LBL-33868
UC-410

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

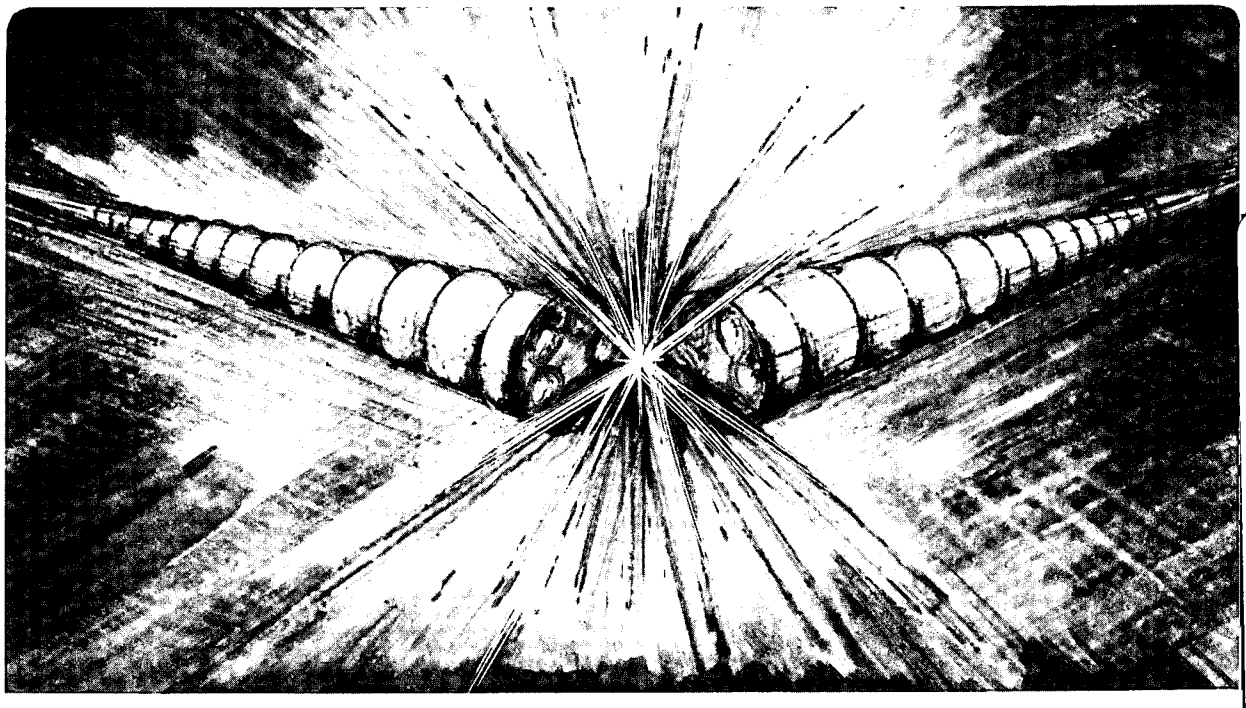
Accelerator & Fusion Research Division

Presented at the Computational Accelerator Physics Conference, Pleasanton, CA, February 22–23, 1992, and to be published in the Proceedings

Simulating a Singularity-Free Universe Outside the Problem Boundary in Poisson

K. Halbach and R. Schlueter

January 1992



REFERENCE COPY
Does Not
Circulate
Bldg. 50 Library.
LBL-33868
Copy 1

Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. Neither the United States Government nor any agency thereof, nor The Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or The Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or The Regents of the University of California and shall not be used for advertising or product endorsement purposes.

Lawrence Berkeley Laboratory is an equal opportunity employer.

This publication has been reproduced from the best available copy

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL-33868
UC-410
LSGN-138

SIMULATING A SINGULARITY-FREE UNIVERSE OUTSIDE THE PROBLEM BOUNDARY IN POISSON

KLAUS HALBACH AND ROSS SCHLUETER

ACCELERATOR and FUSION RESEARCH DIVISION
Lawrence Berkeley Laboratory
University of California
Berkeley, CA 94720

JANUARY 1992

This work was supported by the Director, Office of Energy Research, Office of Basic Energy Sciences, Materials Sciences Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

recycled paper

Simulating a Singularity-Free Universe Outside the Problem Boundary in POISSON

Klaus Halbach and Ross Schlueter

January 30, 1992

Abstract

An exact analytical solution developed from the Dirichlet problem exterior to a circle is employed in the magnetostatics code POISSON[1] to provide a boundary condition option which simulates a singularity-free universe external to the problem domain. Problems with domains of large unequal extents in perpendicular directions are treated by first conformally mapping the exterior of an ellipse onto the exterior of the unit circle. Problems exhibiting symmetry in one or two planes are modeled using a semi-circle or quarter-circle, respectively, in conjunction with the singularity-free rest-of-universe boundary condition.

1 Introduction

Currently POISSON allows for a fixed potential or a zero normal derivative of the potential on problem boundaries. The analysis of many magnetostatics problems would benefit from the ability to simulate a singularity-free universe external to the problem domain. Here, the analytical development and numerical adaptation for POISSON of a free-floating potential boundary condition that simulates a singularity-free rest-of-the-universe is described. An approximate technique addressing this general problem has been already been developed[2] whereby the harmonic description of the potential function on an arc just interior to the problem domain's boundary is used to approximate and update the boundary potentials in a relaxation procedure. Herein, a solution is developed from the Dirichlet problem exterior to a circle which is analytical and exact (except for the discrete nature of the mesh); no approximations are employed. Problems with domains of large unequal extents in perpendicular directions may also be handled by first conformally mapping the exterior of an ellipse onto the exterior of a circle; the analytical development and implementation in POISSON are treated below. Problems exhibiting symmetry in one or two planes may be modeled using a semi-circle or quarter-circle, respectively, in conjunction with the singularity-free rest-of-universe boundary condition.

The complete analytical and numerical procedure of the next two sections can be summarized as follows, starting with the simulation of a singularity-free universe outside a circle, to be followed by a general description of the process for outer boundaries other than circles: POISSON is instructed to make a mesh with meshpoints on a circular boundary and with a set of meshpoints just outside that circle, connected to the points on the circle by regular point-to-point meshlines. Subject to the condition that one has only vacuum outside that circle, we derive below an exact, and fortunately, linear relationship between

the vector potentials on the circle and both the scalar and vector potentials at any point outside that circle. The matrix that allows the calculation of the vector potential for the points just outside the circle from the value of the vector potential of the points located on the circle is then computed. The program then goes through a modified iteration procedure to satisfy the Poisson equation everywhere: first, using current vector potential values of the meshpoints outside the circle, it calculates 'new' values of the vector potential for all points on and inside the circle; then, using the above mentioned matrix, it calculates 'new' values of the vector potential for the meshpoints outside the circle from those on the circle. The program continues this two-step cycle until the solution has converged.

If an elliptical boundary can better accommodate the geometry under investigation, the procedure is identical to that of the case of a circular boundary, with one additional step: After the meshpoints on and just outside the ellipse have been generated, a coordinate transformation is used that maps the ellipse and the points just outside the ellipse conformally (without any singular point on or outside the ellipse) onto a circle and points just outside the circle, respectively. The matrix relating the vector potential at the points on the ellipse to the points just outside the ellipse is then calculated as above. If one wants to use yet a different boundary, an appropriate conformal map to a circle and points outside the circle must be used. So far, the ellipse is the only non-circular boundary that has been implemented, but the program is set up to use any user-provided conformal map.

2 The Dirichlet Problem Outside a Circle

In this section an expression is derived for the complex magnetic potential $F(z)$ outside a circle from the given vector (or alternatively scalar) potential on the circle when there are no external singularities. This problem is analogous to that of determining the complex potential at locations in a singularity-free region inside a circle from the values of the vector potential on the bounding circle, found in many complex analysis texts.

Outside a circle the complex potential can be expressed by a Laurent series expansion:

$$F(z) = A + iV = A_\infty + iV_\infty + \sum_{n=1}^{\infty} \frac{a_n}{z^n} \tag{1}$$

where $z \equiv x + iy$ and the a_n are complex constants.

Taking the real part of Eq. (1) gives for the vector potential on a circle of radius r_0:

$$A(r_0, \theta) = A_\infty + \sum_{n=1}^{\infty} \left\{ \frac{Re(a_n)\cos(-n\theta)}{r_0^n} - \frac{Im(a_n)\sin(-n\theta)}{r_0^n} \right\} \tag{2}$$

Expressing the trigonometric functions in exponential form gives:

$$A(r_0, \theta) = A_\infty + \frac{1}{2}\sum_{n=1}^{\infty} \left\{ \frac{Re(a_n)(e^{-in\theta} + e^{in\theta})}{r_0^n} + i\frac{Im(a_n)(e^{-in\theta} - e^{in\theta})}{r_0^n} \right\} \tag{3}$$

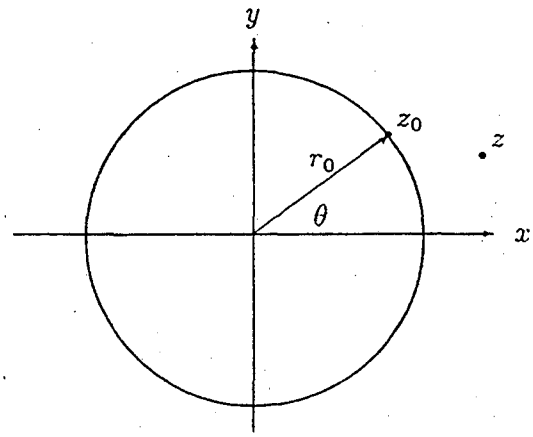


Figure 1. Geometry for Dirichlet problem outside a circle

The vector potential at infinity A_∞ and the complex constants a_n can be determined by multiplying both sides of Eq. (3) by $r_0^m e^{im\theta}$ ($\equiv z_0^m$), where m is a nonnegative integer, and integrating over θ from 0 to 2π.

For $m = 0$ we have:

$$A_\infty = \frac{1}{2\pi}\int_0^{2\pi} A(r_0,\theta)d\theta \tag{4}$$

For m a positive integer we have:

$$\int_0^{2\pi} A(r_0,\theta)r_0^m e^{im\theta}d\theta = \int_0^{2\pi} A_\infty r_0^m e^{im\theta}d\theta +$$

$$\frac{1}{2}\sum_{n=1}^{\infty}\left\{\int_0^{2\pi}\frac{Re(a_n)}{r_0^{n-m}}(e^{i(m-n)\theta}+e^{i(m+n)\theta})d\theta + i\int_0^{2\pi}\frac{Im(a_n)}{r_0^{n-m}}(e^{i(m-n)\theta}-e^{i(m+n)\theta})d\theta\right\} \tag{5}$$

All terms on the right hand side of Eq. (5) are identically zero except those terms in the summation for which $n = m$:

$$\int_0^{2\pi} A(r_0,\theta)r_0^m e^{im\theta}d\theta = \frac{1}{2}2\pi\{Re(a_m) + iIm(a_m)\} = \pi a_m \tag{6}$$

Substituting the expressions derived for A_∞ and a_n into Eq. (1) we have for $F(z)$ outside the circle:

$$F(z) = \frac{1}{2\pi}\int_0^{2\pi} A(r_0,\theta)d\theta + iV_\infty + \frac{1}{\pi}\sum_{n=1}^{\infty}\int_0^{2\pi} A(r_0,\theta)\frac{z_0^n}{z^n}d\theta, \qquad |z| > |z_0| \tag{7}$$

V_∞ can be set to zero without loss of generality. Using a closed form expression for the infinite geometrical series $\sum_1^\infty (z_0/z)^n$ results in the following expression for the complex potential $F(z)$ outside a circle in terms of the known vector potential on the circle:

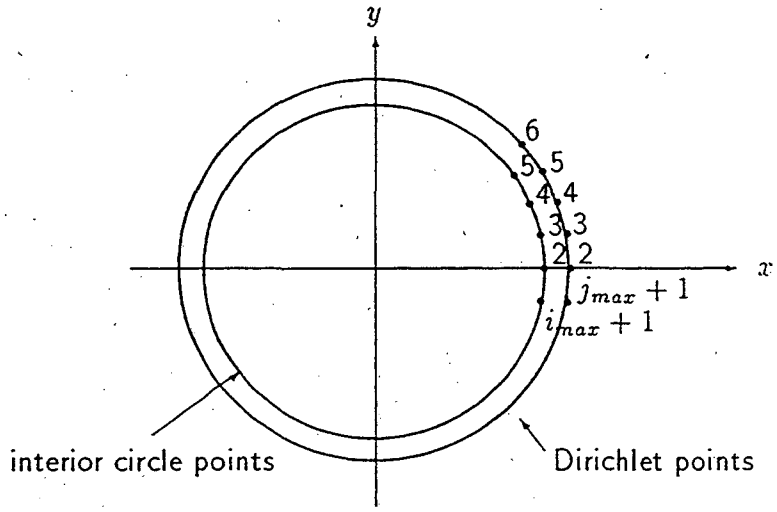


Figure 2. Mesh discretization and indices

$$F(z) = A_\infty + \frac{1}{\pi}\int_0^{2\pi} A(r_0,\theta)\frac{z_0}{z-z_0}d\theta = A_\infty + \frac{1}{\pi}\int_0^{2\pi} A(r_0,\theta)\frac{r_0 e^{i\theta}}{re^{i\phi}-r_0 e^{i\theta}}d\theta \qquad (8)$$

where $z \equiv re^{i\phi}$ is a point outside the circle, $z_0 \equiv r_0 e^{i\theta}$ is a point on the circle and $A_\infty = \frac{1}{2\pi}\int_0^{2\pi} A(r_0,\theta)d\theta$.

3 Numerical Discretization of Analytical Solution to Dirichlet Problem Outside a Circle

Here we derive a discretized expression relating the known vector potentials A_{int_i}, $2 \leq i \leq i_{max}+1$ of a finite set of points $z_{int_i} \equiv r_0 e^{i\theta_i}$ on a[n interior] circle to the vector potentials A_{dir_j} $2 \leq j \leq j_{max}+1$ of a finite set of [Dirichlet] points $z_{dir_j} \equiv r_j e^{i\phi_j}$ exterior to that circle. This discretized expression takes the form of a i_{max} by j_{max} matrix M which is independent of the values of the vector potentials (i.e. need be determined only once):

$$\vec{A}_{dir} = [M]\vec{A}_{int} \qquad (9)$$

We interpolate the vector potential $A(r_0,\theta)$ between known vector potentials on the interior circle [1] [2] :

[1] A simpler interpolation expression $A(r_0,\theta) = A_i + \frac{(A_{i+1}-A_i)(\theta-\theta_i)}{\theta_{i+1}-\theta_i}$ is not employed because the linear term in θ, when substituted into the integral of Eq. (8) cannot be integrated in closed form. This expression would require either numerical integration or an approximation amenable to closed form integration to be made. An alternative interpolation expression $A(r_0,\theta) = A_i + \frac{(A_{i+1}-A_i)(\sin(\theta-\theta_i))}{\sin(\theta_{i+1}-\theta_i)}$ when substituted in the integral of Eq. (8) can be integrated in closed form. The interpolation expression given in Eq. (10) also allows closed form integration of Eq. (8) and is symmetric with respect to (A_{int_i},θ_i) and $(A_{int_{i+1}},\theta_{i+1})$. Finally note that the three interpolation expressions discussed are equivalent in the limit $\Delta\theta_i \equiv \theta_{i+1}-\theta_i \longrightarrow 0$. Errors introduced into the solution are of exactly the same order as those already present in POISSON where the potential is assumed linear between nodes.

