27
8378
7567715

UCID-17725 Rev. 1

Lawrence Livermore Laboratory

MAGIC: A ONE-DIMENSIONAL MAGNETO-INDUCTIVE PARTICLE CODE

T. A. Brengle
B. I. Cohen

MASTER

July 18, 1978

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

MAGIC: A ONE-DIMENSIONAL MAGNETO-INDUCTIVE
PARTICLE CODE

ABSTRACT

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or 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.

MAGIC, a new one-dimensional particle code, simulates magneto-inductive phenomena in a cylindrically-symmetric magnetized plasma. We describe the physical model and the computational algorithm used for the code. A user's guide to and a listing of MAGIC are also included.

INTRODUCTION

A one-dimensional particle code, MAGIC, simulates magneto-inductive phenomena in a cylindrically-symmetric magnetized plasma. We present the physical model and the computational algorithm and contrast them with those of LMR, a one-dimensional version of SUPERLAYER. The results of the two codes are very close, despite their differences in physics and algorithms.

The MAGIC I code solves for the vector potential from the accumulated currents; MAGIC II solves for the magnetic flux. With MAGIC, the user can have either a uniform-density (warm or cold) background plasma and can use one of several injection modes (pulsed, constant, or linearly increasing rates).

A user's guide (see Appendix A) to MAGIC I and II is also included. It briefly provides the information needed to execute MAGIC. We have also included the Fortran listing of the code (see Appendix B).

DESCRIPTION OF THE PHYSICAL MODEL

MAGIC is one-dimensional, magneto-inductive particle code. It simulates an infinitely long, cylindrically-symmetric magnetized plasma. The physical model describes magneto-inductive phenomena and is a radial version of the r-z codes used by Dickman, Morse, and Nielson,¹ and Byers.² In our model, MAGIC I solves the following vector potential equation from the accumulated currents:

$$\left(\frac{1}{r} \frac{\partial}{\partial r} r \frac{\partial}{\partial r} - \frac{1}{r^2} \right) A_{\theta} = -4\pi J_{\theta}/c. \quad (1)$$

MAGIC II solves this equivalent equation for the magnetic flux rA_{θ} , and

$$B_z = \frac{1}{r} \frac{\partial}{\partial r} (r A_\theta), \quad E_\theta = -\partial A_\theta / \partial t. \quad (2)$$

Radiation and electrostatic fields are legislated out of the model because the longitudinal and transverse displacement currents are absent on the r. h. s. of Eq. (1). The model of the magnetic field is self-consistent and the code can move two ion species in the self-consistent fields. We assume implicitly that electrons completely neutralize the charge of the ions, but do not contribute to the currents. We assume the plasma is surrounded by a cylindrical conductor.

Maxwell's equations and the equations of motion are finite-differenced and integrated in the following manner:

$$\begin{aligned} \left(\frac{1}{r} \frac{\partial}{\partial r} r \frac{\partial}{\partial r} - \frac{1}{r^2} \right) A_\theta^n &= \frac{A_{\theta j+1}^n - 2A_{\theta j}^n + A_{\theta j-1}^n}{\Delta r^2} \\ &+ \frac{1}{2r_j \Delta r} \left(A_{\theta j+1}^n - A_{\theta j-1}^n \right) - \frac{A_{\theta j}^n}{r_j^2} = 4\pi c^{-1} j_{\theta j} \\ &= -4\pi c^{-1} \sum_i \left\{ \frac{e}{m r_i} S(r_i^n - r_j) \left[p_{\theta i} - \frac{e}{c} \sum_j r_i S(r_i^n - r_j) A_{\theta j}^n \right] \right\}, \end{aligned} \quad (3)$$

$$B_{zj}^n = \frac{A_{\theta j+1}^n - A_{\theta j-1}^n}{2\Delta r} + \frac{A_{\theta j}^n}{r_j}, \quad \frac{1}{2} (E_{\theta j}^n + E_{\theta j}^{n-1}) = \frac{-(A_\theta^n - A_\theta^{n-1})}{c \Delta t}, \quad (4)$$

where $p_{\theta i}$ $\equiv$ canonical angular momentum, S $\equiv$ linear interpolation factor, i $\equiv$ super-particle index, j, j' $\equiv$ grid indexes, and n $\equiv$ time level. E_θ and A_θ vanish at $r = 0$, and at $r = a$ on the surface of a perfect conductor. A_θ has a smooth first derivative at $r = 0$, and $A_\theta = 0$ for $r = a$.

The super particles, advanced with a Boris mover,³ can be represented formally by:

$$\begin{aligned}
(v_r, v_\theta)^{n+1/2} = (v_r, v_\theta)_i^{n-1/2} + \Delta t (e/m) \left\{ \sum_j s(r_i^n - r_j) E_{\theta j}^n \hat{e}_\theta \right. \\
\left. + \left(\frac{1}{2c} \right) \left[(v_r, v_\theta)^{n+1/2} + (v_r, v_\theta)^{n-1/2} \right] \sum_j s(r_i^n - r_j) B_{zj} \hat{e}_z \right\}
\end{aligned} \quad (5)$$

and

$$r_i^{n+1} = \left[\left(r_i^n + v_r^{n+1/2} \Delta t \right)^2 + \left(v_\theta^{n+1/2} \Delta t \right)^2 \right]^{1/2}_i. \quad (6)$$

We chose a system of nondimensional units internal to the code:

$$r^* = r/c\Delta t, \quad v_{r,\theta}^* = v_{r,\theta}/c, \quad \Delta t^* = \omega_{ci}^0 \Delta t,$$

$$B_z^* = (B_z/B_0)\Delta t^*, \quad E_\theta^* = (E_\theta/B_0)\Delta t^*, \quad \text{and} \quad A_0^* = (\omega_{ci}^0/c) (A_0/B_0),$$

where ω_{ci}^0 is the ion cyclotron frequency of the principal ion species in the applied magnetic field B_0 , and Δt is the time step. These are the input quantities. The input and output are in physical units (Gaussian cgs).

The uniform-density background plasma can either be a warm Maxwellian or cold. There are three injection modes: pulsed, constant, and linearly increasing rates. Slow particles ($v_r \lesssim a/\Delta t$) are elastically reflected by the conductor. Fast particles ($v_r \gtrsim a/\Delta t$) are inelastically reflected at a random angle with the thermal velocity of the wall (which is specified by the user).

The computational scheme is explicit after taking the A_0 dependence of Eq. (3) to the l.h.s. It is also perfectly time-centered and yields the following linear dispersion relation^{4,5} describing the propagation of compressional Alfvén waves across an applied magnetic field in a cold plasma:

$$\lim_{k\Delta r \rightarrow 0} \tan^2(\omega\Delta t/2) = \frac{k^2 v_a^2 \Delta t^2/4}{1 + (\omega_{ci}\Delta t/2)^2 + k^2 v_a^2/\omega_{ci}^2}, \quad (7)$$

where k is the effective radial wave number $k = \chi_n^{-1}$, $J_1(\chi_n) = 0$, and $v_A = (\omega_{ci}/\omega_{pi}) c$ the Alfvén wave velocity. Thus as $k v_a/\omega_{ci} \rightarrow \infty$, $\omega \rightarrow \omega_{ci}$, if $\omega_{ci}\Delta t/2 \ll 1$. There is no Courant condition, and the magnetosonic waves are stable at all wave numbers. Numerical stability of a cyclotron

orbit in a warm plasma requires $\omega_{ci} \Delta t / 2 < 1$. Figure 1 displays results of simulations that verify the linear dispersion relation given by Eq. (7).

The MAGIC code is reversible because of the perfect time-centering, and there is no numerical damping. Pulsed injection necessarily excites normal modes at all frequencies, and in particular at the highest (Nyquist) frequency of the simulation, $\omega = \pi / \Delta t$. The lack of numerical damping permits these high frequency oscillations to persist, i.e., the system rings! Byers' code SUPERLAYER uses the pulsed injection mode, but numerically damps high frequency oscillations.⁶ We recommend against using the pulsed-injection mode in MAGIC, since the ringing is physical but the finite time-step distortion is not.

Figures 2-5 display simulation results obtained using MAGIC and LMR, a one-dimensional version of SUPERLAYER. LMR conserves canonical angular momentum exactly. These four simulations use injection of energetic deuterium to drive the magnetic field to a null inside an injected ring. MAGIC injected one super particle per time step for 1024 time steps. Using a constant rate of injection: 800 A at an energy of 20 keV and a radius of 5.8 cm. LMR injected a nearly equivalent amount of current, but with the canonical angular momentum of each newly injected ion not self-consistent with the plasma self-magnetic field. The results are obviously quite similar despite the differences in algorithms and physics.

Fig. 1. Linear dispersion relation for compressional Alfvén waves.

Fig. 2. Simulation results from a ring-like injection for MAGIC II (a) and LMR (b). The ion Larmor radius ρ_i in the applied magnetic field is noted for reference.

Fig. 3. Simulation results from a ring-like injection for MAGIC II (a) and LMR (b). Note the reversal of the magnetic field inside the ring.

Fig. 4. Simulation results from a ring-like injection for MAGIC II (a) and LMR (b).

Fig. 5. Simulation results at $r \sim 1/3 \rho_i$ from a ring-like injection for MAGIC II (a) and LMR (b).

