittivity (where applicable) and potential on adjacent points orthogonal to such a symmetry plane point (one inside the box and the other a "fictitious" one outside the box) are assumed to be equal and hence the normal gradients of charge density, permittivity and potential are zero for these symmetry boundary points. Thus equipotential lines always cross perpendicular to NEUMANN boundaries.

The third type is a dielectric boundary between two media of different permittivity (currently restricted to 2D problems). Although dielectric boundaries must be restricted to lie on the grid points their precise position is not very important to the fields a few mesh points away, only the existence of the boundary having the dominant effect.

Although restricting the boundaries to lie on the grid (mesh) points limits the resolution with which one can describe a particular geometry (given the maximum number of allowed grid points) this greatly simplifies the program structure and consequently its execution speed.

4) Rate of Convergence

The rate of convergence to where the sum over all mesh nodes of the absolute fractional value change is reduced below a user specified tolerance depends strongly on the over-relaxation factor and types of boundaries describing the geometry, Dirichlet boundaries being much better than Neumann type. A lot of time can be saved by making some initial guess, no matter how crude, of the potentials at each mesh point which is free to vary.

5) Variable Mesh Size

To speed up convergence the initial iteration can be done on a coarse mesh using the "REDU" command. Expansion to a finer mesh ("EXPA") by doubling the number of mesh points in each dimensions is simply accomplished by linear interpolation (this being of the same accuracy as the difference equations). This process is continued until a sufficiently fine mesh is attained (the initial specified mesh) over the region of particular interest. To improve convergence successive sweeps over the volume are performed in alternate directions.

The intrinsic error is proportional to the second power of the grid spacing. Where the solution of one problem can be used as the starting point for a similar problem much computational time can be saved.

A "typical" Laplace problem requires a few hundred sweeps over the volume to converge to <1% error (although in some cases involving largely Neumann boundary conditions or non-zero ρ several thousand sweeps may be required). For a 20,000 point grid a single sweep over all points takes about 1.0 sec on the VAX/780. The CPU time for other problem sizes will be in direct proportion.

6) Friendly Features

Some friendly features built into RELAX3D are:

- a) Extensive HELP facility.
- b) Reprompts on bad user data making program virtually "bomb" proof.
- c) Ability to abort to main menu, e.g. contour plotting may be terminated at any stage.
- d) Display or write to a file the status of a node, a line of nodes, or a box of nodes. This feature is particularly useful in monitoring convergence in critical regions.
- e) Single command (ITER) to automatically converge to a user specified residue.
- f) Save or restore the problem so as to be able to continue at some later date
- g) User "hooks" to BND routine for more sophisticated control of boundaries (e.g. explore multiple geometries).

7) Examples

The following is an example of the "textbook" Poisson equation

$\nabla^2 \phi = xze^y$ in a 3 dim. region with Dirichlet boundary conditions that

$\phi = xze^y$ at the edges of the region.

The listing of the corresponding BND subroutine is shown below.

```

      SUBROUTINE BND2 (I,J,K,PHI,RHO,LL)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C      BND SUBROUTINE FOR RELAX3D                                     C
C                                                                 C
C      TO SOLVE THE EQUATION DEL**2(PHI)=X*Z*EXP(Y)                 C
C                                                                 C
C      WITH PHI=X*Z*EXP(Y) ON THE BOUNDARIES.                       C
C                                                                 C
C      EXACT SOLUTION IS PHI=X*Z*EXP(Y)                             C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

C==Initial values for interior of region:
 PHI=1.0

```

C==Dimensions of problem
      IMAX=21
      JMAX=21
      KMAX=21
C==Grid spacing:
      HX=.100000
      HY=.200000
      HZ=.400000
      X=(FLOAT(I)-1.0)*HX
      Y=(FLOAT(J)-1.0)*HY
      Z=(FLOAT(K)-1.0)*HZ
C
      RHO=X*Z*EXP(Y)
      IF(I.EQ.1.OR.J.EQ.1.OR.K.EQ.1.OR.
#      I.EQ.IMAX. OR. J.EQ.JMAX .OR. K.EQ.KMAX) PHI=-(X*Z*EXP(Y))
C===Freeze zeros at boundary by setting small negative:
      IF(PHI.EQ.0.0)PHI=-1.0E-25
C==Exact solution:
      IF(LL.EQ.1)PHI=X*Z*EXP(Y))
      RETURN
      END

```

A simple parallel plates capacitor (infinitely repetitive) having voltage on the facing plates of +1.0 and -1.0 volts would have median plane symmetry so we need only examine the upper half. Actually we need only look at one quarter, but to test program symmetry we include entire upper half.

The user coded routine describing this geometry would be

```

      SUBROUTINE BND (I,J,K,PHI,RHO,LL)
      PHI=0.5
      IF (K .EQ. 1) PHI=-.000001 (freeze median plane)
      IF( (I.LT.12).OR.(I.GT.22).OR.(J.LT.12).OR.(J.GT.22) ) RETURN
C set voltage on a plate
      PHI=-1.
      RETURN
      END

```

The commands to the program to "solve" this problem and contour plot the J=16 slice were

```

33 33 21      (specify problem size)
OPT 1         (specify this is a Laplace problem)
1 1 1         (specify hx=hy=hz=1)
LOAD          (load the object module describing geometry)
INIT          (initialize problem by calling BND)
REDU          (reduce to coarse grid)
ITER          (relax coarse grid)
EXPA          (expand to fine grid)
ITER          (relax fine solution)
PLOT          (plot contours, desired list follows)
16 J          (specify which slice)
0             (terminate slice requests)

```

The equipotentials plot shown in Fig. 2 would then be produced.

Fig. 2

	3D Cartesian	2D Cartesian	2D Cylindrical
LAPLACE	Yes	Yes	Yes
LAPLACE Dielectric	No	Yes	No
POISSON	Yes	Yes	Yes

TABLE 1 RELAX3D Capabilities

References

1. K.J. Binns, P.J. Lawrenson, "Analysis and Computation of Electric and Magnetic Field Problems", Chap.11,
2. Leon Lapidus, George F. Pinder, "Numerical Solution of Partial Differential Equation in Science and Engineering", Chap. 5, John Wiley & Sons (1982).
3. H. Houtman, TRIUMF Design Note (unpublished) TRI-DN-73-22 (1973).
4. D.L. Nelson, "Relaxation Code", University of Maryland, Technical Report 960 (1969).
5. R.J. Louis "A Guide to Relaxing at U.B.C.", TRIUMF Design Note (unpublished) TR-DN-69-13 (1970).
6. C.F. Gauss, Brief an Gerling, Werke 9, 278-281 (December 26, 1823).

Acknowledgment

The authors would like to thank Fred Jones for making significant upgrades to the program.