[2] Another attractive interpolation technique is to recast Eq. (8) in the form $F(z) = A_\infty + \frac{1}{i\pi}\oint_0^{2\pi} A(z_0)\frac{dz_0}{z-z_0}$ with

$$A(r_0,\theta) = A_{int_i}\left[\frac{\sin(\theta-\theta_{i+1})}{\sin(\theta_i-\theta_{i+1})}\right] + A_{int_{i+1}}\left[\frac{\sin(\theta-\theta_i)}{\sin(\theta_{i+1}-\theta_i)}\right] \tag{10}$$

where $\theta_i \le \theta \le \theta_{i+1}$.

Substituting the interpolated expression for $A(r_0,\theta)$ into Eq. (8), expanding the trigonometric functions in exponential form, integrating, and simplifying expressions yields:

$$\frac{1}{2\pi}\int_0^{2\pi} A(r_0,\theta)d\theta = \frac{1}{2\pi}\sum_{i=2}^{i_{max}+1} A_{int_i}\left\{\tan\left(\frac{\Delta\theta_i}{2}\right) + \tan\left(\frac{\Delta\theta_{i-1}}{2}\right)\right\} \tag{11}$$

and

$$\frac{1}{\pi}\int_0^{2\pi} A(r_0,\theta)\frac{r_0e^{i\theta}}{re^{i\phi}-r_0e^{i\theta}}d\theta \equiv f(\theta_{i-1},\theta_i,\theta_{i+1},z_{dir_j}) = \frac{1}{2\pi}\sum_{i=2}^{i_{max}+1} A_{int_i}$$

$$\left\{\left(\frac{1}{\sin(-\Delta\theta_i)}\right)\left(1-e^{-i\Delta\theta_i} - i(-\Delta\theta_i)B^{-1}_{i+1,j} + \ln[D_{i,j}]^{[B_{i+1,j}-B^{-1}_{i+1,j}]}\right) + \right. \tag{12}$$

$$\left.\left(\frac{-1}{\sin(\Delta\theta_{i-1})}\right)\left(1-e^{i\Delta\theta_{i-1}} - i\Delta\theta_{i-1}B^{-1}_{i-1,j} + \ln[D_{i-1,j}]^{[B^{-1}_{i-1,j}-B_{i-1,j}]}\right)\right\}$$

where

$$D_{i,j} \equiv \frac{z_{dir_j}-z_{int_{i+1}}}{z_{dir_j}-z_{int_i}}, \qquad B_{i,j} \equiv \frac{z_{dir_j}}{z_{int_i}}, \qquad z_{dir_j} \equiv r_je^{i\phi_j}, \qquad z_{int_i} \equiv r_0e^{i\theta_i},$$

$$\Delta\theta_i \equiv \theta_{i+1}-\theta_i, \qquad 2 \le i \le i_{max}+1, \qquad 2 \le j \le j_{max}+1$$

The vector potential A_{dir_j} is then:

$$A_{dir_j} = \frac{1}{2\pi}Re\left\{\sum_{i=2}^{i_{max}+1} A_{int_i}\left[\tan\left(\frac{\Delta\theta_i}{2}\right) + \tan\left(\frac{\Delta\theta_{i-1}}{2}\right) + f(\theta_{i-1},\theta_i,\theta_{i+1},z_{dir_j})\right]\right\} \tag{13}$$

and thus the matrix M of Eq. (9) is given by

$$M(i,j) = \frac{1}{2\pi}Re\left\{\tan\left(\frac{\Delta\theta_i}{2}\right) + \tan\left(\frac{\Delta\theta_{i-1}}{2}\right) + f(\theta_{i-1},\theta_i,\theta_{i+1},z_{dir_j})\right\} \tag{14}$$

where $\Delta\theta_i$, $f(\theta_{i-1},\theta_i,\theta_{i+1},z_{dir_j})$, and z_{dir_j} are defined above.

---

the integration path on the interior circle. Approximating this integration path by that of a polygon passing through all node points on the interior circle and interpolating $A(z)$ along each line segment by $A(z) = A_i + \frac{(A_{i+1}-A_i)(z-z_i)}{z_{i+1}-z_i}$, where z lies on the line segment with endpoints z_i and z_{i+1}, gives closed expressions for all integrals.

POISSON has been modified[3] to allow for free-floating 'Dirichlet boundary points' to simulate a singularity-free universe outside problem boundaries. Given an initial assumed set of potentials on a circular Dirichlet outer boundary, Poisson's equation is [indirectly] solved in the usual manner as given in [1], yielding vector potential A values at all interior points, including the 'interior circle points'. Then the vector potential solution on the set of 'interior circle points' is used along with the matrix of Eq. (14) to reset the potentials on the outer 'Dirichlet boundary points'. The procedure is iterated until both the relaxed solution over the entire problem domain for the given current set of Dirichlet potentials converges and the iterated Dirichlet potentials themselves converge.

4 Running POISSON Using Boundary Condition Option Simulating a Singularity-Free Universe Outside Problem Boundaries

To run POISSON using the boundary condition option simulating a singularity-free universe outside problem boundaries:

- Make an AUTOMESH file (e.g. *fcirc.dat*; see Figure 3) with a circular boundary as region 1. Region 2 must be a circular boundary just inside (i.e. a distance $\sim$ DX) region 1's boundary. To assure accuracy of the numerically discretized equation, DX must be such that angular spacing of nodes on region 2's boundary is $\sim$ 10 degrees or less.
- Run AUTOMESH
- Run LATTICE with *14 1 *58 .001 *108 5 *22 0 etc. (e.g. *49 2) S. Also, user may want to set *32 -1 to get mesh coordinates (x, y)=f(k, l) in *outlat.lis* and *45 LMAX to get vector potential values at all mesh points in *outpoi.lis*.
- Run TEKPLOT with M, 0 1 0 0 S, S, GO to view mesh (See Figure 4).
- Run POISSON (with e.g. 0 Dump, S, 6 10 1., 6 6 -1., -1 S). Here Dirichlet points are free floating.
- Run TEKPLOT with M, 1 0 29 S, S, GO to view B-field lines (See Figure 5).

New CON array variables defined for this new boundary condition case are:

- CON(14) = IDIRICH = 1 for singularity-free universe outside problem boundaries
- CON(58) = EPSILD = 0.001 (default) = convergence criterion for Dirichlet points
- CON(108)= ICYCLED_MAX = 3 (default) = initial number of IVERGE iteration cycles between resetting of Dirichlet boundary points

Also, the rate of convergence ETADIR is set to 1.0 initially, overrelaxation factor RHODIR is set to 1.9 initially, and the spectral norm of the iteration matrix is calculated to optimize RHODIR thereafter.

```
001 full circular arc
$REG NREG=2, DX=.2, XMIN = -1.2, XMAX=1.2, YMIN =  -1.2, YMAX=1.2,  NPOINT=5,
IBOUND=0$
$PO R=   1.2    , THETA=  0.     $
$PO R=   1.2    , THETA= 90.     , X0=0., Y0=0., NT=2$
$PO R=   1.2    , THETA=180.     , X0=0., Y0=0., NT=2$
$PO R=   1.2    , THETA=270.     , X0=0., Y0=0., NT=2$
$PO R=   1.2    , THETA=360.     , X0=0., Y0=0., NT=2$
$REG MAT=1, NPOINT=5$
$PO R=   1.0    , THETA=  0.     $
$PO R=   1.0    , THETA= 90.     , X0=0., Y0=0., NT=2$
$PO R=   1.0    , THETA=180.     , X0=0., Y0=0., NT=2$
$PO R=   1.0    , THETA=270.     , X0=0., Y0=0., NT=2$
$PO R=   1.0    , THETA=360.     , X0=0., Y0=0., NT=2$
```

Figure 3. Sample AUTOMESH file *fcirc.dat* for a circular domain

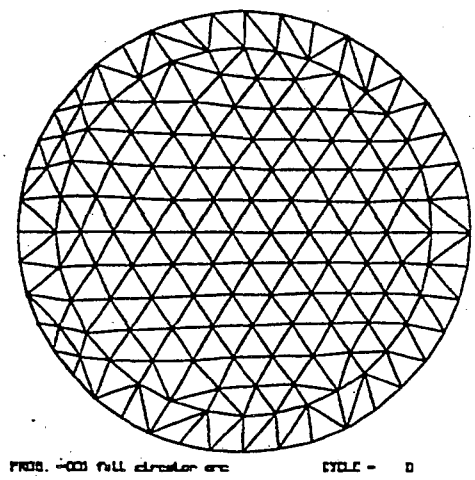


Figure 4. Sample problem mesh for a circular domain

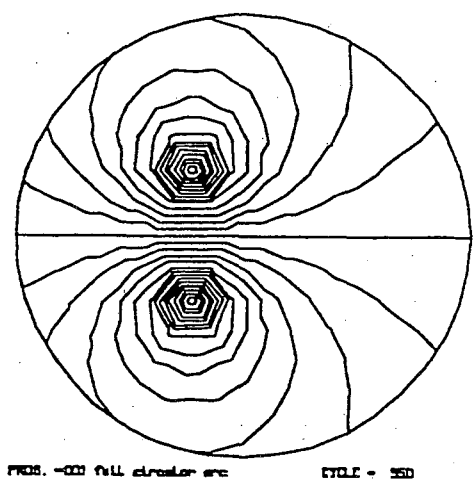


Figure 5. Sample field line plot for a circular domain problem (two current filaments, $I_1 = -I_2$) with a singularity-free external universe

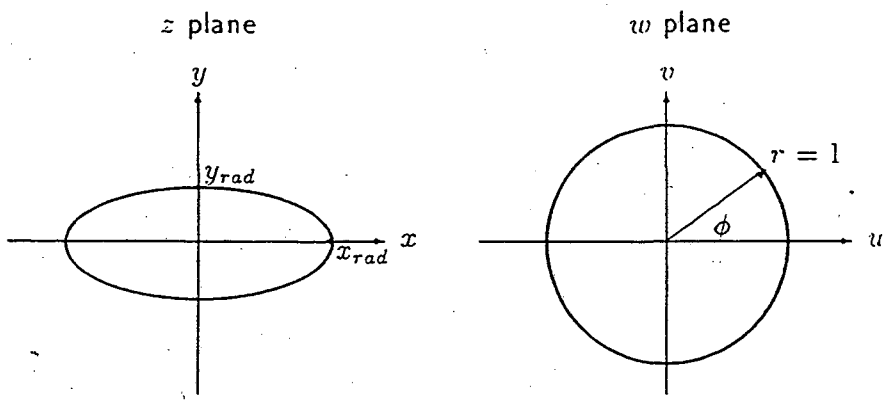


Figure 6. Ellipse in z plane mapped to unit circle in w plane

Additional input information for POISSON is written to [preexisting] file *dirich_in.dat* when LATTICE is run with IDIRICH=1. This file lists Dirichlet k, l and x, y values and interior circle k, l and x, y values. It is a supplemental input file to POISSON.

Additional output information is written to [preexisting] file *dirich_out.dat* when POISSON is run. This file lists the matrix relating vector potentials of Dirichlet points to those of interior circle points, and the POISSON converged vector potential values of Dirichlet and interior circle points.

Appendix A lists the Fortran code additions to LATSO, POISO, and LIBSO to allow for the new boundary condition simulating a singularity-free universe outside problem boundaries.

Appendix B lists a test series to check the accuracy of the newly incorporated POISSON boundary condition option.

5 Mapping the Region Outside an Ellipse to the Region Outside a Circle

An ellipse in the $z(x, y)$ plane can be represented by:

$$z = x_{rad} \cos\phi + i y_{rad} \sin\phi. \qquad 0 \leq \phi \leq 2\pi \tag{15}$$

(ϕ is not to be confused with the angular location θ of a point on the ellipse with respect to the x-axis.)

Introducing a complex w plane through the polar representation

$$w = \rho e^{i\phi}, \tag{16}$$

the ellipse and its exterior in the z plane can be conformally mapped onto the unit circle and its exterior in the w plane with

$$z = x_{rad}\left(\frac{w + w^{-1}}{2}\right) + y_{rad}\left(\frac{w - w^{-1}}{2}\right). \tag{17}$$

All singular points of this map (given by $dz/dw = 0$ or $dw/dz = 0$) are clearly located inside the unit circle in the w plane:

$$dz/dw = 0 \Longrightarrow 0 = x_{rad}(1 - \frac{1}{w^2}) + y_{rad}(1 + \frac{1}{w^2}) \Longrightarrow w^2 = \frac{x_{rad} - y_{rad}}{x_{rad} + y_{rad}}. \tag{18}$$

Thus the exterior of an ellipse in the z plane is indeed mapped to the exterior of the unit circle in the w plane.

POISSON has been modified[3] to allow for boundaries of the shape of an ellipse. This feature is especially useful for simulating a singularity-free universe outside problem boundaries when the problem domain itself is longer in one direction than in a perpendicular direction. In such cases an ellipse just interior to the problem's exterior boundary is first mapped onto a unit circle in the w plane and the exterior boundary itself is also mapped to the w plane (not necessarily onto a circle). Since the map is conformal, the vector potentials of the interior circle in w are identical to those of the interior ellipse in z and likewise for the exterior boundaries in w and in z. The matrix M of Eq. (14) is calculated and the iterative procedure described previously for solving Poisson's equation, with the free-floating Dirichlet potentials on the exterior boundary simulating the singularity-free external universe, is executed.