REFERENCES

1. D. O. Dickman, R. L. Morse, and C. W. Nielson, *Phys. Fluids* 12 1708 (1969).
2. J. A. Byers, *Phys. Rev. Lett.* 39 1476 (1977).
3. J. P. Boris, "Relativistic Plasma Simulation — Optimization of a Hybrid Code," in *Proc. of the Fourth Conf. on Numerical Simulation of Plasmas*, J. P. Boris and R. Shanny, Eds. (U.S. Government Printing Office, Washington, D.C., 1970) p. 3.
4. B. I. Cohen, "A New Particle Pusher and a Time-Centered Field Solve for SUPERLAYER," MFC/TC/78-110 (March 14, 1978).
5. J. A. Byers, B. I. Cohen, W. C. Condit, and J. D. Hanson, *Hybrid Simulations of Quasineutral Phenomena in Magnetized Plasma*, Lawrence Livermore Laboratory, Rept., UCRL-79437 (1977).
6. R. P. Freis and B. I. Cohen, Lawrence Livermore Laboratory, private communication (May 1978).

LJH

APPENDIX A: USER'S GUIDE TO MAGIC I AND II

MAGIC I and II are one-dimensional, magneto-inductive particle simulation codes written in FORTRAN, and designed to run on the DEC 10 computer. There is one major difference between the two codes. MAGIC I has a field solver that solves for the vector potential from the accumulated currents; MAGIC II solves for the magnetic flux from the currents. MAGIC II also has some additional convenience features, particularly in the plotting of output data.

MAGIC allows either uniform-density or injection runs, or combination of the two. The user can also launch cold magnetosonic waves, for which he supplies the wave number and amplitude.

This Guide briefly provides the user with the information needed to execute MAGIC. It describes the input file and output plot formats and gives an example of the input file. This Appendix, in somewhat altered form, is also available online. See the section on Source File Availability for details.

Input File Format

MAGIC reads all of the necessary input data (in Gaussian cgs) from a NAMELIST input file. An example of an input file that tests the constant rate injection in MAGIC II looks like (where the parameters are defined in the next section):

THIS IS A RUN FOR BETA OF ONE.

\$INPUT

CURR=8E+2

EINJ=20.

ITRMAX=1024

ITRPLT=64, INJMOD=1

NGYRO=1

GYRRAN=0.

RINJ=5.84

NINJ=1

WINJ=.1

DELTS=.1

The first dollar sign must appear in column 2. Data items can be separated by commas. Comments can appear before the \$INPUT line. The second dollar sign (in column 1) is a required delimiter.

Input Parameters. The NAMELIST parameters that can be set in the data file are:

AWAVE	Amplitude for launching a magnetosonic wave in volts.
BO	Background magnetic field in gauss.
CURR	Injection current to be reached at the end of the injection in amperes.
DELTS	Ion cyclotron frequency times the time step.
DENSI	Initial uniform density in particles per cubic centimetre.
EINJ	Energy of the injected particles in hilelectron volts.
GYRRAN	Range of gyrophase angles, from -GYRRAN to +GYRRAN, in radians.
IATHS	Logical switch for the self-vector potential versus the radial position plot.
ISZS	Logical switch for the self-magnetic field versus the radial position plot.
IDENS	Logical switch for the particle density versus the radial position plot.
IETHS	Logical switch for the self-magnetic field versus the radial position plot.
IJTH	Logical switch for the theta current versus the radial position plot.
INJMOD	Injection mode: 0. pulsed injection ($T = 0$). 1. constant-rate injection. 2. linearly-increasing-rate injection.
IPT1	Grid cell numbers for sampling self-vector potential versus time.
IPT2	
IPT3	
IPT4	
ITEK	Switch to turn on plotting for the Tektronix terminal.
ITRMAX	Number of time steps to the end of the run.
ITRPLT	Number of time steps between the phase space and field quarters plots.
IVRR	Logical switch for the radial velocity versus the radial position plot.

LVTHVR Logical switch for the theta velocity versus the radial velocity plot.
 KWAVE Wave number for launching a magnetosonic wave.
 NGYRO Number of gyrophase angles per injection point.
 NINJ Number of particles to be injected per time step per gyrophase angle.
 NUS Drag coefficient, v_{ie}/ω_{ci}^0 .
 RINJ Radius of the center of the injection profile in centimetres.
 RMAX Outer radius of system in centimetres.
 TEMPO Energy of the uniform background particles in electron volts (Maxwellian-loaded).
 TWALL Energy given to the fast particles that are absorbed by the wall and then re-emitted.
 WINJ Width of the injection profile in centimeters.

Running the Code

To run the code, type:

```
RU MAGIC2 [30,3057]
```

Follow this line with a carriage return. Then enter the name of the input file when the code requests it, and again follow this line with a carriage return. When the run is completed, the DD80 output file, called MAGIC2.DD8, can be processed by GRAF10 and then plotted on the Versatec printer. The file can also be looked at with TEK10. GRAF10 and TEK10 are available from the magnetic Fusion Energy Network's Computer Center.

The user should not set the ITEK parameter to 1 when his terminal is not a Tektronix; if he does so, he receives a vast amount of useless, printed binary output.

Output Plot Format

Output plots, in the form of DD80 files, are packed four to a page. The output consists of:

- A heading page. This page contains:
 - (a) The date and time of the run.
 - (b) The name of the input data file.
 - (c) A complete printout of the contents of the data file.

- (d) A list of the run parameters calculated by the code.
- (e) A list of the current values, including defaulted ones, of all the NAMELIST parameters.
- Phase space and field quantity plots, taken at preset intervals. The plots included in this group are:
 - (a) Radial velocity versus radial position.
 - (b) Theta velocity versus radial velocity.
 - (c) Particle density versus radial position.
 - (d) Self-vector potential versus radial position.
 - (e) Self-magnetic field ($\Delta B/B_0$) versus radial position.
 - (f) Self-electric field versus radial position.
 - (g) Theta current versus radial position.
- History plots. The plots included in this group are:
 - (a) Self-vector potential versus time. This is plotted for four different radial locations.
 - (b) Injected and calculated energy versus time.
 - (c) Energy error (fluctuation) versus time.
 - (d) Maximum beta versus time.
 - (e) Average beta versus time.

MAGIC is designed so that any of the phase space-field quantity plots can be switched off by a NAMELIST parameter. (See the parameter descriptions in the preceding section that begin, "Logical switch....") MAGIC then repacks the plots four to a page.

Source File Availability

These seven MAGIC files are available publicly on the M Division DEC 10 from user number [30,3057] through the PIP program. The files are:

MAGIC1.STR and MAGIC2.STR	(contain the common blocks),
MAGIC1.FOR and MAGIC2.FOR	(contain the main routine),
MAGIC1.LIB and MAGIC2. LIB	(contain the subroutines),
MAGIC2.SWP	(online copy of the user's guide).

APPENDIX B: MAGIC LISTING

```

C      THIS IS THE STORAGE BLOCK FOR MAGIC2
C      FOR DESCRIPTION OF VARIABLES. SEE MAGDOC IN MAGIC2.LIB

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      PARAMETER NP=1000,NG=64,NT=200
      PARAMETER NGROUP=2

      DOUBLE PRECISION JI,M,KWAVE,NUMER,NUS
      DOUBLE PRECISION JTHETA

      COMMON /INDATA/ BO,DELT,RMAX,DENS1,NUS,CURR,ITRMAX,ITRPLT
1      ,NINJ,RINJ,WINJ,*NUMOD,EINJ,NGYRO,GYRRAN
1      ,TEMPO,TWALL,ITF

      COMMON /RODATA/ ITER,NIN,NLAST
1      ,VINJS,RINJS,WINJS
1      ,ENRTOT,NHOT,NCOLD,VTHERM,VWALL

      COMMON /PRTCLS/ R(NP),VR(NP),PTHETA(NP),VTHETA(NP)

      COMMON /GRID/ FLUX(NG),JTHETA(NG),BZ(NG),ETHETA(NG)
1      ,CH(NG),CZ(NG),CP(NG),TV1(NG),TV2(NG)

      COMMON /CONST/ PI,C,DELTS,DELR,DELR5,E,M,WCI

      COMMON /SWITCH/ IVRR,IVTHR,IVTHVR,IDENS,IBZS,IFLXS,IETHS
1      ,IJTH,IATHS

      COMMON /PLOTS/ JPLOT,XOFF,YOFF,FNAME,KSAMPL,ISAMPL,KMAX

      COMMON /HISTORY/ XTIME(NT),BAVG(NT),BMAX(NT),TENERC(NT)
1      ,TENERI(NT),ATTIME(4,NT)
1      ,IPT1,IPT2,IPT3,IPT4

      COMMON /GROUP/ NMIN(NGROUP),NMAX(NGROUP),BETA(NGROUP)
1      ,SUPN(NGROUP)

      COMMON /WAVES/ WPI,WALF,VALF,KWAVE,AWAVE,WWAVE,AMP