6 Running POISSON with an Elliptical Problem Domain

To run POISSON with an elliptical problem domain and using the boundary condition option simulating a singularity-free universe outside problem boundaries:

- Make an AUTOMESH file (e.g. *felli.dat*; see Figure 7) with an elliptical boundary as region 1. Region 2 must be an elliptical boundary just inside (i.e. a distance $\sim$ DX) region 1's boundary. To assure accuracy of the numerically discretized equation, DX must be such that angular spacing of nodes on region 2's boundary is $\sim$ 10 degrees or less.
- Run AUTOMESH
- Run LATTICE with *14 1 *58 .001 *108 5 *22 0 etc. (e.g. *49 2) S. Also, user may want to set *32 -1 to get mesh coordinates (x, y)=f(k, l) in *outlat.lis* and *45 LMAX to get vector potential values at all mesh points in *outpoi.lis*.
- Run TEKPLOT with M, 0 1 0 0 S, S, GO to view mesh (See Figure 8).
- Run POISSON (with e.g. 0 Dump, S, 6 10 1., 6 6 -1., -1 S). Here Dirichlet points are free floating.
- Run TEKPLOT with M, 1 0 29 S, S, GO to view B-field lines (See Figure 9).

```
001 full ellipse
$REG NREG=2, DX=.10, XMIN = -2.0, XMAX=2.0, YMIN =  -1.2, YMAX=1.2,  NPOINT=5,
IBOUND=0$
$PO R=   2.0     , THETA=  0.      $
$PO R=   1.0     , THETA= 90.      , X0=0., Y0=0., NT=4$
$PO R=   2.0     , THETA=180.      , X0=0., Y0=0., NT=4$
$PO R=   1.0     , THETA=270.      , X0=0., Y0=0., NT=4$
$PO R=   2.0     , THETA=360.      , X0=0., Y0=0., NT=4$
$REG MAT=1, NPOINT=5$
$PO R=   1.80    , THETA=  0.      $
$PO R=   0.80    , THETA= 90.      , X0=0., Y0=0., NT=4$
$PO R=   1.80    , THETA=180.      , X0=0., Y0=0., NT=4$
$PO R=   0.80    , THETA=270.      , X0=0., Y0=0., NT=4$
$PO R=   1.80    , THETA=360.      , X0=0., Y0=0., NT=4$
```

Figure 7. Sample AUTOMESH file *felli.dat* for an elliptical domain

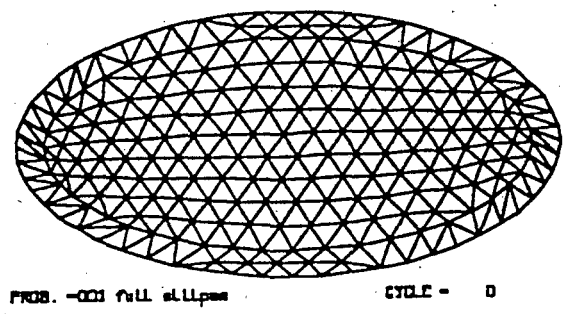


Figure 8. Sample problem mesh for an elliptical domain

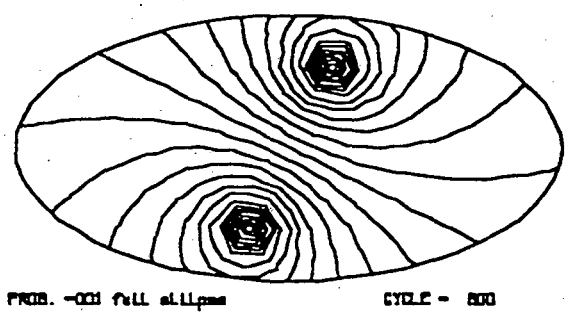


Figure 9. Sample field line plot for an elliptical domain problem (two current filaments, $I_1 = -I_2$) with a singularity-free external universe

New AUTOMESH input option for NT:

- NT = 4 for elliptical curve between current point and previous point

A full ellipse must be entered $\frac{1}{4}$ of an ellipse per arc segment (see *felli.dat*).

Appendix A lists the Fortran code additions to AUTSO and POISO to allow for the new boundary curve shape option and associated conformal maps necessary to implement the new boundary condition option simulating a singularity-free universe outside problem boundaries.

7 Special Features

Special case symmetries for the singularity-free external universe boundary condition have been incorporated[3] into POISSON; quarter-circles or half-circles may be employed to reduce mesh area size and enhance the accuracy of field harmonic component calculations.

For a problem with the Neumann midplane symmetry $A(x,-y) = A(x,y)$, a half-circle may be employed (see Figure 10). Since current sources below the midplane are identical in both magnitude and sign to those above the midplane and the rest of the universe is singularity-free, the net current must be zero in the half-circle problem domain with this symmetry type.

Again, for a problem with the Dirichlet midplane symmetry $A(x,-y) = -A(x,y)$, a half-circle may be employed (see Figure 11-a,b). Since current sources below the midplane are identical in magnitude but opposite in sign to those above the midplane and the rest of the universe is singularity-free, no restriction is imposed on the net current in the half-circle problem domain with this symmetry type (Figure 11-b).

For a problem exhibiting symmetry along both the x- and y-axes, a quarter-circle may be employed. For H-magnet symmetry as shown in Figure 12, i.e. $A(-x,y) = -A(x,y)$ and $A(x,-y) = A(x,y)$, no restriction is imposed on the net current in the quarter-circle problem domain. For an elliptical quadrupole symmetry with Neumann boundary conditions along both the x- and y-axes, i.e. $A(-x,y) = A(x,y)$ and $A(x,-y) = A(x,y)$, the net current must be zero in the quarter-circle problem domain. For the quarter-plane symmetry with Dirichlet boundary conditions along both the x- and y-axes, i.e. $A(-x,y) = -A(x,y)$ and $A(x,-y) = -A(x,y)$, no restriction is imposed on the net current in the quarter-circle problem domain.

To run POISSON with a quarter-circle or half-circle problem domain using the boundary condition option simulating a singularity-free universe outside problem boundaries:

- Make an AUTOMESH file (e.g. *qcirc.dat*; see Figure 13) with a quarter (or half) circle line region as region 1. Region 2 must be a quarter (or half) circle line region just inside (i.e. a distance $\sim$ DX) region 1's boundary. To assure accuracy of the numerically discretized equation, DX must be such that angular spacing of nodes on region 2's boundary is $\sim$ 10 degrees or less.

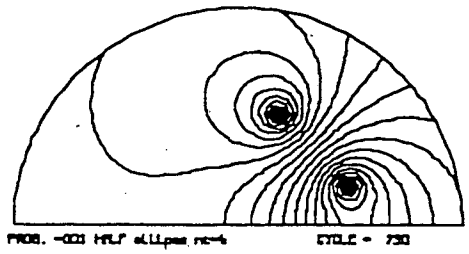


Figure 10. Sample field line plot for midplane symmetry type with $A(x,-y) = A(x,y)$ and a singularity-free external universe

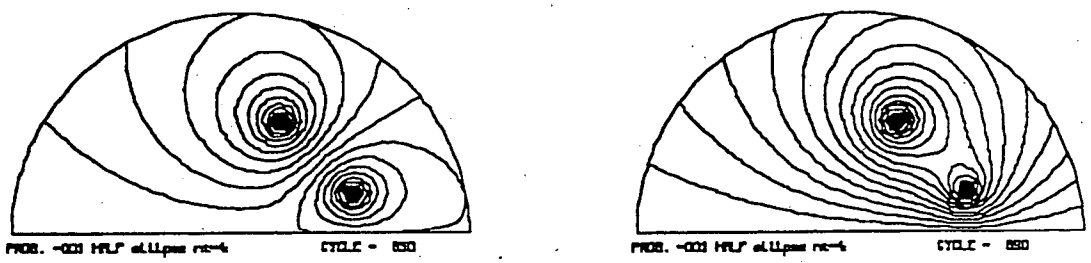


Figure 11. Sample field line plot for midplane symmetry type with $A(x,-y) = -A(x,y)$ and a singularity-free external universe

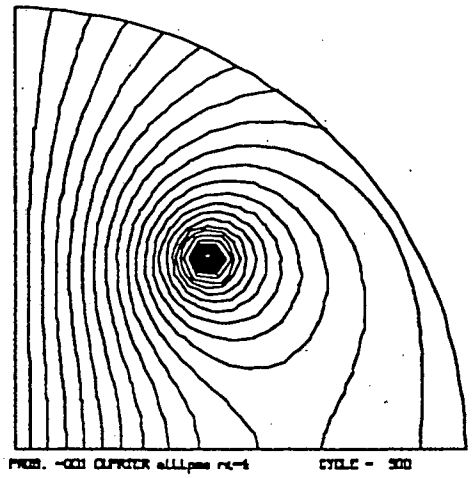


Figure 12. Sample field line plot for H-magnet symmetry type with $A(-x,y) = -A(x,y)$, $A(x,-y) = A(x,y)$, and a singularity-free external universe

```
qcirc2    QUARTER ellipse nt=4
$REG NREG=3, DX=.05, XMIN = 0., XMAX=1.2, YMIN =  0., YMAX=1.2,  NPOINT=2,
IBOUND=0$
$PO R=   1.2    , THETA=  0.     $
$PO R=   1.2    , THETA= 90.     , X0=0., Y0=0., NT=4$
$REG MAT=1, NPOINT=2$
$PO R=   1.0    , THETA=  0.     $
$PO R=   1.0    , THETA= 90.     , X0=0., Y0=0., NT=4$
$REG MAT=1, NPOINT=4, IBOUND=0$
$PO R=   1.2    , THETA=  0.     $
$PO R=   1.2    , THETA= 90.     , X0=0., Y0=0., NT=4$
$PO R=   0.0    , THETA= 90.     $
$PO R=   1.2    , THETA=  0.     $
```

Figure 13. Sample AUTOMESH file *qcirc.dat* for a quarter-circle domain

- continue as outlined in the previous sections

New CON array variable defined for the singularity-free external universe new boundary condition with the above symmetry conditions:

- CON(14) = IDIRICH

  = 2 for singularity-free universe outside a half-circle (or half-ellipse) with midplane symmetry. If midplane symmetry is Neumann with $A(x,-y) = A(x,y)$, then CON(22) = 1. If midplane symmetry is Dirichlet with $A(x,-y) = -A(x,y)$, then CON(22) = 0.

  = 4 for singularity-free universe outside a quarter-circle (or quarter ellipse) with symmetry with respect to both the x- and y-axes. For H-magnet type symmetry, CON(22) = 1 and CON(24) = 0. For Neumann symmetry at both the x- and y-axes, CON(22) = 1 and CON(24) = 1. For Dirichlet symmetry at both the x- and y-axes, CON(22) = 0 and CON(24) = 0.

Appendix A lists the Fortran code additions to LATSO, LIBSO, and POISO to allow for the special case symmetry types in conjunction with the boundary condition option simulating a singularity-free universe outside problem boundaries.

8 Nontrivial POISSON Test Case Checks

Below we check the accuracy of the new singularity-free rest-of-universe boundary condition in POISSON with two test cases: (a) with current filaments in free-space and (b) with current filaments in the presence of infinitely permeable iron, which is computationally difficult, but for which an exact closed form solution is known.

(a) Current filaments in free-space. The complex potential for the current filament distribution shown in Figure 14a is

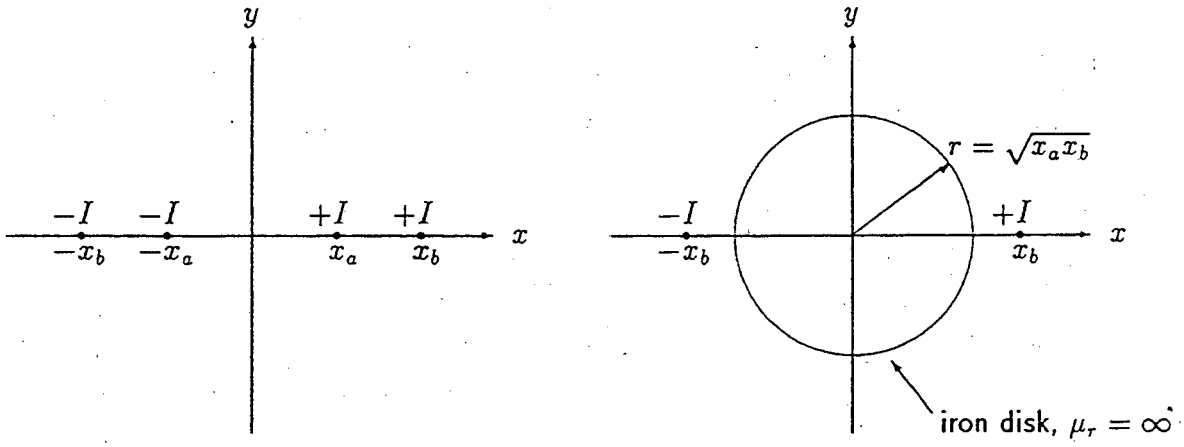


Figure 14a,b. POISSON test case geometries, (a) iron-free and (b) iron present

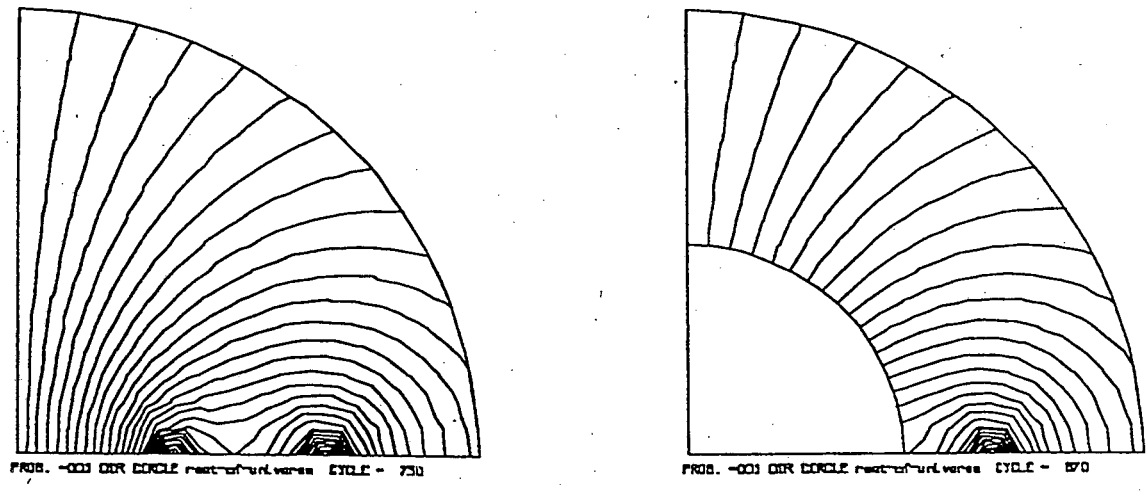


Figure 15a,b. Field line plots using new singularity-free rest-of-universe boundary condition for the geometries of Figure 14a,b.

$$F(z) = \sum_{i=1}^{4} \frac{-\mu_0 I_i}{2\pi} \ln(z - z_i) \tag{19}$$

where $z_1 = (-x_b, 0) = -z_4, z_2 = (-x_a, 0) = -z_3$ and $I_1 = I_2 = -1, I_3 = I_4 = +1$.

A POISSON run for the geometry of Figure 14a, using the quarter-circle H-magnet symmetry of the previous section and the new singularity-free rest-of-universe boundary condition with the singularities input in the first quadrant gives the field line plot shown in Figure 15a. Examination of the corresponding POISSON output file shows that the computationally obtained vector potential A agrees everywhere with the closed form solution given by the real part of $F(z)$ in Eq. (19).

(b) Current filaments in the presence of infinitely permeable iron. It is shown in Appendix C that the imaginary part of $F(z)$ equals zero when either $y = 0$ or $x^2 + y^2 = x_a x_b$. Thus in Figure 14a, the magnetic field outside a radius $\sqrt{x_a x_b}$ will be identical to that produced by the geometry of Figure 14b where an infintely permeable iron circle of radius $r = \sqrt{x_a x_b}$ is centered at $z = 0$ and the location and magnitude of the outermost current filaments are identical to those of Figure 14a.

A POISSON run for the geometry of Figure 14b (using the quarter-circle H-magnet symmetry of the previous section and the new singularity-free rest-of-universe boundary condition) with $\mu_r = \infty$ in the iron and with the singularity input in the first quadrant gives the field line plot shown in Figure 15b, which is identical to that of Figure 15a outside the radius $r = \sqrt{x_a x_b}$. Examination of the corresponding POISSON output file shows that the computationally obtained vector potential A agrees at all locations $r \geq \sqrt{x_a x_b}$ with the closed form solution given by the real part of $F(z)$ in Eq. (19).

9 Literature Cited

1. *Reference Manual for the Poisson/Superfish Group of Codes*, LA-UR-87-126, Los Alamos National Laboratory, Los Alamos, NM, 1987.

2. Caspi, S., et.al., *Incorporation of Boundary Condition into the Program POISSON*, LBL-19172, Aug. 1985.

3. Schlueter, R.D., POISSON version _DSA123:[ROSS.POIS.MODS]POISSON, Lawrence Berkeley Laboratory, 1990.

Appendix A: Fortran code additions to AUTSO, LATSO, LIBSO, and POISO

```
C*********************************************************************
AUTSO MODIFICATIONS

IN MAIN PROGRAM:
.
.
C     n= LOGXeq('AUTOMESH.POI')                                        !RDS
.
.
.
C     PRINT *,  NPOTS, NSTAR, NNF        !DIAGNOSTIC                   !RDS
.
.
.
C     PRINT *,NPOTS, NSTAR, NNB, NNB     !DIAGNOSTIC                   !RDS
.
.
.
IN SUBROUTINE DATUPS:
.
.
.
      IF(NT.EQ.4) GO TO 51                                             !RDS
.
.
.
      GO TO 46                                                         !RDS
C                                                                      !RDS
C    ELLIPSE                                                           !RDS
C                                                                      !RDS
   51 IF(LEGVAR(X1)+LEGVAR(Y1).GT.0) GO TO 52                          !RDS
      X1=XZ1+X1                                                        !RDS
      Y1=YZ1+Y1                                                        !RDS
      THETA1=ANGLE(XZ2,YZ2,X1,Y1,0)                                    !RDS
      GO TO 54                                                         !RDS
   52 IF(LEGVAR(R1)+LEGVAR(THETA1).GT.0) GO TO 65                      !RDS
      IF(THETA1.LT.0.0) THETA1=360.0+THETA1                            !RDS
      THETA1=THETA1*PI/180.0                                           !RDS
      X1=XZ1+R1*COS(THETA1)                                            !RDS
      Y1=YZ1+R1*SIN(THETA1)                                            !RDS
C                                                                      !RDS
   54 IF(LEGVAR(X2)+LEGVAR(Y2).GT.0) GO TO 56                          !RDS
      R2=SQRT(X2**2+Y2**2)                                             !RDS
      X2=XZ2+X2                                                        !RDS
      Y2=YZ2+Y2                                                        !RDS
      THETA2=ANGLE(XZ2,YZ2,X2,Y2,0)                                    !RDS
      GO TO 58                                                         !RDS
   56 IF(LEGVAR(R2)+LEGVAR(THETA2).GT.0) GO TO 65                      !RDS
      IF(THETA2.LT.0.0) THETA2=360.0+THETA2                            !RDS
      THETA2=THETA2*PI/180.0                                           !RDS
      X2=XZ2+R2*COS(THETA2)                                            !RDS
      Y2=YZ2+R2*SIN(THETA2)                                            !RDS
C                                                                      !RDS
   58 R1=SQRT((X1-XZ2)**2+(Y1-YZ2)**2)                                 !RDS
      IF(ABS(R2-R1).GT.0.001*R2) PRINT 158, IREG !GO TO 65             !RDS
  158 FORMAT(' ELLIPSE SEGMENT INPUT FOR BOUNDARY OF REG',I3)          !RDS
C                                                                      !RDS
C    CHECK THAT ARC LENGTH IS LESS THAN OR EQUAL TO PI                 !RDS
C                                                                      !RDS
      IF(ABS(THETA2-THETA1).LE.PI) GO TO 46                            !RDS
      IF(THETA1.GT.THETA2) GO TO 59                                    !RDS
      THETA1=THETA1+2.0*PI                                             !RDS
      GO TO 46                                                         !RDS
   59 THETA2=THETA2+2.0*PI
      GO TO 46                                                         !RDS
```

```
.
.
.
   65 WRITE(6,165)                                                      !RDS
      PRINT  165                                                        !RDS
      GO TO 90                                                          !RDS
.
.
.
  165 FORMAT(/'--ERROR-- DATA FOR ELLIPSE FROM (X1,Y1)/(R1,THET        !RDS
     XA1)'/'            TO (X2,Y2)/(R2,THETA2) IS INCONSISTENT')       !RDS
.
.
.
IN SUBROUTINE DISTAN:
.
.
.
  300 IF(NT.GT.3) GO TO 400                                             !RDS
  301 CSQ=(X(IROW)-X0)**2-Y(IROW)**2                                    !RDS
.
.
.
C                                                                       !RDS
  400 CONTINUE                                                          !RDS
      THETA=ANGLE(X0,Y0,X(IROW),Y(IROW),ITHETA)                         !RDS
      IF(IOVER.EQ.1) GO TO 430                                          !RDS
      IF(THETA.LT.THEMIN .OR. THETA.GT.THEMAX) RETURN                   !RDS
  430 CONTINUE                                                          !RDS
      RAD=45./ATAN(1.)                                                  !RDS
      X1X0=ABS(X1-X0)                                                   !RDS
      Y1Y0=ABS(Y1-Y0)                                                   !RDS
      X2X0=ABS(X2-X0)                                                   !RDS
      Y2Y0=ABS(Y2-Y0)                                                   !RDS
      XRAD=AMAX1(X1X0,X2X0)                                             !RDS
      YRAD=AMAX1(Y1Y0,Y2Y0)                                             !RDS
C    A FUNCTIONAL, IF NOT VERY ELEGANT WAY TO GET DSQ:                  !RDS
      THINK=2.*RAD                                                      !RDS
      DSQ_MIN=1.E10                                                     !RDS
      DO 80 KTH=1,4                                                     !RDS
      DTHET=-5.*1./RAD                                                  !RDS
      THET=FLOAT(KTH-1)*THINK                                           !RDS
      DSQ_OLD=1.E10                                                     !RDS
      DSQ=DSQ_OLD                                                       !RDS
      DO 79 JTH=1,5                                                     !RDS
      DSQ_OLD=DSQ                                                       !RDS
      DTHET=-DTHET/5.                                                   !RDS
      DO 77 ITH=1,92                                                    !RDS
      THET=THET+DTHET                                                   !RDS
      DSQ=(SQRT((X(IROW)-X0-XRAD*COS(THET))**2+                         !RDS
     &          (Y(IROW)-Y0-YRAD*SIN(THET))**2))**2                     !RDS
      IF(DSQ .GT. DSQ_OLD)GOTO 78                                       !RDS
      DSQ_OLD=DSQ                                                       !RDS
   77 CONTINUE                                                          !RDS
   78 DSQ_SAVE=DSQ_OLD                                                  !RDS
   79 CONTINUE                                                          !RDS
      IF(DSQ_SAVE .LT. DSQ_MIN) DSQ_MIN=DSQ_SAVE                        !RDS
   80 CONTINUE                                                          !RDS
      DSQ=DSQ_MIN                                                       !RDS
      DSQDSQ=(SQRT((X(IROW)-X0)**2+(Y(IROW)-Y0)**2)-R)**2               !RDS
C     PRINT *,THETA*360./RAD,XRAD,YRAD,DSQ,X(IROW),Y(IROW),R,DSQDSQ     !RDS
      NTHET=IFIX((THETA2-THETA)/100.)                                   !RDS
      IF(NTHET.EQ.0. .AND. THETA.LT.THETA2)NTHET=1                      !RDS
      IF(NTHET.EQ.0. .AND. THETA.GE.THETA2)NTHET=-1                     !RDS
      NTHETABS=ABS(NTHET)                                               !RDS
      DTHET=(THETA2-THETA)/FLOAT(NTHET)                                 !RDS
```

```
      ARCLEN=0.                                                         !RDS
      THET=THETA-DTHET/2.                                               !RDS
      THET_OLD=THETA                                                    !RDS
      DO 435 NTH=1,NTHETABS                                             !RDS
      THET=THET+DTHET                                                   !RDS
      ARCLEN=ARCLEN+                                                    !RDS
     &SQRT(XRAD*(COS(THET)-COS(THET_OLD))**2+                           !RDS
     &     YRAD*(SIN(THET)-SIN(THET_OLD))**2)                           !RDS
  435 CONTINUE                                                          !RDS
      ENDSQ=ARCLEN**2                                                   !RDS
      RETURN                                                            !RDS
.
.
.
IN SUBROUTINE INTERX:
.
.
.
      IF(NTS(N).EQ.4) GOTO 60                                           !RDS
.
.
.
   60 NI=4                                                              !RDS
      XZI=XZ2S(N)                                                       !RDS
      YZI=YZ2S(N)                                                       !RDS
      X1X0=ABS(X1S(N)-XZI)                                              !RDS
      Y1Y0=ABS(Y1S(N)-YZI)                                              !RDS
      X2X0=ABS(X2S(N)-XZI)                                              !RDS
      Y2Y0=ABS(Y2S(N)-YZI)                                              !RDS
      XRAD=AMAX1(X1X0,X2X0)                                             !RDS
      YRAD=AMAX1(Y1Y0,Y2Y0)                                             !RDS
      XI=XREG                                                           !RDS
      PHI=ACOS(XI/XRAD)                                                 !RDS
      DY=YRAD*SIN(PHI)                                                  !RDS
      YI=YZI+DY                                                         !RDS
      THEMIN=AMIN1(T1S(N),T2S(N))                                       !RDS
      THEMAX=AMAX1(T1S(N),T2S(N))                                       !RDS
      ITHETA=0                                                          !RDS
      IF(THEMAX.GT.2.0*PI) ITHETA=1                                     !RDS
      THETAI=ANGLE(XZI,YZI,XI,YI,ITHETA)                                !RDS
      IF(THETAI.GE.THEMIN .AND. THETAI.LE.THEMAX) GO TO 200             !RDS
      YI=YZI-DY                                                         !RDS
      THETAI=ANGLE(XZI,YZI,XI,YI,ITHETA)                                !RDS
      GO TO 200                                                         !RDS
.
.
.
IN SUBROUTINE MINMAX:
.
.
.
      IF(NT.EQ.4) GO TO 400                                             !RDS
.
.
.
  400 THEMIN=AMIN1(THETA1,THETA2)                                       !RDS
      THEMAX=AMAX1(THETA1,THETA2)                                       !RDS
      ITHETA=0                                                          !RDS
      IF(THEMAX.GT.2.0*PI) ITHETA=1                                     !RDS
      X1X0=ABS(X1-X0)                                                   !RDS
      Y1Y0=ABS(Y1-Y0)                                                   !RDS
      X2X0=ABS(X2-X0)                                                   !RDS
      Y2Y0=ABS(Y2-Y0)                                                   !RDS
      XRAD=AMAX1(X1X0,X2X0)                                             !RDS
      YRAD=AMAX1(Y1Y0,Y2Y0)                                             !RDS
      RXY=(XRAD-YRAD)/(XRAD+YRAD)                                       !RDS
```

```
      SQSEG=                                                            !RDS
     &0.25*PI*(XRAD+YRAD)*(1.+RXY**2/4.+RXY**4/64.+RXY**6/256.)         !RDS
C                          !GOOD ONLY FOR 1/4 ELLIPSE CURRENTLY         !RDS
.
.
.
IN SUBROUTINE XYPOTS:
.
.
.
  300 IF(NT.GT.3) GO TO 400                                             !RDS
  301 DXP=0.5*(X2-X1)/FINTS                                             !RDS
.
.
.
C                                                                       !RDS
C    ELLIPSE                                                            !RDS
C                                                                       !RDS
  400 PI=4.*ATAN(1.)                                                    !RDS
      X1X0=ABS(X1-X0)                                                   !RDS
      Y1Y0=ABS(Y1-Y0)                                                   !RDS
      X2X0=ABS(X2-X0)                                                   !RDS
      Y2Y0=ABS(Y2-Y0)                                                   !RDS
      XRAD=AMAX1(X1X0,X2X0)         !GOOD FOR 1/4 ELLIPSE ONLY          !RDS
      YRAD=AMAX1(Y1Y0,Y2Y0)         !GOOD FOR 1/4 ELLIPSE ONLY          !RDS
      PHI1=THETA1                                                       !RDS
      PHI2=THETA2                                                       !RDS
C     PHI1=ATAN((XRAD/YRAD)*TAN(THETA1))                                !RDS
C     PRINT *,X1,Y1,PHI1,THETA1                   !DIAGNOSTIC           !RDS
C     IF(THETA1.GT.PI/2.)PHI1=PHI1+PI                                   !RDS
C     IF(THETA1.LT.-PI/2.)PHI1=PHI1-PI                                  !RDS
C     PHI2=ATAN((XRAD/YRAD)*TAN(THETA2))                                !RDS
C     PRINT *,X2,Y2,PHI2,THETA2                   !DIAGNOSTIC           !RDS
C     IF(THETA2.GT.PI/2.)PHI2=PHI2+PI                                   !RDS
C     IF(THETA2.LT.-PI/2.)PHI2=PHI2-PI                                  !RDS
      DPHI=(PHI2-PHI1)/FINTS                                            !RDS
      PHI=PHI1                                                          !RDS
C     PRINT *, PHI1,PHI2,DPHI,XRAD,YRAD,N,NPOTS1  !DIAGNOSTIC           !RDS
      DO 410 I=N,NPOTS1                                                 !RDS
      PHI=PHI+DPHI                                                      !RDS
      XR(I)=X0+XRAD*COS(PHI)                                            !RDS
      YR(I)=Y0+YRAD*SIN(PHI)                                            !RDS
C     PRINT *,XR(I),YR(I)                         !DIAGNOSTIC           !RDS
  410 CONTINUE                                                          !RDS
      RETURN                                                            !RDS
.
.
C*********************************************************************

LIBSO MODIFICATIONS

IN SUBROUTINE POTSET:
.
.
      EQUIVLANENCE (IDIRICH,CON(14))                                    !RDS
.
.
.
      IF(IDIRICH.GE.1)CALL MATDIR                                       !RDS
.
.
C*********************************************************************

LATSO MODIFICATIONS
```



```
IN MAIN PROGRAM:
```

```
.
.
.
C     n = logxeq('LATTICE.POI')                                        !RDS
.
.
.
IN SUBROUTIN PBLOCK:
.
.
.
C          (58)=EPSID -- SOURCELESS UNIVERSE CONVERGENCE CRIT.         !RDS
.
.
.
C         (108)=ICYCLED_MAX -- NO. OF IVERGE ITERATION LOOPS           !RDS
C                              BETWEEN UNIVERSE B.C ADJUSTMENT         !RDS
.
.
.
      EQUIVALENCE (CON(1),ICO(1)),(IDIRICH,CON(14))                    !RDS
.
.
.
      CON(58)=0.001  !EPSILD DEFAULT VALUE                             !RDS
      ICO(108)=3    !ICYCLED_MAX DEFAULT VALUE                         !RDS
.
.
.
CCC      ICO(58)=0                                                     !RDS
.
.
.
CCC      CON(108)=0.                                                   !RDS
.
.
.
IN SUBROUTINE REREG:
.
.
.
     X            ,(IDIRICH,CON(14))                                   !RDS
.
.
.
      IF(IDIRICH.GE.1)OPEN(UNIT=7,FILE='DIRICH_IN.DAT',STATUS='OLD')!RDS
.
.
.
      IF(IDIRICH.GE.1.AND.IREG.LE.2)WRITE(7,*) NDATA/4 -1  !JMA,IMA !RDS
.
.
.
      IF(IDIRICH.GE.1.AND.IREG.LE.2)                                   !RDS
     &   WRITE(7,507)IB(J),IB(J+1),B(J+2),B(J+3)                       !RDS
.
.
.
      IF(IDIRICH.GE.1)CLOSE(UNIT=7, STATUS='KEEP')                     !RDS
.
.
.
C*********************************************************************
```