```



```

C      MAGIC2 :
C          TIME CENTERED
C          DOUBLE PRECISION
C          MAGNETO - INDUCTIVE
C          IONS - ONLY
C          CODE

C      ALGORITHMS BY BRUCE COHEN
C      CODING BY TOM BRENGLE

C      CURRENT VERSION - 19-APR-78

C      INCLUDE 'MAGIC2.STR'

C      SET UP PLOT FILE
C      CALL KEEP80 (5HMAGIC2,2)
C      CALL ODB01D (6HMAGIC2,1)

C      GET INPUT DATA FROM INPUT FILE
C      CALL INPUT

C      SET UP CONSTANTS AND CALCULATE RUN PARAMETERS
C      CALL SETCON

C      PRINT OUT THE HEADING PAGE
C      CALL HEADER

C      IF ITEK IS ONE, TURN ON TEKTRONICS SCREEN
C      IF (ITEK .EQ. 1) CALL TEKID

C      ITER = 0

C      ZERO OUT GRID ARRAYS, CALCULATE ENERGY DUE TO BACKGROUND
C      MAGNETIC FIELD, AND IF A WAVE IS TO BE LAUNCHED,
C      SET ETHETA FOR T=0.
C      CALL INITLZ

C      FOR COLD PARTICLES (GROUP 1) SET UP VR, VTHETA, PTHETA AND R
C      (USING MAXWELLIAN IF TEMPO IS NOT ZERO). ALSO ACCUMULATE
C      JTHETA AND ENERGY FOR T=0. IF A WAVE IS TO BE LAUNCHED,
C      SET UP VR FOR T=0.
C      CALL MAXLOD

C      BEGIN TIME LOOP
101  CONTINUE

C      LOOP THROUGH UNTIL TIME TO PLOT
C      DO 100 I = 1, ITRPLT

C      INJECT NEW PARTICLES AND ACCUMULATE JTHETA AND ENERGY FOR THEM
C      CALL INJECT

C      SOLVE FOR NEW SELF FIELDS
C      CALL FIELDS

C      MOVE NEW PARTICLES BACKWARD BY DELT/2 AND THEN MOVE
C      ALL PARTICLES AHEAD BY DELT
C      CALL BORIS MOVER

```

```

CALL BORMOV

C      IF APPROPRIATE, SAMPLE FOR TIME PLOTS
      IF (ITER .GE. ISAMPL * (KSAMPL - 1)) CALL SPLOT

      ITER = ITER + 1
100    CONTINUE

C      DIAGNOSTICS (FIELD QUANTITY AND PHASE SPACE PLOTS)
C      PLOT VR VS. R
      IF (IVRR .EQ. 1) CALL VRR
C      PLOT VTHETA VS. VR
      IF (IVTHVR .EQ. 1) CALL VTHVR
C      PLOT DENSITY VS. R
      IF (IDENS .EQ. 1) CALL DENSR
C      PLOT ATHETA VS. R
      IF (IATHS .EQ. 1) CALL ATHSR
C      PLOT BZSELF VS. R
      IF (IBZS .EQ. 1) CALL BZSR
C      PLOT ETHETA VS. R
      IF (IETHS .EQ. 1) CALL ETHSR
C      PLOT JTHETA VS. R
      IF (IJTH .EQ. 1) CALL JTHR

C      PUT NEXT SET OF PLOTS TO START ON NEW PAGE
      JPLOT = 5
      CALL IPLOT
      JPLOT = 1

C      IF NOT FINISHED, REPEAT TIME LOOP
      IF (ITER .LT. ITRMAX) GO TO 101

C      IF FINISHED, TAKE LAST TIME SAMPLE
      CALL SPLOT

C      TIME DIAGNOSTICS (PLOTS VS. TIME)

C      PLOT ATHETA VS. TIME AT IPT?
      CALL ATHT(1)
      CALL ATHT(2)
      CALL ATHT(3)
      CALL ATHT(4)
C      PLOT ENERGY INJECTED AND ENERGY CALCULATED VS. TIME
      CALL EICT
C      PLOT ENERGY ERROR (FLUCTUATION) VS. TIME
      CALL EFLT
C      PLOT MAXIMUM BETA VS. TIME
      CALL BMAXT
C      PLOT AVERAGE BETA VS. TIME
      CALL BAVGT

C      FINISH CURRENT PAGE
      JPLOT = 5
      CALL IPLOT

C      ALL DONE
      CALL EXIT

END

```

```

SUBROUTINE ATHSR
INCLUDE 'MAGIC2.STR'
C .....

C      THIS PLOTS SELF ATHETA VERSUS R

      CALL IPLOT

      CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
      CALL SETLCH (0.,.07,1,0,0,1)
      WRITE (100,1300)
1300   FORMAT ('SELF VECTOR POTENTIAL : THETA')
      CALL SETLCH (.217,-.055,1,0,0,0)
      WRITE (100,1301)
1301   FORMAT ('RADIAL POSITION (CM)')

      TV1(1) = 0.
      TV2(1) = 0.

      DO 1302 I = 2,NG
      TV1(I) = (I - 1) * DELR
      TV2(I) = FLUX(I) / ((I - 1) * DELRS)
1302   CONTINUE

      TS1 = 0.
      TS2 = 0.

      CALL CARTMM (NG,TS1,TS2,TV2,2)
      YMIN = 1.3*AMINI(SNGL(TS1),0.)
      YMAX = 1.3*AMAXI(0.,SNGL(TS2))
      CALL MAPS (0.,RMAX,(80*C/WC1)*YMIN,(80*C/WC1)*YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
      CALL MAP (0.,RMAX,YMIN,YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
      CALL TRACE (TV1,TV2,NG,2,2)

C .....

      RETURN
      END

SUBROUTINE ATHT(J)
INCLUDE 'MAGIC2.STR'
C .....

C      THIS PLOTS ATHETA AS A FUNCTION OF TIME
C      FOR SEVERAL DIFFERENT POINTS

      CALL IPLOT

      CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
      CALL SETLCH (0.,.09,1,0,0,1)
      WRITE (100,200) J
200   FORMAT ('THETA VS. TIME AT IPT',11)
      CALL SETLCH (.217,-.055,1,0,0,0)
      WRITE (100,250)
250   FORMAT ('TIME')

      TS1 = 0.

```

```

TS2 = 0.

CALL CARTMM (KMAX,TS1,TS2,ATTIME(J,1),8)
YMIN = AMIN1(1.2 * SNGL(TS1) , SNGL(TS1) / 1.2)
YMAX = AMAX1(1.3 * SNGL(TS2) , SNGL(TS2) / 1.3)
IF (YMIN .NE. 0. .OR. YMAX .NE. 0.) GO TO 300
YMIN = -1.E-4
YMAX = 1.E-4
300 CALL MAPS (XTIME(1),XTIME(KMAX),(80*C/WC1)*YMIN,(80*C/WC1)*YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL MAP (XTIME(1),XTIME(KMAX),YMIN,YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL TRACE (XTIME,ATTIME(J,1),KMAX,2,8)

C.....

RETURN
END

SUBROUTINE BAVGT
INCLUDE 'MAGIC2.STR'

C.....

C      THIS PLOTS THE AVERAGE BETA

CALL IPLOT

CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
CALL SETLCH (0.,.09,1,0,0,1)
WRITE (100,100)
100  FORMAT ('AVERAGE BETA')
CALL SETLCH (.217,-.055,1,0,0,0)
WRITE (100,200)
200  FORMAT ('TIME')

CALL CARTMM (KMAX,TS1,TS2,BAVG,2)
1  CALL MAPS (XTIME(1),XTIME(KMAX),TS1/1.2,1.3*TS2,
      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL TRACE (XTIME,BAVG,KMAX,2,2)

C.....

RETURN
END

SUBROUTINE BMAXT
INCLUDE 'MAGIC2.STR'

C.....

C      THIS PLOTS THE MAXIMUM VALUE OF BETA

CALL IPLOT

CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
CALL SETLCH (0.,.09,1,0,0,1)
WRITE (100,100)
100  FORMAT ('MAXIMUM BETA')
CALL SETLCH (.217,-.055,1,0,0,0)
WRITE (100,200)

```

```

200      FORMAT ('TIME')

      CALL CARTMM (KMAX,TS1,TS2,BMAX,2)
      TS1=TS1/1.2
      TS2=1.3*TS2
      CALL MAPS (XTIME(1),XTIME(KMAX),TS1,TS2,
1          XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
      CALL TRACE (XTIME,BMAX,KMAX,2,2)

C.....

      RETURN
      END

      SUBROUTINE BORMOV
      INCLUDE 'MAGIC2.STR'
C.....

C      THIS IS THE 2D BORIS MOVER IN POLAR COORDINATES

C      FIRST ROTATE AND ACCELERATE NEWLY INJECTED SUPERPARTICLES
C      BACKWARD BY DELT/2.

      DELRS1 = 1. / DELRS

      IF (NLAST.EQ. NMAX(2)) GO TO 350
C      NOTE THAT ON FIRST TIMESTEP (ITER=0) , NLAST=0
C      SINCE NMAX(2) IS ALWAYS THE INDEX OF THE LAST SUPERPARTICLE,
C      EVEN IF THERE ARE NO HOT SUPERPARTICLES, ALL SUPERPARTICLES IN
C      THE SYSTEM AT THE FIRST TIMESTEP ARE MOVED THE HALF STEP.

      DO 300 I = NLAST+1,NMAX(2)
      RPGC = R(I) * DELRS1
      J = RPGC
      DR = RPGC - J
      DRC = 1. - DR
      BZP = DRC*BZ(J+1)+DR*BZ(J+2)+DELTS
      ETHP = DRC*ETHETA(J+1)+DR*ETHETA(J+2)
C      ALPHA=-B(CODE,Z)/4
      A = .25*BZP
      A2 = A*A
      T = 1.+A2*(.333333+A2*.133333)
      F2 = -.25*T
C      TAN(ALPHA)=
      T = T*A

      VT2 = VTHETA(I)+F2*ETHP

      VR2 = VR(I)-VT2*T
      S = (T+T)/(1.+T*T)
      VT2 = VT2+VR2*S
      VR(I) = VR2-VT2*T

      VTHETA(I) = VT2+F2*ETHP

300      CONTINUE

C      NOW MOVE ALL SUPERPARTICLES IN SYSTEM AHEAD BY DELT.
C      THIS TIME ACCUMULATE C'S, JTHETA, AND ENERGY.