POISO MODIFICATIONS

IN MAIN PROGRAM:

```
.
.
      DOUBLE PRECISION A,GTU,GTL,WT1,WT5,WT6,ZMAT,A_INT          !FC !DP
.
.
.
      COMMON /RDS/ZMAT(200,200),IMA,JMA,JDIR(200),IINT(200),RESIDD, !RDS
     &   JKDUM(500), JLDUM(500), IKDUM(500), ILDUM(500), A_INT(500) !RDS
CCC     &       THETA(200), PHI(200),                               !RDS
.
.
.
     X            ,(IDIRICH,CON(14)),                               !RDS
     X             (EPSILD,CON(58)), (ICYCLED_MAX,CON(108))         !RDS
.
.
.
C     n = logxeq('POISSON.POI')                                     !RDS
.
.
.
      IF(IDIRICH.GE.1)THEN                                         !RDS
         OPEN (unit=7, FILE='DIRICH_IN.DAT', STATUS='OLD')         !RDS
         OPEN (unit=8, FILE='DIRICH_OUT.DAT', STATUS='OLD')        !RDS
C        EPSILD=0.001  IS CON(58)  DEFAULT VALUE, SET IN LATSO     !RDS
C        ICYCLED_MAX=3 IS CON(108) DEFAULT VALUE, SET IN LATSO     !RDS
C        RHODIR=1.9 INITIALLY (SEE SETDIR)                         !RDS
C        ETADIR=1.0 INITIALLY (SEE SETDIR)                         !RDS
      END IF                                                       !RDS
.
.
.
      ICYCLED=0                                                    !RDS
      RESIDD=0.                                                    !RDS
.
.
.
      ICYCLED=ICYCLED+1                                            !RDS
      IF(IDIRICH.GE.1 .AND.ICYCLED.EQ.ICYCLED_MAX)THEN             !RDS
         PRINT 539, ICYCLE,AMIN,AMAX,RESIDA,ETAAIR,RHOAIR,XJFACT   !RDS
         CALL SETDIR                                               !RDS
         IF(RESIDD .LT. .009)ICYCLED_MAX=3                         !RDS
         IF(RESIDD .LT. .003)ICYCLED_MAX=4                         !RDS
         IF(RESIDD .LT. .0007)ICYCLED_MAX=7                        !RDS
         IF(RESIDD .LT. .0001)ICYCLED_MAX=9                        !RDS
         IF(RESIDD .LT. .001)ICYCLED_MAX=10 ! .001 is low enough   !RDS
         ICYCLED=0                                                 !RDS
      END IF                                                       !RDS
.
.
.
      IF(RESIDA.LT.EPSILA .AND. RESIDD.LT.EPSILD) GO TO 220        !RDS
.
.
.
     &   .AND. RESIDD.LT.EPSILD) GO TO 230                         !RDS
.
.
.
C    CONVERGED  ON DIRICHLET POINTS                                !RDS
C                                                                  !RDS
      IF(IDIRICH.GE.1)THEN                                         !RDS
      WRITE(8,489)                                                 !RDS
      IQMA=JMA+1                                                    !FC
      IF(IDIRICH*IMA.GT.JMA) IQMA=IDIRICH*IMA+1                     !FC
```

```
      DO 101 J=2,IQMA                                                   !RDS
      WRITE(8,491) J,JKDUM(J),JLDUM(J),A(JDIR(J)),                      !RDS
     &              IKDUM(J),ILDUM(J),A_INT(J)                          !RDS
101   CONTINUE                                                          !RDS
      CLOSE (UNIT=7, STATUS='KEEP')                                     !RDS
      CLOSE (UNIT=8, STATUS='KEEP')                                     !RDS
      END IF                                                            !RDS
C                                                                       !RDS
.
.
.
  489 FORMAT(//,' J ',1X,'JLDUM JKDUM ',' A_DIRICH ',                   !RDS
     &       'IKDUM ILDUM ','   A_INT   ',/)                            !RDS
  491 FORMAT(I3,1X,2(1X,I3,2X,I3,2X,D10.4,1X))                          !RDS
.
.
```

ADDED SUBROUTINES SETDIR, MATDIR, AND ZTOW:

```
.
.
C***********************************************************************
C
      SUBROUTINE SETDIR                                                 !RDS
C
C    "SETDIR" RESETS THE "DIRICHLET" BOUNDARY POINT POTENTIALS FOR THE
C     CASE OF B.C. SIMULATING REST OF UNIVERSE WITHOUT SOURCES/SINKS AND
C     CALCULATES RESIDUAL FOR DIRICHLET POINT CONVERGENCE
C
C     LOCALLY USED VARIABLE:
C     DIMENSION A_INT(500)
C
      DOUBLE PRECISION A, ZMAT, ADIR, SUMA, SUMDA, ANORM,               !DP
     &                 A_INT, AVG_TWO                                   !FC
      COMMON /A1/A(20000)
      COMMON /PROB/CON(125)
      COMMON /RDS/ZMAT(500,500),IMA,JMA,JDIR(500),IINT(500),RESIDD,
     &   JKDUM(500), JLDUM(500), IKDUM(500), ILDUM(500), A_INT(500) !RDS
      REAL LAMBDAD
C
      EQUIVALENCE (RHOAIR,CON(75)),(ETAAIR,CON(106)),
     X            (OMEGA,CON(52)),(ICYCLE,CON(90)),
     &            (ICYCLED_MAX,CON(108)),
     &            (IHOR,CON(22)), (IVER,CON(24)), (IDIRICH,CON(14))  !FC
C
      PI=4.*ATAN(1.)
C
      IF(ETADIR.EQ.0.)ETADIR=1.
      IF(RHODIR.EQ.0.)RHODIR=1.9
C
C    EXTEND REGION 2'S BOUNDARY VALUES OF A_INT TO FULL ELLIPSE (OR CIRCLE)
C=====================================================================!FC
      do 348 i=2,ima+2
         A_int(i) = A(iint(i))
  348 continue
      ima_fc=ima
      if(idirich.gt.1)then
         ima_fc=ima*idirich
         if(idirich.eq.4)then
            if(iver.eq.1 .and. ihor.eq.1)then !  (r2s)
c              invariant to rotation by 180 deg (& possibly by 90 deg w/ sign chang
c              as in an elliptical aperture quadrupole (& possibly symmetrical quad
c              (i.e. reflection in both x-axis and y-axis)
               print 101, A_int(2), A_int(ima+2)
  101          format(' A_int(2) = ',f9.3,3x,
     &         ' A_int(ima+2) = ',f9.3)
```

```
ccc                 avg_two = A_int(2)+A_int(ima+2)
                  do 349 i=2,ima+1
                     A_int(2*(ima+2)-i) = A_int(i)
  349             continue
                  do 350 i=3,2*ima+1
                     A_int(ima_fc+4-i) = A_int(i)
  350             continue
               else if(iver.eq.0 .and. ihor.eq.0)then !  (r2s)
c                 invariant to rotation by 180 deg (& possibly by 90 deg w/ sign chang
c                 (i.e. reflection w/ sign change in both x-axis and y-axis)
c                 Check: Is A_int(2) = - A_int(ima+2) ? (Should be)
                  print 105, A_int(2), A_int(ima+2)
  105             format(' Is A_int(2) = ',f9.3,
     &                  ' equal to A_int(ima+2) = ',f9.3, ' ?')
                  avg_two = A_int(2)+A_int(ima+2)
                  do 351 i=2,ima+1
                     A_int(2*(ima+2)-i) = avg_two - A_int(i)
  351             continue
                  do 352 i=3,2*ima+1
                     A_int(ima_fc+4-i) = avg_two - A_int(i)
  352             continue
               else if(iver.eq.0 .and. ihor.eq.1)then
c                 invariant to rotation by 180 deg w/ sign change  (r2c)
c                 as in an H-magnet or elliptical aperture sextupole
c                 (i.e. reflection in x-axis and reflection w/ sign change in y-axis)
                  print 101, A_int(2), A_int(ima+2)
                  avg_two = 2.*A_int(ima+2)
                  do 353 i=2,ima+1
                     A_int(2*(ima+2)-i) = avg_two - A_int(i)
  353             continue
                  do 354 i=3,2*ima+1
                     A_int(ima_fc+4-i) = A_int(i)
  354             continue
               else
                  print 104, idirich
  104             format(' invalid symmetry type for idirich =',i2,/,
     &                  'program abort')
                  stop
               end if
            else if(idirich.eq.2)then
               if(ihor.eq.1)then
c                 ! reflection in x-axis   (r1s)
c                 Check: A_int(2) should be equal to A_int(ima+2)
                  print 101, A_int(2), A_int(ima+2)
                  do 355 i=2,ima+1
                     A_int(ima_fc+4-i) = A_int(i)
  355             continue
               else if(ihor.eq.0)then
c                 ! reflection w/ sign change in x-axis  (r1s)
c                 Check: A_int(2) should be equal to A_int(ima+2)
                  print 105, A_int(2), A_int(ima+2)
                  avg_two = A_int(2)+A_int(ima+2)
                  do 356 i=2,ima+1
                     A_int(ima_fc+4-i) = avg_two - A_int(i)
  356             continue
               else
                  print 104, idirich
                  stop
               end if
            else
               print 106, idirich
  106          format(' invalid value for idirich: idirich =',i2,/,
     &                 'program abort')
               stop
            end if
         end if
```

```
C
C    RESET "DIRICHLET" BOUNDARY POINT POTENTIALS AND CALCULATE RESIDUALS
C=====================================================================
      SUMDA=0.
      SUMA=0.
      DO 372 J=2,JMA+1                                                !FC
      ADIR=0.
      DO 370 I=2,IMA_FC+1
      ADIR=ADIR+ZMAT(I,J)*A_INT(I) ! A_INT(I) = A(IINT(I)) IN FIRST QUADRANT
CCC      IF(J.EQ.2)PRINT *, I, A(IINT(I))
  370 CONTINUE
      SUMDA=SUMDA+(ADIR-A(JDIR(J)))**2
CCC      PRINT *, J, ADIR, A(JDIR(J)), SUMDA , A(IINT(J))
      A(JDIR(J))=A(JDIR(J))+RHODIR*(ADIR-A(JDIR(J)))
      SUMA=SUMA+A(JDIR(J))
  372 CONTINUE
      ANORM=SUMA/FLOAT(JMA)
CCC      PRINT *, J, J, SUMDA, SUMA, ANORM
C
      SUMA=0.
      DO 374 J=2,JMA+1
      SUMA=SUMA+(A(JDIR(J))-ANORM)**2
  374  CONTINUE
      RESIDD=1.E8
      IF(SUMA .NE. 0.)RESIDD=SQRT(SUMDA/SUMA)
CCC      PRINT 381, SUMDA, SUMA, RESIDD
C
C    ESTIMATE SPECTRAL NORM OF ITERATION MATRIX AND USE TO CALCULATE ===
C    OPTIMUM OVERRELAXATION FACTOR, 'RHODIR'===========================
C
      IF(SNOLDD.EQ.0.)GO TO 382                ! FIRST TIME THROUGH
      PRINT *,QWE,ICYCLED_MAX
      ROOTD=1.0/(2.0*FLOAT(ICYCLED_MAX))
      ETADIR=(SUMDA/SNOLDD+1.E-20)**ROOTD
  382 SNOLDD=SUMDA
      LAMBDAD=(RHODIR+ETADIR-1.0)/(RHODIR*SQRT(ETADIR+1.E-10))
      IF(LAMBDAD.LT.1.0) GO TO 384
      RHODIR=AMAX1(RHODIR-10.0*OMEGA,1.0)
      GO TO 386
  384 RHODIR=(2./(1.+SQRT(1.-LAMBDAD**2)))-OMEGA
  386 WRITE(6,388)ICYCLE,RESIDD,ETADIR,RHODIR,LAMBDAD
      PRINT  388,ICYCLE,RESIDD,ETADIR,RHODIR,LAMBDAD
C
C     TEMPORARY DIAGNOSTIC PRINTOUT ====================================
C
         IQMA=JMA+1                                                   !FC
         IF(IDIRICH*IMA.GT.JMA)IQMA=IDIRICH*IMA+1                     !FC
CCC         DO 376,J=2,IQMA                                           !FC
CCC         AJ=A(JDIR(J)
CCC         AI=A_INT(J)                                               !FC
CCC         PRINT 398, J, AJ, AI, RESIDD
  376    CONTINUE
C
  381 FORMAT(' SUMDA=',D9.3,2X,'SUMA=',D9.3,2X,'RESIDD=',E9.3)        !DP
CCC  388 FORMAT(1H0,I9,26X16HRHODIR OPTIMIZED,2F9.4,11H  LAMBDAD =,
CCC     X 1PE13.4)
  388 FORMAT(1H0,I9,30H RESIDD,ETADIR,RHODIR,LAMBDAD:,
     &E11.4,1X,2F9.4,1PE11.4)
  398 FORMAT(' J=',I3, '  A(JDIR(J))=',E10.4,'  A(IINT(J))=',E10.4,
     &          '  RESIDD =',E8.2)
C
      RETURN
      END
C
C*********************************************************************
C
```

```
      SUBROUTINE MATDIR                                                 !RDS
C
C     "MATDIR" CALCULATES THE MATRIX "ZMAT" USED FOR RESETTING THE
C      "DIRICHLET" BOUNDARY POINT POTENTIALS FOR THE CASE
C      OF B.C. SIMULATING REST OF UNIVERSE WITHOUT SOURCES/SINKS
C
C      LOCALLY USED VARIABLES:
       DIMENSION DEL_THETA(500), DEL_PHI(500),
     &         XDIR(500), YDIR(500), XINT(500), YINT(500),
     &         THETA(500), PHI(500)
       DOUBLE COMPLEX E_ITHETA(500), ZDIR(500),  ZINT(500),             !DP
     &         D(500,500),     ZLND(500,500),
     &         B(500,500),     BINV(500,500),
     &         ZI, P1, P2, P3 ,P4, ZMAP
       DOUBLE PRECISION ZMAT, ZM1, XRAD, YRAD, PI,  PI2INV              !DP
       DOUBLE PRECISION DEL_THETA, DEL_PHI, XDIR, YDIR, XINT, YINT,     !DP
     &                  THETA, PHI  !, RINT
C
       COMMON /PROB/CON(125)
       COMMON /RDS/ZMAT(500,500),IMA,JMA,JDIR(500),IINT(500),RESIDD,
     &   JKDUM(500), JLDUM(500), IKDUM(500), ILDUM(500), A_INT(500)
C
       EQUIVALENCE (IMAX,CON(5)), (IDIRICH,CON(14))                     !FC
C
C     READ IN DIRICHLET BOUNDARY POINTS ================================
C
       PI=4.*ATAN(1.)
C
       READ(7,*)JMA
       DO 301 J=2,JMA+2
       READ(7,*) JKDUM(J), JLDUM(J), X, Y                               !DP
       XDIR(J)=X                                                        !DP
       YDIR(J)=Y                                                        !DP
  301  CONTINUE
C
C     READ IN INTERIOR ELLIPSE (CIRCLE) POINTS =========================
C
       XRAD=0.
       YRAD=0.
       ACEPT=0.
       BCEPT=0.
       READ(7,*)IMA
       DO 305 I=2,IMA+2
       READ(7,*) IKDUM(I), ILDUM(I), X, Y                               !DP
       XINT(I) = X                                                      !DP
       YINT(I) = Y                                                      !DP
       IF(XINT(I) .EQ. 0. .AND. YINT(I) .GT. 0.) YRAD=YINT(I)
       IF(YINT(I) .EQ. 0. .AND. XINT(I) .GT. 0.) XRAD=XINT(I)
       ACEPT=AMAX1(ACEPT,X)
       BCEPT=AMAX1(BCEPT,Y)
  305  CONTINUE
       IF(XRAD .NE. 0.)THEN
       PRINT 306, XRAD !XRAD EXACT
       ELSE
       XRAD=ACEPT
       PRINT 307, XRAD !XRAD APPROXIMATE...CAUTION
       END IF
       IF(YRAD .NE. 0.)THEN
       PRINT 308, YRAD !YRAD EXACT
       ELSE
       YRAD=BCEPT
       PRINT 309, YRAD !YRAD APPROXIMATE...CAUTION
       END IF
C
C     EXTEND REGION 2'S BOUNDARY TO FORM FULL ELLIPSE (OR CIRCLE) ====!FC
C
```

```
      ima_fc=ima
      if(idirich.gt.1)then
         ima_fc=ima*idirich
         if(idirich.eq.4)then
            do 1 i=2,ima+1
               xint(2*(ima+2)-i) = - xint(i)
               yint(2*(ima+2)-i) =   yint(i)
    1       continue
            do 2 i=2,2*(ima+1)-1
               xint(ima_fc+4-i) =   xint(i)
               yint(ima_fc+4-i) = - yint(i)
    2       continue
         else if(idirich.eq.2)then
            do 3 i=2,2*(ima+1)-1
               xint(ima_fc+4-i) =   xint(i)
               yint(ima_fc+4-i) = - yint(i)
    3       continue
         else
            print 106, idirich
  106       format(' invalid value for idirich: idirich =',i2,/,
     &             'program abort')
            stop
         end if
      end if
C
C    MAP REGION 1 BOUNDARY, EXTERIOR TO ELLIPSE IN Z-PLANE TO
C    EXTERIOR OF UNIT CIRCLE IN W-PLANE ================================
C
      DO 310 J=2,JMA+2
      JDIR(J)=JLDUM(J)*IMAX+JKDUM(J)+1
      ZMAP=CMPLX(XDIR(J),YDIR(J))                ! ZMAP ARE Z-PLANE COORDINATES
      CALL ZTOW(XRAD,YRAD,ZMAP,ZDIR(J))          ! ZDIR ARE W-PLANE COORDINATES
      XDIR(J)=DREAL(ZDIR(J))
      YDIR(J)=DIMAG(ZDIR(J))
C     PRINT *, ZDIR(J), ZMAP, XDIR(J),YDIR(J) ! DIAGNOSTIC
      RDIR=SQRT(XDIR(J)**2+YDIR(J)**2)
      IF(J.EQ.2)RDIR_LAST=RDIR
C     IF(ABS((RDIR-RDIR_LAST)/RDIR).GT.0.0001)THEN
C        PRINT 320, J,XDIR(J),YDIR(J),RDIR   ! ELLIPSE RATHER THAN CIRCLE
C     END IF
      RDIR_LAST=RDIR
C
C    SET UP DIRICHLET POINT ANGLES FOR ZMAT, BELOW ======================
C
      IF(XDIR(J).NE.0.)THEN
         PHI(J)=ATAN(YDIR(J)/XDIR(J))
         IF(XDIR(J).LT.0.)PHI(J)=PHI(J) + PI
C        TYPE *,J, PHI(J)*180./PI, XDIR(J), YDIR(J)              ! DIAGNOSTIC
      ELSE IF(YDIR(J).GE.0.) THEN
         PHI(J)=PI/2.
      ELSE
         PHI(J)=3.*PI/2.
      END IF
      IF(PHI(J).LT.0.)PHI(J)=PHI(J)+2.*PI
      IF(J.NE.2) DEL_PHI(J-1)=PHI(J)-PHI(J-1)
310   CONTINUE
      IF(IDIRICH .EQ. 1) THEN ! THIS SEC. NOT USED - FOR DIAGNOSTICS ONLY
         PHI(1)=PHI(JMA+1)-2.*PI
         DEL_PHI(1)=PHI(2)-PHI(1)
         PHI(JMA+2)=PHI(2)+2.*PI
         DEL_PHI(JMA+1)=PHI(JMA+2)-PHI(JMA+1)
         ZDIR(1)=ZDIR(JMA+1)
         ZDIR(JMA+2)=ZDIR(2)
      END IF
      DO 313 J=1,JMA+2
C     PRINT 323, J, DREAL(ZDIR(J)), DIMAG(ZDIR(J)), PHI(J)*180./PI,
```

```
C     &            DEL_PHI(J)*180./PI
313    CONTINUE
C
C     MAP REGION 2 BOUNDARY, AN ELLIPSE IN Z-PLANE TO
C     THE UNIT CIRCLE IN W-PLANE ========================================
C
       DO 312 I=2,IMA_FC+2                                          !FC
       IF(I .LE. IMA+2) IINT(I)=ILDUM(I)*IMAX+IKDUM(I)+1
       ZMAP=CMPLX(XINT(I),YINT(I))              ! ZMAP ARE Z-PLANE COORDINATES
       CALL ZTOW(XRAD,YRAD,ZMAP,ZINT(I))        ! ZINT ARE W-PLANE COORDINATES
ccc       PRINT *,xrad,yrad,zmap,zint(i)
       XINT(I)=DREAL(ZINT(I))
       YINT(I)=DIMAG(ZINT(I))
       RINT=SQRT(XINT(I)**2+YINT(I)**2)
C!       E_ITHETA(I)=ZINT(I)/RINT
       IF(I.EQ.2)RINT_LAST=RINT
       PRINT *,RINT          ! , i, xint(i), yint(i)
       IF(ABS((RINT-RINT_LAST)/RINT).GT.0.0002)THEN
          PRINT 322
          STOP
       END IF
       RINT_LAST=RINT
C
C     SET UP INTERIOR POINT (UNIT CIRLCE) ANGLES FOR ZMAT, BELOW ========
C
       IF(XINT(I).NE.0.)THEN
          THETA(I)=ATAN(YINT(I)/XINT(I))
          IF(XINT(I).LT.0.)THETA(I)=THETA(I) + PI
C         TYPE *,I, THETA(I)*180./PI, XINT(I), YINT(I)            !DIAGNOSTIC
       ELSE IF(YINT(I).GE.0.) THEN
          THETA(I)=PI/2.
       ELSE
          THETA(I)=3.*PI/2.
       END IF
       IF(THETA(I).LT.0.)THETA(I)=THETA(I)+2.*PI
       IF(I.NE.2) DEL_THETA(I-1)=THETA(I)-THETA(I-1)
312    CONTINUE
       THETA(1)=THETA(IMA_FC+1)-2.*PI                               !FC
       DEL_THETA(1)=THETA(2)-THETA(1)
       THETA(IMA_FC+2)=THETA(2)+2.*PI                               !FC
       DEL_THETA(IMA_FC+1)=THETA(IMA_FC+2)-THETA(IMA_FC+1)          !FC
       ZINT(1)=ZINT(IMA_FC+1)                                       !FC
       ZINT(IMA_FC+2)=ZINT(2)                                       !FC
       DO 314 I=1,IMA_FC+2                                          !FC
C      PRINT 324, I, DREAL(ZINT(I)), DIMAG(ZINT(I)), THETA(I)*180./PI,
C     &            DEL_THETA(I)*180./PI
314     CONTINUE
C
C     CALCULATE MATRIX "ZMAT": ADIR(J) = ZMAT(I,J)*AINT(I) ==============
C
       IF(IDIRICH.GT.1)JMA=JMA+1 ! ADD 1 FOR INCOMPLETE CIRCLES     !FC
       DO 318 I=1,IMA_FC+2                                          !FC
       DO 316 J=2,JMA+1
       D(I,J)=(ZDIR(J)-ZINT(I+1))/(ZDIR(J)-ZINT(I))
       B(I,J)=ZDIR(J)/ZINT(I)
       BINV(I,J)=1./B(I,J)
       ZLND(I,J)=LOG(D(I,J))
C      IF(I.EQ.J.OR.I.EQ.10.OR.J.EQ.10)TYPE *,I,J,D(I,J),B(I,J), !DIAGNOSTIC
C     &   BINV(I,J),ZLND(I,J)
316    CONTINUE
318    CONTINUE
C
       ZI=CMPLX(0.,1.)
       DO 332 I=2,IMA_FC+1                                          !FC
       ZM1=TAN(DEL_THETA(I-1)/2.) - TAN(-DEL_THETA(I)/2.)
       P3=1.-EXP(ZI*DEL_THETA(I-1))
```

```
      P2=-DEL_THETA(I)*ZI*ZINT(I+1)
      P1=1.-EXP(-ZI*DEL_THETA(I))
      P4=+DEL_THETA(I-1)*ZI*ZINT(I-1)
C     TYPE *, I,J,P1,P2,P3,P4                                    !DIAGNOSTIC
C
      PI2INV=1./(2.*PI)
      DO 330 J=2,JMA+1
      ZMAT(I,J)=PI2INV*DREAL( ZM1+
     &(1./SIN(-DEL_THETA(I)))*
     &(P1-(P2/ZDIR(J))+(B(I+1,J)-BINV(I+1,J))*ZLND(I,J))+
     &(-1./SIN(DEL_THETA(I-1)))*
     &(P3-(P4/ZDIR(J))+(BINV(I-1,J)-B(I-1,J))*ZLND(I-1,J)) )
330   CONTINUE
332   CONTINUE
C
C    OUTPUT ZDIR,ZINT,ZMAT TO FILE DIRICH_OUT.DAT ======================
C
      WRITE(8,326)IMA, JMA, (I,I=2,9)
      IQMA = JMA+1                                                !FC
      IF(IMA_FC.GT.JMA)IQMA=IMA_FC+1                              !FC
      DO 342, J=1,IQMA
      WRITE(8,327)J, DREAL(ZDIR(J)), DIMAG(ZDIR(J)),
     &              DREAL(ZINT(J)), DIMAG(ZINT(J)),
     &              (ZMAT(I,J),I=1,9)
342   CONTINUE
C
      ILO=10
      DO 344, ITER=1,15
      IHI=ILO+12
      IF(ILO.LT.IMA_FC+1)THEN                                     !FC
         WRITE(8,328) (I,I=ILO,IHI)
         DO 343 J=1,JMA+1
         WRITE(8,329) J, (ZMAT(I,J),I=ILO,IHI)
343      CONTINUE
         ILO=IHI+1
      END IF
344   CONTINUE
C
      PRINT 325
C
  306 FORMAT(' XRAD =',F7.2,' IS EXACT')
  307 FORMAT(' XRAD =',F7.2,' IS APPROXIMATE...CAUTION')
  308 FORMAT(' YRAD =',F7.2,' IS EXACT')
  309 FORMAT(' YRAD =',F7.2,' IS APPROXIMATE...CAUTION')
  320 FORMAT(' DIR CIRCLE PTS DO NOT ALL HAVE SAME RADIUS; ',
     &       ' J=',I3,2X,'XDIR=',F7.3,2X,'YDIR=',F7.3,2X,'RDIR=',F7.3,/,
     &       ' INT CIRCLE MUST BE ELLIPSE RATHER THAN CIRCLE')
  322 FORMAT(' INT CIRCLE PTS DO NOT ALL HAVE SAME RADIUS, CHECK INPUT')
  323 FORMAT(' J=',I3,'  ZDIR(J)=',F5.2,' +i ',F5.2,'    PHI(J)=',F8.2,
     &       '    DEL_PHI(I)=',F8.2)
  324 FORMAT(' I=',I3,'  ZINT(I)=',F5.2,' +i ',F5.2,'  THETA(I)=',F8.2,
     &       '  DEL_THETA(I)=',F8.2)
  325 FORMAT(' MATDIR COMPLETED WITHOUT ERRORS,',
     &       ' ZDIR, ZINT, ZMAT WRITTEN TO DIRICH_OUT.DAT')
  326 FORMAT('PROGRAM DIRICH:',//,'IMA=',I3,1X,'JMA=',I3,//,
     &        ' J ',1X,'  R&I ZDIR  ','  R&I ZINT  ',
     &        ' ZMAT ',8(' I=',I2,1X),/)
  327 FORMAT(I3,1X,4(F5.2,1X),9(F6.2))
  328 FORMAT(/,4X,13(' I=',I3))
  329 FORMAT(I3,1X,13(F6.2))
C
      RETURN
      END
C
C*********************************************************************
C
```

```
      SUBROUTINE ZTOW(A,B,ZMAP,WBACK)                                   !RDS
C
C     "ZTOW" DOES THE CONFORMAL TRANSFORMATION FOR THE EXTERIOR
C      OF AN ELLIPSE TO THE EXTERIOR OF A UNIT CIRCLE
C
      DOUBLE PRECISION  A, B                                            !DP
      DOUBLE COMPLEX ZMAP,WBACK                                         !DP
C
      IF(DREAL(ZMAP).GT.0..OR.DREAL(ZMAP).EQ.0..AND.DIMAG(ZMAP).GT.0.)
     &THEN
      WBACK=(ZMAP + (ZMAP*ZMAP + B*B -A*A)**0.5 )/(A+B) !-PI/2<ANGLE<=PI/2
      ELSE
      WBACK=(ZMAP - (ZMAP*ZMAP + B*B -A*A)**0.5 )/(A+B)
      END IF
      RETURN
      END
C
C*********************************************************************
```

Appendix B: Example and Test Series Check of Accuracy of the Newly Incorporated POISSON Boundary Condition Option

10 Appendix B: Example and Test Series Check of Accuracy of the Newly Incorporated POISSON Boundary Condition Option

To test the POISSON program modification allowing for singularity-free universe outside problem boundaries the following example solves analytically (Program DIRAN) and with POISSON the case of two current filaments ($+I$ at z_{q_1} and $-I$ at z_{q_2}) in air. The analytical solution to this problem is $F = -\frac{\mu_0 I}{2\pi} ln(\frac{z-z_{q_1}}{z-z_{q_2}})$.