```

```

C      THE C'S ARE THE LINEARLY WEIGHTED COEFFS OF FLUX
C      WHICH ARISE FROM THE IMPLICIT CALCULATION OF JTHETA
350    DO 390 IGROUP = 1,NGROUP
      IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 390

      DO 400 I = NMIN(IGROUP),NMAX(IGROUP)
        RPGC = R(I) * DELRSI
        J = RPGC
        DR = RPGC - J
        DRC = 1. - DR
        BZP = DRC*BZ(J+1)+DR*BZ(J+2)+DELTS
        ETHP = DRC*ETHETA(J+1)+DR*ETHETA(J+2)
C      ALPHA=-B(CODE,Z)/2
        A = -.5*BZP
        A2 = A*A
        T = 1.-A2*(.333333+A2*.133333)
        F2 = .5*T
C      TAN(ALPHA)=
        T = T*A

        VT2 = VTHETA(I)+F2*ETHP - VTHETA(I) * NUS * DELTS
        VR2 = VR(I) - VR(I) * NUS * DELTS

        VR2 = VR2-VT2*T
        S = (1+T)/(1+T+1)
        VT2 = VT2+VR2*S
        VR2 = VR2-VT2*T

        VT3 = VT2+F2*ETHP - VT2 * NUS * DELTS
        VR2 = VR2 - VR2 * NUS * DELTS

        R2 = R(I)+VR2
        R(I) = DSQRT(R2**2+VT3**2)
        PTHETA(I) = PTHETA(I) - 2. * R(I) * VTHETA(I) * NUS

        RPGC = R(I) * DELRSI
        J = RPGC
        DR = RPGC - J
        DRC = 1. - DR

      IF (R(I)) 4071,4072,4071
4071    RI = 1. / R(I)
        RISQ = RI ** 2
        CA = R2*RI
        SA = VT3*RI

        CM(J+2) = CM(J+2) - BETA(IGROUP) * DRC * DR * RISQ
        CZ(J+1) = CZ(J+1) - BETA(IGROUP) * DR * DR * RISQ
        CZ(J+2) = CZ(J+2) - BETA(IGROUP) * DRC * DRC * RISQ
        CP(J+1) = CM(J+2)

      GO TO 407
4072    CA = 1.
        SA = 0.

407    VR(I) = VR2*CA+VT3*SA
        VTHETA(I) = -VR2*SA+VT3*CA

        RMAXS = FLOAT(NG - 1) * DELRS
        IF (R(I) .LT. RMAXS) GO TO 408

```

```

C      SLOW SUPERPARTICLES HITTING WALL BOUNCE BACK
      R(I) = 2. * RMAXS - R(I)
      VR(I) = - VR(I)
      IF (R(I) .GT. 0.) GO TO 408
C      FAST SUPERPARTICLES ARE ABSORBED, RE-EMITTED AT VWALL,
C      RANDOM ANGLE.
      R(I) = RMAXS
      EKIN = SUPN(IGROUP) * (M / 2.) * (C ** 2)
1      * (VTHETA(I) ** 2 + VR(I) ** 2)
      ENRTOT = ENRTOT - EKIN
      THETA = PI * (.5 - RAN(-1))
      VR(I) = - VWALL * COS(THETA)
      VTHETA(I) = VWALL * SIN(THETA)
      PTHETA(I) = R(I) * (VTHETA(I) / DELTS + R(I) / 2.)
      EKIN = SUPN(IGROUP) * (M / 2.) * (C ** 2)
1      * (VTHETA(I) ** 2 + VR(I) ** 2)
      ENRTOT = ENRTOT + EKIN

408    JTHETA(J+1) = JTHETA(J+1) + BETA(IGROUP) * DRC *
      ( -.5 + PTHETA(I) / (R(I) ** 2) )
1      JTHETA(J+2) = JTHETA(J+2) + BETA(IGROUP) * DR *
      ( -.5 + PTHETA(I) / (R(I) ** 2) )

400    CONTINUE
390    CONTINUE

C.....

      RETURN
      END

      SUBROUTINE BZSR
      INCLUDE 'MAGIC2 STR'
C.....

C      THIS PLOTS SELF BZ VERSUS R
      CALL IPLOT

      CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
      CALL SETLCH (0.,.09,1,0,0,1)
      WRITE (100,1300)
1300    FORMAT ('DELTA B / B0 : Z')
      CALL SETLCH (.217,-.055,1,0,0,0)
      WRITE (100,1301)
1301    FORMAT ('RADIAL POSITION (CM)')

      CONST = RMAX / (NG - 1)

      DO 1302 I = 1,NG
      TV(I) = (I - 1) * CONST
1302    CONTINUE

      TS1 = 0.
      TS2 = 0.

```

```

CALL CARTMM (NG,TS1,TS2,BZ,2)
YMIN = 1.3*AMIN1(SNGL(TS1),0.)
YMAX = 1.3*AMAX1(0.,SNGL(TS2))
CALL MAPS (0.,RMAX,YMIN/DELTS,YMAX/DELTS,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL MAP (0.,RMAX,YMIN,YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL TRACE (TV1,BZ,NG,2,2)

```

C.....

```

RETURN
END

```

```

SUBROUTINE DENSX
INCLUDE 'MAGIC2.STR'

```

C.....

C THIS PLOTS DENSITY VERSUS R

```

CALL IPLOT

CALL MAP (0...5,0...3,XOFF,XOFF+.5,YOFF,YOFF+.3)
CALL SETLCH (0...1,1,0,0,1)
WRITE (100,502)
502 FORMAT ('DENSITY OF PARTICLES')
CALL SETLCH (.217,-.055,1,0,0,0)
WRITE (100,503)
503 FORMAT ('RADIAL POSITION (CM)')

CONST = RMAX / (NG - 1)

DO 500 I = 1,NG
TV1(I) = 0.
TV2(I) = (I - 1) * CONST
500 CONTINUE

DO 504 IGROUP = 1,NGROUP
IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 504

DO 501 I = NMIN(IGROUP),NMAX(IGROUP)
RPGC = R(I)/DELRS
J = INT(RPGC)
TV1(J+1) = TV1(J+1) + SUPN(IGROUP) /
1      (PI * ((J + 1) ** 2 - J ** 2)
1      * DELR ** 2)
501 CONTINUE
504 CONTINUE

TV1(NG) = 0.

CALL CARTMM (NG,YMIN,YMAX,TV1,2)
CALL MAPS (0.,RMAX,0.,1.3*YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)

R1 = 0.

```



```

DO 505 I = 1,NG-1
R2 = 1 * DELR
CALL LINE(R1,TV1(I),R2,TV1(I))
CALL LINE(R2,TV1(I),R2,TV1(I+1))
R1 = R2
505 CONTINUE

C.....

RETURN
END

SUBROUTINE EFLT
INCLUDE 'MAGIC2.STR'
C.....

C THIS PLOTS THE ENERGY CONSERVATION ERROR
CALL IPLOT

CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
CALL SETLCH (0.,.09,1,0,0,1)
WRITE (100,100)
100 FORMAT ('ENERGY ERROR')
CALL SETLCH (1,217,-.055,1,0,0,0)
WRITE (100,200)
200 FORMAT ('TIME')

DO 300 I = 1,KMAX
TENERI(I) = TENERI(I) - TENERC(I)
300 CONTINUE

CALL CARTMM (KMAX,TS1,TS2,TENERI,2)
YMIN = AMINI(1.2*SNGL(TS1),SNGL(TS1)/1.2)
YMAX = AMAXI(1.3*SNGL(TS2),SNGL(TS2)/1.3)
IF (YMIN .NE. 0. .OR. YMAX .NE. 0.) GO TO 302
YMIN = -1.E-4
YMAX = 1.E-4
302 1 CALL MAPS (XTIME(I),XTIME(KMAX),YMIN,YMAX,
XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL TRACE (XTIME,TENERI,KMAX,2,2)

C.....

RETURN
END

SUBROUTINE EICT
INCLUDE 'MAGIC2.STR'
C.....

C THIS PLOT INJECTED ENERGY AND CALCULATED ENERGY
C AS A FUNCTION OF TIME

CALL IPLOT

CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
CALL SETLCH (0.,.09,1,0,0,1)
WRITE (100,400)
400 FORMAT ('ENERGY')

```

```

CALL SETLCH (.217,-.055,1.0,0.0)
WRITE (100,401)
401  FORMAT ('TIME')

CALL CARTMM (KMAX,TS1,TS2,TENER1,2)
CALL CARTMM (KMAX,TS3,TS4,TENERC,2)
TS5 = AMIN1(SNGL(TS1),SNGL(TS3))
TS6 = AMAX1(SNGL(TS2),SNGL(TS4))
YMIN = .9999 * TS5 - .1 * (TS6 - TS5)
YMAX = 1.0001 * TS6 + .1 * (TS6 - TS5)
IF (YMIN .NE. YMAX) GO TO 402
YMIN = .99 * YMIN
YMAX = 1.01 * YMAX
402  CALL MAPS (XTIME(1),XTIME(KMAX),YMIN,YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL SETPCH (1,1,1.0,100)
CALL TRACEC (1HC,XTIME,TENERC,KMAX,2,2)
CALL TRACEC (1H1,XTIME,TENER1,KMAX,2,2)

C.....

RETURN
END

SUBROUTINE ETHSR
INCLUDE 'MAGIC2.STR'
C.....

C      THIS PLOTS SELF ETHETA VERSUS R

CALL IPLOT

CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
CALL SETLCH (0.,.076,1.0,0.1)
WRITE (100,1300)
1300  FORMAT ('SELF ELECTRIC FIELD : THETA')
CALL SETLCH (.217,-.055,1.0,0.0)
WRITE (100,1301)
1301  FORMAT ('RADIAL POSITION (CM)')

CONST = RMAX / (NG - 1)

DO 1302 I = 1,NG
TV1(I) = (I - 1) * CONST
1302  CONTINUE

TS1 = 0.
TS2 = 0.

CALL CARTMM (NG,TS1,TS2,ETHETA,2)
YMIN = 1.3*AMIN1(SNGL(TS1),0.)
YMAX = 1.3*AMAX1(0.,SNGL(TS2))
CALL MAPS (0.,RMAX,(80/DELTS)*YMIN,(80/DELTS)*YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL MAP (0.,RMAX,YMIN,YMAX,
1      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL TRACE (TV1,ETHETA,NG,2,2)

C.....