- Make an AUTOMESH file (e.g. *fcirc.dat*; see Figure 3 above) with a circular boundary as region 1. Region 2 must be a circular boundary just inside (i.e. a distance $\sim$ DX) region 1's boundary. To assure accuracy of the numerically discretized equation, DX must be such that angular spacing of nodes on region 2's boundary is $\sim$ 10 degrees or less.
- Run AUTOMESH
- Run [.mods]LATTICE with *14 1 *58 .001 *108 5 *22 0 *32 -1 *45 15 *20 42 *49 2 S to get: (1) mesh coordinates (x, y)=f(k, l) in *outlat.lis*, (see Figure 16) and (2) k, l, x, y lists for Dirichlet circle points and inner circle points in *dirich_in.dat* suitable for input to programs POISSON, DIRICH, & (with the addition of one line) DIRAN. Notes: (a) from *tape73.dat* we know that LMAX=15 and there are 42 Dirichlet boundary points on the outer circle and (b) the lower boundary condition must be set for the Dirichlet condition.
- Run TEKPLOT with M, 0 1 0 0 S, S, GO to view mesh (See Figure 4, above).
- Make (in [.mods]) a *diran_in.dat* file (see Figure 17) from [ross.pois]*dirich_in.dat* (created when LATTICE is run with IDIRICH=CON(14)=1) and 2 current filament strengths with suitable locations obtained from grid points in *outlat.lis*.
- Run [.mods]diran to (1) verify that Matrix is accurate by comparing analytical solution $\vec{A}d_c$ with solution $\vec{A}_{jdir} = [Matrix] * \vec{A}_{iint}$, where $\vec{A}_{iint}$ is analytical, and (2) get a listing of fixed Dirichlet potentials for POISSON. (See *diran_out.dat* in Figure 18.)
- Run [.mods]POISSON with *14 0 and fixed potential inputs from *diran_out.dat* and 2 current filaments from *diran_in.dat* or *outlat.lis* in *pfcirc.dat* (see Figure 19) to compare POISSON's relaxed inner circle vector potential values in *outpoi.lis* (see Figure 20) with those of analytical values, A_{iint_i}, in *diran_out.dat*.
- Run TEKPLOT with M, 1 0 29 S, S, GO to view B-field lines (identical to Fig. 5).
- Run [.mods]LATTICE with *14 1 *58 .001 *108 5 *22 0 *32 -1 *45 15 *49 2 S
- Run [.mods]POISSON with Dirichlet circle point potentials free floating and 2 current filaments from *diran_in.dat* or *outlat.lis* in *pfcirc.dat* (see Fig. 21) to compare (1) POISSON's converged solution on Dirichlet circle and on inner circle in *dirich_out.dat* (see Fig. 22) with those of analytical values, A_{iint_i} and Ad_c_j in *diran_out.dat* and (2) POISSON's converged solution everywhere in *outpoi.lis* (see Fig. 23) with those from the previous POISSON run with Dirichlet point potentials fixed.

- Run TEKPLOT with M, 1 0 29 S, S, GO to view B-field lines (identical to Fig. 5).

Figure 18 verifies that the matrix relating Dirichlet circle points and interior circle points is correct.

A comparison of Figures 20 and 18 verifies additionally that POISSON relaxes to the correct solution for a singularity-free universe outside problem boundaries when the vector potential of boundary points are fixed at the appropriate values (i.e. if we somehow know beforehand the correct vector potentials on the boundary).

A comparison of Figures 22 (or 23) and 18 verifies additionally that POISSON converges to the correct solution for a singularity-free universe outside problem boundaries when the vector potential of boundary points are allowed to float.

THE FOLLOWING IS A MAP OF ARRAYS X,Y , CYCLE. 0

```
L
.
.     X .....
...
.
```

		5	6	7	8	9	10	11
15	X	-0.338	-0.171	0.000	0.188	0.371	0.000	0.000
	Y	1.151	1.188	1.200	1.185	1.141	0.000	0.000
14	X	-0.498	-0.195	0.000	0.174	0.342	0.545	0.705
	Y	1.092	0.981	1.000	0.985	0.940	1.069	0.971
13	X	-0.383	-0.136	0.070	0.270	0.500	0.643	0.849
	Y	0.924	0.854	0.837	0.829	0.866	0.766	0.849
12	X	-0.481	-0.264	-0.046	0.162	0.370	0.567	0.766
	Y	0.705	0.702	0.683	0.672	0.671	0.657	0.643
11	X	-0.377	-0.161	0.050	0.258	0.461	0.662	0.866
	Y	0.528	0.520	0.510	0.505	0.499	0.494	0.500
10	X	-0.485	-0.271	-0.059	0.149	0.355	0.557	0.755
	Y	0.357	0.350	0.344	0.339	0.335	0.333	0.334
9	X	-0.377	-0.166	0.043	0.250	0.454	0.655	0.849
	Y	0.177	0.174	0.171	0.169	0.167	0.167	0.170
8	X	-0.481	-0.272	-0.062	0.146	0.351	0.555	0.761
	Y	0.000	0.000	0.000	0.000	0.000	0.000	0.000
7	X	-0.377	-0.166	0.043	0.250	0.454	0.655	0.849
	Y	-0.177	-0.174	-0.171	-0.169	-0.167	-0.167	-0.170
6	X	-0.485	-0.271	-0.059	0.149	0.355	0.557	0.755
	Y	-0.357	-0.350	-0.344	-0.339	-0.335	-0.333	-0.334
5	X	-0.377	-0.161	0.050	0.258	0.461	0.662	0.866
	Y	-0.528	-0.520	-0.510	-0.505	-0.499	-0.494	-0.500

Figure 16. Partial *outlat.lis* file listing

```
-0.271 .350 1. -0.271 -0.350 -1.
      42
13    8     1.20000    0.00000
13    9     1.18520    0.18770
13   10     1.14130    0.37080
12   11     1.06920    0.54480
12   12     0.97080    0.70530
11   13     0.84850    0.84850
11   14     0.70530    0.97080
10   14     0.54480    1.06920
 9   15     0.37080    1.14130
 8   15     0.18770    1.18520
 7   15     0.00000    1.20000
 6   15    -0.17080    1.18780
 5   15    -0.33810    1.15140
 5   14    -0.49850    1.09160
 4   14    -0.64880    1.00950
 3   13    -0.78580    0.90690
 2   13    -0.90690    0.78580
 2   12    -1.00950    0.64880
 1   11    -1.09160    0.49850
 1   10    -1.15140    0.33810
 1    9    -1.18780    0.17080
 1    8    -1.20000    0.00000
 1    7    -1.18780   -0.17080
 2    6    -1.15140   -0.33810
 1    5    -1.09160   -0.49850
 2    4    -1.00950   -0.64880
 2    3    -0.90690   -0.78580
 3    3    -0.78580   -0.90690
 4    2    -0.64880   -1.00950
 5    2    -0.49850   -1.09160
 5    1    -0.33810   -1.15140
 6    1    -0.17080   -1.18780
 7    1     0.00000   -1.20000
 8    1     0.18770   -1.18520
 9    1     0.37080   -1.14130
10    2     0.54480   -1.06920
11    2     0.70530   -0.97080
11    3     0.84850   -0.84850
12    4     0.97080   -0.70530
12    5     1.06920   -0.54480
13    6     1.14130   -0.37080
13    7     1.18520   -0.18770
13    8     1.20000    0.00000
```

```
      34
12    8     1.00000    0.00000
12    9     0.98480    0.17360
12   10     0.93970    0.34200
11   11     0.86600    0.50000
11   12     0.76600    0.64280
10   13     0.64280    0.76600
 9   13     0.50000    0.86600
 9   14     0.34200    0.93970
 8   14     0.17360    0.98480
 7   14     0.00000    1.00000
 6   14    -0.19510    0.98080
 5   13    -0.38270    0.92390
 4   13    -0.55560    0.83150
 4   12    -0.70710    0.70710
 3   11    -0.83150    0.55560
 3   10    -0.92390    0.38270
 2    9    -0.98080    0.19510
 2    8    -1.00000    0.00000
 2    7    -0.98080   -0.19510
 3    6    -0.92390   -0.38270
 3    5    -0.83150   -0.55560
 4    4    -0.70710   -0.70710
 4    3    -0.55560   -0.83150
 5    3    -0.38270   -0.92390
 6    2    -0.19510   -0.98080
 7    2     0.00000   -1.00000
 8    2     0.17360   -0.98480
 9    2     0.34200   -0.93970
 9    3     0.50000   -0.86600
10    3     0.64280   -0.76600
11    4     0.76600   -0.64280
11    5     0.86600   -0.50000
12    6     0.93970   -0.34200
12    7     0.98480   -0.17360
12    8     1.00000    0.00000
```

Figure 17. File *diran_in.dat*

```
37   0.01  0.01  0.02  0.03  0.04  0.09  0.21  0.26  0.13  0.06  0.03  0.02  0.01
38   0.01  0.01  0.01  0.02  0.03  0.05  0.10  0.23  0.25  0.12  0.05  0.03  0.02
39   0.01  0.01  0.01  0.01  0.02  0.03  0.05  0.11  0.24  0.24  0.11  0.05  0.03
40   0.00  0.01  0.01  0.01  0.01  0.02  0.03  0.05  0.12  0.25  0.23  0.10  0.05
41   0.00  0.01  0.01  0.01  0.01  0.01  0.02  0.03  0.06  0.13  0.26  0.21  0.09
42   0.00  0.00  0.01  0.01  0.01  0.01  0.01  0.02  0.03  0.06  0.14  0.27  0.20
43   0.00  0.00  0.00  0.01  0.01  0.01  0.01  0.01  0.02  0.03  0.07  0.16  0.27

ab(1)=-.2348E-03  ab(2)=0.3006E+00  ab(3)=0.2348E-03  ab(4)=-.2476E+00
ab(5)=-.2853E+00  ab(6)=0.2672E+00  ab(7)=0.9389E-09  ab_tot=0.3487E-01

 analtyical check for dirichlet matrix, [Z]

 input: z1=-.2710E+00 +i 0.3500E+00 cur1= 1.00
        z2=-.2710E+00 +i -.3500E+00 cur2=-1.00

output: sumda=0.3055E+00  suma=0.3055E+00  anorm=0.3237E-07  residd=0.1000E+01

 j    R&I ZDIR    R&I ZINT   a(iint(j))  a(jdir(j))   ad_c(j)

  2  1.20  0.00  1.00  0.00  0.0000D+00  0.3150D-07  0.0000E+00
  3  1.19  0.19  0.98  0.17  0.1407D-01  0.1158D-01  0.1155E-01
  4  1.14  0.37  0.94  0.34  0.2826D-01  0.2319D-01  0.2313E-01
  5  1.07  0.54  0.87  0.50  0.4267D-01  0.3487D-01  0.3477E-01
  6  0.97  0.71  0.77  0.64  0.5738D-01  0.4662D-01  0.4649E-01
  7  0.85  0.85  0.64  0.77  0.7245D-01  0.5843D-01  0.5828E-01
  8  0.71  0.97  0.50  0.87  0.8791D-01  0.7025D-01  0.7008E-01
  9  0.54  1.07  0.34  0.94  0.1036D+00  0.8196D-01  0.8178E-01
 10  0.37  1.14  0.17  0.98  0.1193D+00  0.9335D-01  0.9319E-01
 11  0.19  1.19  0.00  1.00  0.1341D+00  0.1041D+00  0.1039E+00
 12  0.00  1.20 -0.20  0.98  0.1482D+00  0.1136D+00  0.1135E+00
 13 -0.17  1.19 -0.38  0.92  0.1565D+00  0.1205D+00  0.1205E+00
 14 -0.34  1.15 -0.56  0.83  0.1552D+00  0.1249D+00  0.1251E+00
 15 -0.50  1.09 -0.71  0.71  0.1415D+00  0.1262D+00  0.1264E+00
 16 -0.65  1.01 -0.83  0.56  0.1158D+00  0.1234D+00  0.1237E+00
 17 -0.79  0.91 -0.92  0.38  0.8126D-01  0.1162D+00  0.1165E+00
 18 -0.91  0.79 -0.98  0.20  0.4171D-01  0.1044D+00  0.1048E+00
 19 -1.01  0.65 -1.00  0.00  0.0000D+00  0.8858D-01  0.8884E-01
 20 -1.09  0.50 -0.98 -0.20 -.4171D-01  0.6931D-01  0.6949E-01
 21 -1.15  0.34 -0.92 -0.38 -.8126D-01  0.4756D-01  0.4766E-01
 22 -1.19  0.17 -0.83 -0.56 -.1158D+00  0.2417D-01  0.2422E-01
 23 -1.20  0.00 -0.71 -0.71 -.1415D+00  0.3465D-07  0.0000E+00
 24 -1.19 -0.17 -0.56 -0.83 -.1552D+00 -.2417D-01 -.2422E-01
 25 -1.15 -0.34 -0.38 -0.92 -.1565D+00 -.4756D-01 -.4766E-01
 26 -1.09 -0.50 -0.20 -0.98 -.1482D+00 -.6931D-01 -.6949E-01
 27 -1.01 -0.65  0.00 -1.00 -.1341D+00 -.8858D-01 -.8884E-01
 28 -0.91 -0.79  0.17 -0.98 -.1193D+00 -.1044D+00 -.1048E+00
 29 -0.79 -0.91  0.34 -0.94 -.1036D+00 -.1162D+00 -.1165E+00
 30 -0.65 -1.01  0.50 -0.87 -.8791D-01 -.1234D+00 -.1237E+00
 31 -0.50 -1.09  0.64 -0.77 -.7245D-01 -.1262D+00 -.1264E+00
 32 -0.34 -1.15  0.77 -0.64 -.5738D-01 -.1249D+00 -.1251E+00
 33 -0.17 -1.19  0.87 -0.50 -.4267D-01 -.1205D+00 -.1205E+00
 34  0.00 -1.20  0.94 -0.34 -.2826D-01 -.1136D+00 -.1135E+00
 35  0.19 -1.19  0.98 -0.17 -.1407D-01 -.1041D+00 -.1039E+00
 36  0.37 -1.14  1.00  0.00  0.0000D+00 -.9335D-01 -.9319E-01
 37  0.54 -1.07  0.00  0.00  0.0000D+00 -.8196D-01 -.8178E-01
 38  0.71 -0.97  0.00  0.00  0.0000D+00 -.7025D-01 -.7008E-01
 39  0.85 -0.85  0.00  0.00  0.0000D+00 -.5843D-01 -.5828E-01
 40  0.97 -0.71  0.00  0.00  0.0000D+00 -.4662D-01 -.4649E-01
 41  1.07 -0.54  0.00  0.00  0.0000D+00 -.3487D-01 -.3477E-01
 42  1.14 -0.37  0.00  0.00  0.0000D+00 -.2319D-01 -.2313E-01
 43  1.19 -0.19  0.00  0.00  0.0000D+00 -.1158D-01 -.1155E-01
```

Figure 18. Partial file *diran_out.dat*

```
0 Dump
*14 0   S
 13     8   0.00000E+00
 13     9   0.11547E-01
 13    10   0.23128E-01
 12    11   0.34774E-01
 12    12   0.46491E-01
 11    13   0.58277E-01
 11    14   0.70080E-01
 10    14   0.81782E-01
  9    15   0.93186E-01
  8    15   0.10394E+00
  7    15   0.11349E+00
  6    15   0.12047E+00
  5    15   0.12506E+00
  5    14   0.12641E+00
  4    14   0.12374E+00
  3    13   0.11655E+00
  2    13   0.10477E+00
  2    12   0.88838E-01
  1    11   0.69492E-01
  1    10   0.47659E-01
  1     9   0.24218E-01
  1     8   0.00000E+00
  1     7   -.24218E-01
  2     6   -.47659E-01
  1     5   -.69492E-01
  2     4   -.88838E-01
  2     3   -.10477E+00
  3     3   -.11655E+00
  4     2   -.12374E+00
  5     2   -.12641E+00
  5     1   -.12506E+00
  6     1   -.12047E+00
  7     1   -.11349E+00
  8     1   -.10394E+00
  9     1   -.93186E-01
 10     2   -.81782E-01
 11     2   -.70080E-01
 11     3   -.58277E-01
 12     4   -.46491E-01
 12     5   -.34774E-01
 13     6   -.23128E-01
 13     7   -.11547E-01
  6 10   1.
  6  6  -1.
 -1 S
```