```

```

RETURN
END
SUBROUTINE FIELDS
  INCLUDE 'MAGIC2.STR'
  DIMENSION ALPHAS(NG), BETAS(NG), GAMMAS(NG), DELTAS(NG)
  DIMENSION ES(NG), FS(NG)
  EQUIVALENCE (CM, GAMMAS), (CZ, BETAS)
  EQUIVALENCE (CP, ALPHAS, ES), (TV1, DELTAS, FS)
C .....
C      THIS IS THE 1D FIELD SOLVER FOR THE RADIAL DIRECTION
      IF (KWAVE .NE. 0. .AND. ITER .EQ. 0) GO TO 517

      JTHETA(1) = 0.
      JTHETA(NG) = 0.

C      DO TRIDIAGONAL SOLVE FOR FLUX AS FUNCTION OF R FROM THE
C      JTHETA AND C'S THAT HAVE BEEN ACCUMULATED
      DO 504 I = 2, NG-1
        RG = FLOAT (I - 1) * DELRS
        RSQRD = RG ** 2
        ALPHAS(I) = - (RG / DELRS ** 2 - 1. / (2. * DELRS)
                     + RSQRD * CP(I))
        BETAS(I) = (- 2. * RG / DELRS ** 2 + RSQRD * CZ(I))
        GAMMAS(I) = - (RG / DELRS ** 2 + 1. / (2. * DELRS)
                     + RSQRD * CM(I))
        DELTAS(I) = - RSQRD * DELTS * JTHETA(I)
504      CONTINUE

        ALPHAS(1) = 0.
        BETAS(1) = 1.
        GAMMAS(1) = 0.
        DELTAS(1) = 0.
        ALPHAS(NG) = 0.
        BETAS(NG) = 1.
        GAMMAS(NG) = 0.
        DELTAS(NG) = 0.

        ES(1) = - ALPHAS(1) / BETAS(1)
        FS(1) = DELTAS(1) / BETAS(1)

        DO 506 I = 2, NG
          DENOM1 = 1. / (BETAS(I) + GAMMAS(I) * ES(I-1))
          ES(I) = - ALPHAS(I) * DENOM1
          FS(I) = (DELTAS(I) + GAMMAS(I) * FS(I-1)) * DENOM1
506        CONTINUE

        DO 507 J = 1, NG-1
          I = NG - J
          FS(I) = FS(I) - ES(I) * FS(I+1)
507        CONTINUE

C      FS NOW CONTAINS THE NEW FLUX
C      USE IT TO GET THE NEW BZ
      DO 508 I = 2, NG-1
        BZ(I) = (FS(I+1) - FS(I-1)) / (2. * (I - 1) * DELRS ** 2)
508      CONTINUE

```

```

      BZ(1) = 2. * FS(2) / (DELRS ** 2)
      BZ(NG) = - FS(NG-1) / ((NG - 1) * DELRS ** 2)

      ETHETA(1) = 0.

C      NOW USE FS AND OLD FLUX TO GET A TIME CENTERED ETHETA
      DO 509 I = 2,NG
      ETHETA(I) = - 2. * (FS(I) - FLUX(I)) / ((I - 1) * DELRS)
        - ETHETA(I)
509    CONTINUE

C      SAVE NEW FLUX
      DO 510 I = 1,NG
      FLUX(I) = FS(I)
510    CONTINUE

C      ZERO ARRAYS TO BEGIN NEW ACCUMULATIONS
517    DO 511 I = 1,NG
      JTHETA(I) = 0.
      CM(I) = 0.
      CZ(I) = 0.
      CP(I) = 0.
511    CONTINUE

      IF (KWAVE .NE. 0.) RETURN

C      IF NO WAVE HAS BEEN LAUNCHED. TRY TO START ETHETA UP
C      SMOOTHLY
      IF (ITER - 1) 512,514,516

512    DO 513 I = 1,NG
      ETHETA(I) = 0.
513    CONTINUE

      RETURN

514    DO 515 I = 1,NG
      ETHETA(I) = ETHETA(I) / 2.
515    CONTINUE

      RETURN

C.....

516    RETURN
      END

      SUBROUTINE HEADER
      INCLUDE 'MAGIC2.STR'
      DIMENSION BUFF(15)
C.....

C      THIS PLOTS THE OUTPUT HEADING PAGE

      CALL SETCH (2.,60.,1,0,0,0)
      WRITE (100,1500)
1500    FORMAT('MAGIC2 - NO ELECTRON PHYSICS')
      CALL DATE (DATE1)
      CALL TIME (TIME1)
      WRITE (5,1501) DATE1,TIME1

```

```

1501      WRITE (100,1501) DATE1,TIME1
        FORMAT(X,A10,A5)

1502      WRITE (100,1502) FNAME
        FORMAT('INPUT FROM FILE: ',2A5)

        OPEN (UNIT=20,ACCESS='SEQIN',FILE=FNAME)
        WRITE (100,1503)
1503      FORMAT('INPUT FILE FOLLOWS:')
        WRITE (100,1504)
1505      READ (20,1504,END=1506) BUFF
1504      FORMAT(15A5)
        WRITE (100,1504) BUFF
        GO TO 1505
1506      CLOSE (UNIT=20)

        CALL SETCH (70.,60.,1.0,0.0)

        WRITE (100,1545) NHOT,NCOLD,SUPN,BETA
1545      FORMAT('NHOT = ',16,
1          /'NCOLD = ',16,
1          /'SUPN(1) = ',E12.3,
1          /'SUPN(2) = ',E12.3,
1          /'BETA(1) = ',E12.3,
1          /'BETA(2) = ',E12.3)

        WRITE (100,1550) B0,DELTS,RMAX,DENSI,CURR,NUS,ITRMAX,
1          ITRPLT
1550      FORMAT ('B0 = ',F15.2,
1          /'DELTS = ',F12.2,
1          /'RMAX = ',F12.1,
1          /'DENSI = ',E11.1,
1          /'CURR = ',E12.1,
1          /'NUS = ',E12.3,
1          /'ITRMAX = ',19,
1          /'ITRPLT = ',19)

        WRITE (100,1551) IVRR,IVTHR,IVTHVR,IDENS,IBZS,IATHS,IETHS,IJTH
1551      FORMAT ('IVRR = ',111,
1          /'IVTHR = ',110,
1          /'IVTHVR = ',19,
1          /'IDENS = ',110,
1          /'IBZS = ',111,
1          /'IATHS = ',110,
1          /'IETHS = ',110,
1          /'IJTH = ',111)

        WRITE (100,1552) INJMOD,NINJ,RINJ,WINJ,EINJ
1552      FORMAT ('INJMOD = ',19,
1          /'NINJ = ',111,
1          /'RINJ = ',F13.2,
1          /'WINJ = ',F13.2,
1          /'EINJ = ',E13.2)

        WRITE (100,1553) NGYRO,GYRRAN,TEMPO
1553      FORMAT ('NGYRO = ',19,
1          /'GYRRAN = ',F12.3,
1          /'TEMPO = ',E12.2)

        WRITE (100,1554) WCI,WALF,VALF,WPI,WWAVE

```

```

1554  FORMAT ('WC1 = ',E13.3,
1      /'HALF = ',E13.3,
1      /'VALF = ',E13.3,
1      /'WPI = ',E13.3,
1      /'WAVE = ',E13.3)

```

```

CALL FRAME

```

```

C.....
RETURN
END

```

```

SUBROUTINE INITLZ
INCLUDE 'MAGIC2.STR'

```

```

C.....

```

```

C      THIS INITIALIZES THE FIELDS

```

```

C      ZERO GRID ARRAYS

```

```

DO 600 I = 1,NG
CM(I) = 0.
CZ(I) = 0.
CP(I) = 0.
BZ(I) = 0.
ETHETA(I) = 0.
FLUX(I) = 0.
JTHETA(I) = 0.
600 CONTINUE

```

```

ENRTOT = 0.
R1 = 0.

```

```

C      CALCULATE ENERGY DUE TO BACKGROUND MAGNETIC FIELD
CONST = DELRS * C * DELT

```

```

DO 601 I = 1,NG-1
R2 = FLOAT(I) * CONST
ENRTOT = ENRTOT + ((B0 ** 2) / 8.) * (R2 ** 2 - R1 ** 2)
R1 = R2
601 CONTINUE

```

```

IF (KWAVE .EQ. 0.) RETURN

```

```

C      IF A WAVE IS TO BE LAUNCHED, SET UP ETHETA.

```

```

DO 602 I = 1,NG
RAD = (I - 1) * DELR
ETHETA(I) = AWAVE * J1(KWAVE * RAD)
602 CONTINUE

```

```

C.....

```

```

RETURN
END

```

```

SUBROUTINE INJECT
INCLUDE 'MAGIC2.STR'

```

```

C.....

```

```

C      THIS DOES THE SUPERPARTICLE INJECTION

      NLAST = NMAX(2)
      NIN = 0

C      SELECT INJECTION MODE, AND CALCULATE HOW MANY SUPERPARTICLES
C      ARE TO BE INJECTED
C      INJMOD = 0 : ALL INJECTION AT T = 0.
C      = 1 , CONSTANT RATE
C      = 2 , LINEARLY INCREASING RATE
      IF ((INJMOD.EQ. 0).AND. ITER.EQ. 0) NIN = NINJ
      IF (INJMOD.EQ. 1) NIN = NINJ
      IF (INJMOD.EQ. 2) NIN = NINJ * ITER
C      IF NO SUPERPARTICLES ARE TO BE INJECTED, THEN RETURN
      IF (NIN.EQ. 0) RETURN

C      SET UP PARAMETERS TO GIVE PARABOLIC INJECTION PROFILE
      R1 = RINJS - WINJS / 2.
      TS1 = RINJS / WINJS
      TS2 = 2. / NIN

C      INJECTION LOOP
      DO 700 I = NMAX(2)+1,NMAX(2)+(NGYRO*NIN),NGYRO

C      CALCULATE RADIUS FOR NEXT SUPERPARTICLES FROM PARABOLIC PROFILE
      TS3 = R1 / WINJS
      TS4 = TS3 * TS3
      TS5 = TS3 * TS4
      TS4 = TS1*(3.-4.*TS1*(TS1-3.*TS3)-12.*TS4)+4.*TS5-3.*TS3-TS2
      IF (TS4.LT. -1.) TS4 = -1.
      TS6 = RINJS*WINJS*COS((4.*PI)+ACOS(SNGL(TS4)))/3.)
      TS3 = (R1+TS6)/2.
C      TS3 IS THE RADIUS FOR THE NEXT BATCH OF SUPERPARTICLES

C      NOW INJECT NGYRO OF SUPERPARTICLES AT THIS RADIUS WITH NGYRO
C      DIFFERENT GYROPHASE ANGLES AS DISTRIBUTED UNIFORMLY
C      BETWEEN -GYRRAN AND +GYRRAN
      DO 701 K = 0,NGYRO-1

C      CALCULATE THE GYROPHASE ANGLE FOR THE NEXT SUPERPARTICLE
      IF (NGYRO.GT. 1) THETA = - GYRRAN
      + (2. * FLOAT(K) * GYRRAN / (NGYRO - 1))
      IF (NGYRO.EQ. 1) THETA = 0.

C      GIVE IT THE PROPER RADIUS, VR, AND VTHETA
      R(I+K) = TS3
      VR(I+K) = VINJS * SIN(THETA)
      VTHETA(I+K) = VINJS * COS(THETA)

C      GET THE LINEAR WEIGHTING FACTORS TO CALCULATE PTHETA
      RPGC = R(I+K)/DELR5
      J = INT(RPGC)
      DR = RPGC - FLOAT(J)
      DRC = 1. - DR
      PTHETA(I+K) = R(I+K) * VTHETA(I+K) / DELTS
      + (DRC * FLUX(J+1) + DR * FLUX(J+2)) / DELTS
      + (R(I+K) * 2) / 2.

C      GET THE C'S FOR THE FIELD SOLVER

```

```

RISQ = 1. / (R(I+K) ** 2)

CM(J+2) = CM(J+2) - BETA(2) * DRC * DR * RISQ
CZ(J+1) = CZ(J+1) - BETA(2) * DR * DR * RISQ
CZ(J+2) = CZ(J+2) - BETA(2) * DRC * DRC * RISQ
CP(J+1) = CM(J+2)

C      ADD ON THE JTHETA DUE TO THIS NEW SUPERPARTICLE,
C      EVEN THOUGH THE VELOCITIES ARE OFF BY HALF TIME STEP
      JTHETA(J+1) = JTHETA(J+1) + BETA(2) * DRC *
      ( -.5 + PTHETA(I+K) * RISQ)
      JTHETA(J+2) = JTHETA(J+2) + BETA(2) * DR *
      ( -.5 + PTHETA(I+K) * RISQ)

C      ADD ON THE KINETIC ENERGY DUE TO THIS SUPERPARTICLE
      EKin = SUPN(2) * (M / 2.) * (C ** 2)
      * (VTHETA(I+K) ** 2 + VR(I+K) ** 2)
      ENRTOT = ENRTOT + EKin
701    CONTINUE

      R1 = TSG
700    CONTINUE

C      MOVE UP NMAX(2) BY THE NUMBER OF SUPERPARTICLES
C      INJECTED THIS TIME
      NMAX(2) = NMAX(2) + NGYRO * NIN

C.....

      RETURN
      END

      SUBROUTINE INPUT
      INCLUDE 'MAGIC2.STR'
C.....

C      THIS INPUTS THE RUN DATA

      DIMENSION INAME(15)

      NAMELIST /INPUT/ B0,DELTS,RMAX,DENS1,CURR,NUS,ITRMAX,ITRPLT
      ,IVRR,IVTHR,IVTHVR,IDENS,IBZS,IFLXS,IETHS
      ,IJTH,IATHS
      ,INJMOD,NINJ,RINJ,WINJ,EINJ,GYRRAN,NGYRO
      ,TEMPO,TWALL,ITEK
      ,KWAVE,AWAVE
      ,IPT1,IPT2,IPT3,IPT4

      DATA B0/7000./,DELTS/.2/,RMAX/10./,DENS1/0./,NUS/0./
      ,CURR/1.E+3/,ITRMAX/10/,ITRPLT/1/
      ,IVRR/1/,IVTHR/0/,IVTHVR/1/,IDENS/1/,IBZS/1/,IFLXS/1/
      ,IETHS/1/,IJTH/1/,IATHS/1/
      ,INJMOD/0/,NINJ/5/,RINJ/5./,WINJ/8./
      ,EINJ/40./,NGYRO/8/,GYRRAN/1.57/
      ,TEMPO/.20/,TWALL/10./,ITEK/0/
      ,KWAVE/0./,AWAVE/0./
      ,IPT1/10/,IPT2/20/,IPT3/30/,IPT4/40/

      DATA INAME/15*'H /,IDOT/1H./

```



```

      WRITE (5,800)
800  FORMAT (X,'INPUT FILE: ' $)
      READ (5,801) INAME
801  FORMAT (15A1)

      IFLAG = 0

C     THIS LOOP CHECKS TO SEE IF THE DOT WAS LEFT OFF THE FILENAME
C     IF IT WAS, IT IS PUT BACK IN
      DO 803 I = 1,15
      IF (INAME(I) .EQ. IDOT) IFLAG=1
      IF (INAME(I) .NE. IH ) GO TO 803
      IF (IFLAG .EQ. 0) INAME(I)=IDOT
      GO TO 804
803  CONTINUE

804  ENCODE (10,805,FNAME) (INAME(I),1=1,10)
805  FORMAT (10A1)

      OPEN (UNIT=20,ACCESS='SEQIN',FILE=FNAME)

      READ (20,INPUT)

      CLOSE (UNIT=20)

```

```

C.....
      RETURN
      END

```

```

      SUBROUTINE IPLOT
      INCLUDE 'MAGIC2.STR'

```

```

C.....
C     THIS MANAGES THE PLOT OUTPUT
C     BY INCREMENTING THE JPLOT COUNTER, CALLING THIS SUBROUTINE
C     GENERATES THE DIFFERENT OFFSETS NECESSARY TO GET FOUR PLOTS
C     PER PAGE

      DATA JPLOT/1/

      GOTO (1,2,3,4,5), JPLOT

1     XOFF = 0.
      YOFF = .7
      GO TO 6
2     XOFF = 0.
      YOFF = .32
      GO TO 6
3     XOFF = .51
      YOFF = .7
      GO TO 6
4     XOFF = .51
      YOFF = .32
      GO TO 6
5     CALL SETCH(20,.8,.1,0,2,0)
      TIME = FLOAT(ITER) * DELT
      WRITE (100,7) ITER,TIME

```

```

7      FORMAT ('ITER = ',I4,5X,'TIME = ',1PE12.2)
      CALL FRAME
      JPLOT = 1
      GO TO 1
6      JPLOT = JPLOT + 1

```

```

C.....

      RETURN
      END

```

```

      DOUBLE PRECISION FUNCTION J1(RAD)
      IMPLICIT DOUBLE PRECISION (A-Z)

```

```

C.....

C      BESSEL FUNCTION FOR GENERATING WAVES

```

```

      IF (RAD .GT. 3.) GO TO 450

      RAD3 = (RAD / 3.) ** 2

      J1 = .5 - .56249985 * RAD3 + .21093573 * RAD3 ** 2
1      - .03954289 * RAD3 ** 3 + .00443319 * RAD3 ** 4
1      - .00031761 * RAD3 ** 5 + .00001109 * RAD3 ** 6

      J1 = RAD * J1

      RETURN