Figure 19. File *pfcirc.dat* (boundary points with fixed potential)

```
THE FOLLOWING IS A MAP OF THE POTENTIAL ARRAY A, CYCLE  160

  L
  .        .
  .    K .......
 ...        .
  .

      1         2         3         4         5         6         7         8         9        10        11        12        13
 15  0.00E+00  0.00E+00  0.00E+00  0.00E+00  1.25E-01  1.20E-01  1.13E-01  1.04E-01  9.32E-02  0.00E+00  0.00E+00  0.00E+00  0.00E+00
 14  0.00E+00  0.00E+00  0.00E+00  1.24E-01  1.26E-01  1.48E-01  1.34E-01  1.19E-01  1.04E-01  8.18E-02  7.01E-02  0.00E+00  0.00E+00
 13  0.00E+00  1.05E-01  1.17E-01  1.56E-01  1.56E-01  1.69E-01  1.47E-01  1.17E-01  8.76E-02  7.28E-02  5.83E-02  0.00E+00  0.00E+00
 12  0.00E+00  8.88E-02  1.12E-01  1.42E-01  1.93E-01  2.17E-01  1.95E-01  1.44E-01  1.04E-01  7.70E-02  5.75E-02  4.65E-02  0.00E+00
 11  6.95E-02  8.91E-02  1.16E-01  1.84E-01  2.92E-01  2.96E-01  1.87E-01  1.20E-01  8.11E-02  5.74E-02  4.26E-02  3.48E-02  0.00E+00
 10  4.77E-02  5.91E-02  8.11E-02  1.32E-01  2.47E-01  6.07E-01  2.46E-01  1.30E-01  7.86E-02  5.18E-02  3.68E-02  2.82E-02  2.31E-02
  9  2.42E-02  4.10E-02  6.32E-02  1.08E-01  1.97E-01  1.93E-01  1.03E-01  5.73E-02  3.49E-02  2.33E-02  1.68E-02  1.41E-02  1.15E-02
  8  0.00E+00 -1.14E-04 -8.95E-05 -3.31E-05 -6.34E-06  2.23E-06  4.01E-06  3.67E-06  2.82E-06  1.99E-06  1.30E-06  5.32E-07  0.00E+00
  7 -2.42E-02 -4.14E-02 -6.33E-02 -1.08E-01 -1.97E-01 -1.93E-01 -1.03E-01 -5.73E-02 -3.49E-02 -2.33E-02 -1.68E-02 -1.41E-02 -1.15E-02
  6  0.00E+00 -4.77E-02 -8.07E-02 -1.32E-01 -2.47E-01 -6.07E-01 -2.46E-01 -1.30E-01 -7.86E-02 -5.18E-02 -3.68E-02 -2.82E-02 -2.31E-02
  5 -6.95E-02 -8.61E-02 -1.16E-01 -1.84E-01 -2.92E-01 -2.96E-01 -1.87E-01 -1.20E-01 -8.10E-02 -5.74E-02 -4.26E-02 -3.48E-02  0.00E+00
  4  0.00E+00 -8.88E-02 -1.12E-01 -1.42E-01 -1.93E-01 -2.17E-01 -1.95E-01 -1.44E-01 -1.04E-01 -7.70E-02 -5.75E-02 -4.65E-02  0.00E+00
  3  0.00E+00 -1.05E-01 -1.17E-01 -1.56E-01 -1.56E-01 -1.69E-01 -1.47E-01 -1.17E-01 -8.76E-02 -7.28E-02 -5.83E-02  0.00E+00  0.00E+00
  2  0.00E+00  0.00E+00  0.00E+00 -1.24E-01 -1.26E-01 -1.48E-01 -1.34E-01 -1.19E-01 -1.04E-01 -8.18E-02 -7.01E-02  0.00E+00  0.00E+00
  1  0.00E+00  0.00E+00  0.00E+00  0.00E+00 -1.25E-01 -1.20E-01 -1.13E-01 -1.04E-01 -9.32E-02  0.00E+00  0.00E+00  0.00E+00  0.00E+00
```

Figure 20. Partial file *outpoi.lis* (boundary points with fixed potential)

```
0 Dump
S
 6 10  1.
 6  6 -1.
-1 S
```

Figure 21. File *pfcirc.dat*

37	0.01	0.01	0.02	0.03	0.04	0.09	0.21	0.26	0.13	0.06	0.03	0.02	0.01
38	0.01	0.01	0.01	0.02	0.03	0.05	0.10	0.23	0.25	0.12	0.05	0.03	0.02
39	0.01	0.01	0.01	0.01	0.02	0.03	0.05	0.11	0.24	0.24	0.11	0.05	0.03
40	0.00	0.01	0.01	0.01	0.01	0.02	0.03	0.05	0.12	0.25	0.23	0.10	0.05
41	0.00	0.01	0.01	0.01	0.01	0.01	0.02	0.03	0.06	0.13	0.26	0.21	0.09
42	0.00	0.00	0.01	0.01	0.01	0.01	0.01	0.02	0.03	0.06	0.14	0.27	0.20
43	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.01	0.02	0.03	0.07	0.16	0.27

J	JLDOM	JKDOM	A_DIRICS	IKDOM	ILDOM	A_INT
2	13	8	0.2136D-03	12	8	0.2167D-03
3	13	9	0.1182D-01	12	9	0.1433D-01
4	13	10	0.2349D-01	12	10	0.2853D-01
5	12	11	0.3522D-01	11	11	0.4303D-01
6	12	12	0.4700D-01	11	12	0.5790D-01
7	11	13	0.5887D-01	10	13	0.7324D-01
8	11	14	0.7067D-01	9	13	0.8801D-01
9	10	14	0.8233D-01	9	14	0.1045D+00
10	9	15	0.9372D-01	8	14	0.1195D+00
11	8	15	0.1043D+00	7	14	0.1340D+00
12	7	15	0.1136D+00	6	14	0.1480D+00
13	6	15	0.1203D+00	5	13	0.1553D+00
14	5	15	0.1247D+00	4	13	0.1555D+00
15	5	14	0.1258D+00	4	12	0.1415D+00
16	4	14	0.1233D+00	3	11	0.1157D+00
17	3	13	0.1162D+00	3	10	0.8100D-01
18	2	13	0.1044D+00	2	9	0.4097D-01
19	2	12	0.8851D-01	2	8	0.5949D-04
20	1	11	0.6921D-01	2	7	-.4104D-01
21	1	10	0.4738D-01	3	6	-.8014D-01
22	1	9	0.2405D-01	3	5	-.1151D+00
23	1	8	0.1209D-03	4	4	-.1409D+00
24	1	7	-.2379D-01	4	3	-.1550D+00
25	2	6	-.4697D-01	5	3	-.1548D+00
26	1	5	-.6865D-01	6	2	-.1475D+00
27	2	4	-.8795D-01	7	2	-.1335D+00
28	2	3	-.1039D+00	8	2	-.1191D+00
29	3	3	-.1157D+00	9	2	-.1040D+00
30	4	2	-.1228D+00	9	3	-.8758D-01
31	5	2	-.1254D+00	10	3	-.7280D-01
32	5	1	-.1242D+00	11	4	-.5747D-01
33	6	1	-.1199D+00	11	5	-.4261D-01
34	7	1	-.1131D+00	12	6	-.2810D-01
35	8	1	-.1038D+00	12	7	-.1390D-01
36	9	1	-.9325D-01	12	8	0.2167D-03
37	10	2	-.8188D-01	0	0	-.1752D+21
38	11	2	-.7023D-01	0	0	-.1752D+21
39	11	3	-.5844D-01	0	0	-.1752D+21
40	12	4	-.4656D-01	0	0	-.1752D+21
41	12	5	-.3475D-01	0	0	-.1752D+21
42	13	6	-.2304D-01	0	0	-.1752D+21
43	13	7	-.1139D-01	0	0	-.1752D+21

Figure 22. Partial file *dirich_out.dat*

(boundary points with floating potential)

```
THE FOLLOWING IS A MAP OF THE POTENTIAL ARRAY A. CYCLE  550
  L
  .
  .    K .......
 ...
  .
```

	1	2	3	4	5	6	7	8
15	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.25E-01	1.20E-01	1.14E-01	1.04E-
14	0.00E+00	0.00E+00	0.00E+00	1.23E-01	1.26E-01	1.48E-01	1.34E-01	1.20E-
13	0.00E+00	1.04E-01	1.16E-01	1.55E-01	1.55E-01	1.69E-01	1.47E-01	1.17E-
12	0.00E+00	8.85E-02	1.12E-01	1.41E-01	1.93E-01	2.17E-01	1.95E-01	1.44E-
11	6.92E-02	8.89E-02	1.16E-01	1.84E-01	2.92E-01	2.96E-01	1.87E-01	1.20E-
10	4.74E-02	5.89E-02	8.10E-02	1.32E-01	2.47E-01	6.07E-01	2.46E-01	1.30E-
9	2.41E-02	4.10E-02	6.32E-02	1.08E-01	1.97E-01	1.93E-01	1.03E-01	5.75E-
8	1.18E-04	5.69E-05	1.00E-04	1.72E-04	2.08E-04	2.19E-04	2.22E-04	2.21E-
7	-2.38E-02	-4.10E-02	-6.29E-02	-1.08E-01	-1.97E-01	-1.93E-01	-1.03E-01	-5.70E-
6	0.00E+00	-4.70E-02	-8.01E-02	-1.32E-01	-2.47E-01	-6.07E-01	-2.46E-01	-1.30E-
5	-6.87E-02	-8.54E-02	-1.15E-01	-1.83E-01	-2.91E-01	-2.95E-01	-1.87E-01	-1.20E-
4	0.00E+00	-8.80E-02	-1.11E-01	-1.41E-01	-1.92E-01	-2.17E-01	-1.94E-01	-1.44E-
3	0.00E+00	-1.04E-01	-1.16E-01	-1.55E-01	-1.55E-01	-1.69E-01	-1.46E-01	-1.17E-
2	0.00E+00	0.00E+00	0.00E+00	-1.23E-01	-1.25E-01	-1.48E-01	-1.34E-01	-1.19E-
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	-1.24E-01	-1.20E-01	-1.13E-01	-1.04E-

	6	7	8	9	10	11	12	13
15	1.20E-01	1.14E-01	1.04E-01	9.37E-02	0.00E+00	0.00E+00	0.00E+00	0.00E+00
14	1.48E-01	1.34E-01	1.20E-01	1.04E-01	8.23E-02	7.07E-02	0.00E+00	0.00E+00
13	1.69E-01	1.47E-01	1.17E-01	8.80E-02	7.32E-02	5.89E-02	0.00E+00	0.00E+00
12	2.17E-01	1.95E-01	1.44E-01	1.04E-01	7.73E-02	5.79E-02	4.70E-02	0.00E+00
11	2.96E-01	1.87E-01	1.20E-01	8.13E-02	5.78E-02	4.30E-02	3.52E-02	0.00E+00
10	6.07E-01	2.46E-01	1.30E-01	7.88E-02	5.21E-02	3.71E-02	2.85E-02	2.35E-02
9	1.93E-01	1.03E-01	5.75E-02	3.52E-02	2.36E-02	1.71E-02	1.43E-02	1.18E-02
8	2.19E-04	2.22E-04	2.21E-04	2.19E-04	2.16E-04	2.14E-04	2.14E-04	2.11E-04
7	-1.93E-01	-1.03E-01	-5.70E-02	-3.47E-02	-2.31E-02	-1.67E-02	-1.39E-02	-1.14E-02
6	-6.07E-01	-2.46E-01	-1.30E-01	-7.84E-02	-5.16E-02	-3.67E-02	-2.81E-02	-2.30E-02
5	-2.95E-01	-1.87E-01	-1.20E-01	-8.09E-02	-5.74E-02	-4.26E-02	-3.48E-02	0.00E+00
4	-2.17E-01	-1.94E-01	-1.44E-01	-1.04E-01	-7.69E-02	-5.75E-02	-4.66E-02	0.00E+00
3	-1.69E-01	-1.46E-01	-1.17E-01	-8.76E-02	-7.28E-02	-5.84E-02	0.00E+00	0.00E+00
2	-1.48E-01	-1.34E-01	-1.19E-01	-1.04E-01	-8.19E-02	-7.02E-02	0.00E+00	0.00E+00
1	-1.20E-01	-1.13E-01	-1.04E-01	-9.32E-02	0.00E+00	0.00E+00	0.00E+00	0.00E+00

Figure 23. Partial file *outpoi.lis* (boundary points with floating potential)

Appendix C: Proof of Field Equivalence in Figures 15a and 15b

11 Appendix C: Proof of Field Equivalence in Figures 15a and 15b

The complex potential $F(z) \equiv A + iV$ for the current filament distribution shown in Figure 14a is given by Eq. (19). We have[3] when the scalar potential $V = 0$:

$$\Im\left\{\ln[\frac{(z-x_a)(z-x_b)}{z}] - \ln[\frac{(z+x_a)(z+x_b)}{z}]\right\} = 0 \tag{20}$$

Now defining the scalar $\rho \equiv \sqrt{x_a x_b}$, we have

$$\frac{(z \pm x_a)(z \pm x_b)}{z} = z + \frac{x_a x_b}{z} \pm (x_a + x_b) = \rho\left(\frac{z}{\rho} + \frac{\rho}{z} \pm \frac{x_a + x_b}{\rho}\right) \tag{21}$$

which, along the circle defined by $z = \rho e^{i\phi}$, reduces to $2\rho\cos\phi \pm (x_a + x_b)$. The log of this expression, being pure real, implies that along the circle $x^2 + y^2 = x_a x_b$ we have $V = 0$.

Thus for the current filament distribution in Fig. 14a (or 15a), the circle defined by $r = \sqrt{x_a x_b}$ is a constant scalar potential surface; outside this radius the field is identical to that of Fig. 14b (or 15b) where an infinitely permeable iron disk of radius $r = \sqrt{x_a x_b}$ likewise maintains the scalar potential at this radius constant.

[3] Alternatively, for $V = 0$, $\Im\{\ln[(z-x_a)(z-x_b)] - \ln[(z+x_a)(z+x_b)]\} \equiv \Im\left\{\ln[r_p e^{i\theta_p}] - \ln[r_q e^{i\theta_q}]\right\} = 0$, where $r_p = \sqrt{[(x-x_a)(x-x_b) - y^2]^2 + [(2x - x_a - x_b)y]^2}$, $r_q = \sqrt{[(x+x_a)(x+x_b) - y^2]^2 + [(2x + x_a + x_b)y]^2}$, $\theta_p = \arctan\frac{(2x-x_a-x_b)y}{(x-x_a)(x-x_b)-y^2}$, and $\theta_q = \arctan\frac{(2x+x_a+x_b)y}{(x+x_a)(x+x_b)-y^2}$. Solving, we have for $V = 0$: $\theta_p = \theta_q$ from which $y = 0$ or $x^2 + y^2 = x_a x_b$, as above.

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
TECHNICAL INFORMATION DEPARTMENT
BERKELEY, CALIFORNIA 94720