```

```

450      RAD3 = 3. / RAD

      F1 = .79788456 + .00000156 * RAD3
1      + .01659667 * RAD3 ** 2 + .00017105 * RAD3 ** 3
1      - .00249511 * RAD3 ** 4 + .00113653 * RAD3 ** 5
1      - .00020033 * RAD3 ** 6

      T1 = -2.35619449 + .12499612 * RAD3
1      + .00005650 * RAD3 ** 2 - .00637879 * RAD3 ** 3
1      + .00074348 * RAD3 ** 4 + .00079824 * RAD3 ** 5
1      - .00029166 * RAD3 ** 6

      T1 = RAD * T1

      J1 = F1 * COS(T1) / SQRT(RAD)

```

```

C.....

      RETURN
      END

```

```

      SUBROUTINE JTHR
      INCLUDE 'MAGIC2.STR'

```

```

C.....

C      THIS PLOTS JTHETA VERSUS R

      CALL IPLOT

      CALL MAP (0...5,0...3,XOFF,XOFF+.5,YOFF,YOFF+.3)

```

```

CALL SETLCH (0...13,1.0,0,1)
WRITE (100,502)
502 FORMAT ('THETA CURRENT')
CALL SETLCH (1.217,-.055,1.0,0,0)
WRITE (100,503)
503 FORMAT ('RADIAL POSITION (CM)')

CONST = - 80 / (4. * PI * WC1 * DELT ** 2)

DO 500 I = 2,NG-1
  RG = (I - 1) * DELRS
  TV1(I) = (FLUX(I+1) - 2. * FLUX(I) + FLUX(I-1))
  1          / (RG * DELRS ** 2)
  1          - (FLUX(I+1) - FLUX(I-1))
  1          / (2. * RG ** 2 * DELRS)
  TV1(I) = TV1(I) * CONST
  TV2(I) = (I - 1) * DELR
500 CONTINUE

  TV1(1) = 0.
  TV1(NG) = 0.
  TV2(1) = 0.
  TV2(NG) = RMAX

  CALL CARTMM (NG,YMIN,YMAX,TV1,2)
  YMIN = 1.3*AMIN1(SNGL(YMIN),0.)
  YMAX = 1.3*AMAX1(0.,SNGL(YMAX))
  CALL MAPS (0.,RMAX,YMIN,YMAX,
  1          XOFF+.055,XOFF+.49,YOFF,YOFF+.3)

  DO 504 I = 1,NG-1
  CALL LINE(TV2(I),TV1(I),TV2(I+1),TV1(I+1))
  CALL LINE(TV2(I+1),TV1(I+1),TV2(I+1),TV1(I+1))
504 CONTINUE

C.....

RETURN
END

SUBROUTINE MAXLOD
INCLUDE 'MAGIC2.STR'
C.....

C      THIS LOADS A MAXWELLIAN BACKGROUND

C      IF NO COLD SUPERPARTICLES ARE TO BE PUT IN, THEN RETURN
      IF (NCOLD .EQ. 0) RETURN

C      THIS LOOP INITIALIZES THE RANDOM NUMBER GENERATOR
      CALL TIME(J,N)
      N = (N .AND. #3777700000) / #100000
      DO 200 J = 1,N
      B = RAN(-1)
200 CONTINUE

C      CALCULATE THE RADII FOR THE COLD SUPERPARTICLES
      RHO = 2. * FLOAT(NCOLD) / (((FLOAT(NG) - 1) ** 2) * DELR ** 2)
      I = 1

```

```

DO 210 N = 1,NG-1

C   FIGURE OUT HOW MANY COLD SUPERPARTICLES GO IN THIS GRID CELL
NPCELL = .5 + RHO * (FLOAT(N) - .5) * DELR ** 2

DO 215 J = 1,NPCELL

C   GIVE EACH SUPERPARTICLE A RANDOM RADIUS WITHIN THE CELL
R(I) = DELRS * (FLOAT(N) - RAN(-1))
I = I + 1
215  CONTINUE

210  CONTINUE

IF (TEMPO .EQ. 0.) GO TO 250

C   IF TEMPO <> 0, THEN RANDOMLY INTERCHANGE RADII
DO 220 I = 1,NCOLD
NSWP = 1 + INT((NCOLD - 1) * RAN(-1))
SWP = R(I)
R(I) = R(NSWP)
R(NSWP) = SWP
220  CONTINUE

C   PUT IN VTHETA AND VR FOR EACH SUPERPARTICLE FROM MAXWELLIAN
NBINS = 10 * NCOLD
CONST = 4. * PI * FLOAT(NCOLD)
      * (DSQRT(1. / (PI * VTHERM ** 2))) ** 3
1   DELV = 4. * VTHERM / FLOAT(NBINS)
I = 1
VOLD = 0.
VNEW = 0.
VRTOT = 0.
VTTOT = 0.
FOLD = 0.
FNEW = 0.

C   INTEGRATE THE MAXWELLIAN UNTIL TIME TO PUT IN
C   ANOTHER SUPERPARTICLE
DO 230 K = 1,NBINS
VNEW = VNEW + DELV
FNEW = FNEW + CONST * (VNEW ** 2) *
      * EXP(-(VNEW ** 2) / (VTHERM ** 2)) * DELV
1   IF ((FNEW - FOLD) .LT. 1.) GO TO 230
VAVG = .5 * (VOLD + VNEW)
V1 = VAVG * (.1 + .10 * RAN(-1))

C   GIVE SUPERPARTICLE RANDOM PITCH ANGLE
THETA = 2. * PI * RAN(-1)
VR(I) = V1 * COS(THETA)
VRTOT = VRTOT + VR(I)
VTHETA(I) = V1 * SIN(THETA)
VTTOT = VTTOT + VTHETA(I)
EKIN = SUPN(I) * (M / 2.) * (C ** 2)
      * (VTHETA(I) ** 2 + VR(I) ** 2)
1   ENRTOT = ENRTOT + EKIN
I = I + 1
VOLD = VNEW
FOLD = FOLD + 1.
230  CONTINUE

```

```

C      CALCULATE THE AVERAGE VELOCITIES
      VRAVG = VRTOT / (I - 1)
      VTAVG = VTTOT / (I - 1)

      DO 240 I = 1, NCOLD

C      SUBTRACT OFF THE AVERAGE VELOCITIES SO THAT THE
C      NEW AVERAGE VELOCITIES WILL BE ZERO.
      VR(I) = VR(I) - VRAVG
      VTHETA(I) = VTHETA(I) - VTAVG

C      CALCULATE LINEAR WEIGHTING FACTORS TO GET PTHETA
      RPGC = R(I)/DELRS
      J = INT(RPGC)
      DR = RPGC - FLOAT(J)
      DRC = 1. - DR
      PTHETA(I) = R(I) * VTHETA(I) / DELTS
1      + (DRC * FLUX(J+1) + DR * FLUX(J+2)) / DELTS
1      + (R(I) ** 2) / 2.

C      CALCULATE THE C'S FOR THE FIELD SOLVER
      RISQ = 1. / (R(I) ** 2)

      CM(J+2) = CM(J+2) - BETA(I) * DRC * DR * RISQ
      CZ(J+1) = CZ(J+1) - BETA(I) * DR * DR * RISQ
      CZ(J+2) = CZ(J+2) - BETA(I) * DRC * DRC * RISQ
      CP(J+1) = CM(J+2)

C      ADD ON JTHETA FOR NEW SUPERPARTICLE
      JTHETA(J+1) = JTHETA(J+1) + BETA(I) * DRC *
1      ( -.5 + PTHETA(I) ) * RISQ
      JTHETA(J+2) = JTHETA(J+2) + BETA(I) * DR *
1      ( -.5 + PTHETA(I) ) * RISQ

240  CONTINUE

      RETURN

C      DO THIS LOOP IF THE SUPERPARTICLES ARE ALL COLD
250  DO 260 I = 1, NCOLD
      VR(I) = 0.
      VTHETA(I) = 0.

C      CALCULATE LINEAR WEIGHTING FACTORS TO GET PTHETA
      RPGC = R(I)/DELRS
      J = INT(RPGC)
      DR = RPGC - FLOAT(J)
      DRC = 1. - DR
      PTHETA(I) = (DRC * FLUX(J+1) + DR * FLUX(J+2)) / DELTS
1      + (R(I) ** 2) / 2.

C      CALCULATE C'S FOR FIELD SOLVER
      RISQ = 1. / (R(I) ** 2)

      CM(J+2) = CM(J+2) - BETA(I) * DRC * DR * RISQ
      CZ(J+1) = CZ(J+1) - BETA(I) * DR * DR * RISQ
      CZ(J+2) = CZ(J+2) - BETA(I) * DRC * DRC * RISQ
      CP(J+1) = CM(J+2)

```

```

C      ADD ON JTHETA FOR NEW SUPERPARTICLE
      JTHETA(J+1) = JTHETA(J+1) + BETA(1) * DRC *
1      ( -.5 + PTHETA(1) * RISQ)
      JTHETA(J+2) = JTHETA(J+2) + BETA(1) * DR *
1      ( -.5 + PTHETA(1) * RISQ)

260    CONTINUE

      IF (KWAVE .EQ. 0.) RETURN

C      IF A WAVE IS TO BE LAUNCHED, THEN SET UP VR AND ACCUMULATE
C      EKIN FOR ALL THE COLD SUPERPARTICLES
      DO 270 I = 1, NCOLD
      RAD = R(1) * C * DELT
      VR(1) = AMP * J1(KWAVE * RAD)
      EKIN = SUPN(1) * (M / 2.) * (C ** 2)
1      * (VR(1) ** 2)
      ENRTOT = ENRTOT + EKIN
270    CONTINUE

C.....
      RETURN
      END

      SUBROUTINE SETCON
      INCLUDE 'MAGIC2 STR'

C.....

C      THIS SUBROUTINE SETS THE VALUES OF CONSTANTS

C      SET UP CONSTANTS
      PI = 3.1415
      C = 3E+10
      E = 4.803E-10
      M = 3.343E-24
      WCI = E*B0/(M*C)
      DELT = DELTS / WCI
      DELR = RMAX / (NG - 1)
      DELRS = DELR / (C * DELT)
      NUS = NUS / 2.
      RINJS = RINJ/(C*DELT)
      WINJS = WINJ/(C*DELT)
      VINJS = - DSQRT(6.4E-9*EINJ/M)/C
      VTHERM = DSQRT(3.2E-9*TEMP0/M)/C
      VTHALL = DSQRT(3.2E-12*THALL/M)/C
      AWAVE = AWAVE * DELTS / B0
      IF (DENS1 .EQ. 0.) GO TO 505
      RHO = M * DENS1
      VALF = B0 / DSQRT(4. * PI * RHO)
      WPI = DSQRT(4. * PI * DENS1 * (E ** 2) / M)
      WALF = KWAVE * VALF
      S = (DSIN(KWAVE * DELR / 2.) / (KWAVE * DELR / 2.)) ** 2
      DENOM = 1. + (WCI * DELT / 2.) ** 2 + (KWAVE * C / WPI) ** 2 / S
      NUMER = (WALF * DELT / 2.) ** 2 / S
      WWAVE = (2. / DELT) * DATAN(DSQRT(NUMER / DENOM))

      DENOM = (WCI * DELT / 2.) ** 2 +
1      (DSIN(WWAVE * DELT / 2.) / DCOS(WWAVE * DELT / 2.)) ** 2
      NUMER = (WCI * DELT / 2.) ** 2 / DCOS(WWAVE * DELT / 2.)

```

```

AMP = AHAVE * (S / DELTS) * (NUMER / DENOM)

505  NLAST = 0

      IF (CURR .EQ. 0.) GO TO 514

C     CONVERT CURR TO CGS FROM MKS (AMPS TO ESU / SEC)
      CURR = CURR * (E / 1.602E-19)

C     SELECT DESIRED INJECTION MODE
      GO TO (510,520,530) , INJMOD + 1

C     CALCULATE THE NUMBER OF SUPERPARTICLES TO BE INJECTED
C     AND THE CHARGE PER SUPERPARTICLE
510  NHOT = NINJ * NGYRO
      ESUP = CURR * DELT / (NINJ * NGYRO)
      GO TO 550

520  NHOT = ITRMAX * NINJ * NGYRO
      ESUP = CURR * DELT / (NINJ * NGYRO)
      GO TO 550

530  NHOT = (ITRMAX * (ITRMAX + 1) / 2.) * NINJ * NGYRO
      ESUP = CURR * DELT / (ITRMAX * NINJ * NGYRO)

C     CALCULATE THE NUMBER OF PARTICLES PER SUPERPARTICLE
550  SUPN(2) = ESUP / E
      QPP = ESUP / (2. * PI * DELR)

C     CALCULATE THE BETA FOR THE FIELD SOLVER
      BETA(2) = 4. * PI * QPP * DELTS / B0

      IF (NHOT .LE. NP) GO TO 560
      WRITE (5,1000) NHOT
      FORMAT(' NHOT = ',15,' IS TOO LARGE.')
      CALL EXIT

1000

C     SET UP COLD SUPERPARTICLES
560  IF (DENSJ .NE. 0.) GO TO 570
      NCOLD = 0
      NMIN(1) = 0
      NMAX(1) = -1
      NMIN(2) = 1
      NMAX(2) = 0
      GO TO 513

570  NCOLD = 100 * INT((NP - NHOT) / 100.)
      IF (NCOLD .NE. 0) GO TO 571
      WRITE (5,572)
572  FORMAT(' NO PARTICLES LEFT FOR COLD PLASMA.')
      CALL EXIT
571  SUPN(1) = (DENSJ * PI * RMAX ** 2) / NCOLD
      QPP = SUPN(1) * E / (2. * PI * DELR)
      BETA(1) = (4. * PI * QPP * DELTS) / B0
      NMIN(1) = 1
      NMAX(1) = NCOLD
      NMIN(2) = NCOLD + 1
      NMAX(2) = NCOLD
      GO TO 513

514  NCOLD = 100 * INT(NP / 100.)

```

```

SUPN(1) = (DENS1 * PI * RMAX ** 2) / NCOLD
QPP = SUPN(1) * E / (2. * PI * DELR)
BETA(1) = (4. * PI * QPP * DELTS) / B0
NMIN(1) = 1
NMAX(1) = NCOLD
NMIN(2) = NCOLD + 1
NMAX(2) = NCOLD

513 WRITE(5,512) NHOT,NCOLD,SUPN
512 FORMAT(' NHOT = ',I6/' NCOLD = ',I6/
      1 ' QCOLD = ',E12.3/' QHOT = ',E12.3)
ENRTOT = 0.
KSAMPL = 1
ISAMPL = 1
KMAX = NT
IF (ITRMAX .LT. (NT - 1)) KMAX = ITRMAX + 1
IF (ITRMAX .GT. (NT - 1)) ISAMPL = 1 + ITRMAX / (NT - 1)
IF (ITRMAX .GT. (NT - 1)) KMAX = 1 + ITRMAX / ISAMPL

C .....

RETURN
END

SUBROUTINE SPLIT
INCLUDE 'MAGIC2.STR'

C .....

C THIS DOES THE TIME SAMPLING

DO 100 I = 1,NG
TV1(I) = 0.
TV2(I) = 0.
100 CONTINUE

C ACCUMULATE TOTAL KINETIC ENERGY FOR EACH GRID CELL
DO 200 IGROUP = 1,NGROUP
IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 200

DO 201 I = NMIN(IGROUP),NMAX(IGROUP)
J = 1. + R(I) / DELRS
TV1(J) = TV1(J) + (M / 2.) * (C ** 2) * SUPN(IGROUP)
      1 * (VR(I) ** 2 + VTHETA(I) ** 2)
201 CONTINUE

200 CONTINUE

R1 = 0.
ENER = 0.

C CALCULATE THE BETA OF EACH GRID CELL AND THE TOTAL ENERGY
DO 300 I = 1,NG-1
R2 = FLOAT(I) * DELRS * C * DELT
TV2(I) = TV1(I) / (PI * (R2 ** 2 - R1 ** 2))
TV2(I) = TV2(I) / ((B0 ** 2) / (8. * PI))
BINT = B0 + (B0 / DELTS) * .5 * (BZ(I) + BZ(I+1)).
ENER = ENER + TV1(I) + ((BINT ** 2) / 8.)
      1 * (R2 ** 2 - R1 ** 2)

```



```

300      R1 = R2
        CONTINUE

C      GET THE MAXIMUM BETA VALUE ON THE GRID
        CALL CARTMM (NG,TS1,TS2,TV2,2)
        BMAX(KSAMPL) = TS2

        TS1 = 0.

C      CALCULATE THE AVERAGE BETA ON THE GRID
        DO 400 I = 1,NG
          TS1 = TS1 + TV2(I)
400      CONTINUE

        BAVG(KSAMPL) = TS1 / NG

C      SAVE THE ENERGY SAMPLES
        TENERC(KSAMPL) = ENER
        TENERI(KSAMPL) = ENRTOT

C      SAMPLE AT HETA AT IPT?
        ATTIME(1,KSAMPL) = FLUX(IPT1) / ((IPT1 - 1) * DELRS)
        ATTIME(2,KSAMPL) = FLUX(IPT2) / ((IPT2 - 1) * DELRS)
        ATTIME(3,KSAMPL) = FLUX(IPT3) / ((IPT3 - 1) * DELRS)
        ATTIME(4,KSAMPL) = FLUX(IPT4) / ((IPT4 - 1) * DELRS)

C      SAVE THE TIME AT WHICH THIS SAMPLE WAS TAKEN
        XTIME(KSAMPL) = FLOAT(ITER) * DELT
        KSAMPL = KSAMPL + 1

C .....

        RETURN
        END

        SUBROUTINE VRR
        INCLUDE 'MAGIC2.STR'
C .....

C      THIS PLOTS VR VERSUS R

        COMMON /CRIVLS/ CPL(36)

        CALL IPLOT

        TS1 = 0.
        TS2 = 0.

        CALL MAP (0.,.5,0.,.3,XOFF,XOFF+.5,YOFF,YOFF+.3)
        CALL SETLCH (0.,.084,1,0,0,1)
        WRITE (100,1100)
1100      FORMAT ('RADIAL VELOCITY (CM/SEC)')
        CALL SETLCH (.217,-.055,1,0,0,0)
        WRITE (100,1101)
1101      FORMAT ('RADIAL POSITION (CM)')

        TS1 = 1.E+25
        TS2 = -1.E+25

        DO 1105 IGROUP = 1,NGROUP

```

```

      IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 1105
      DO 1106 I = NMIN(IGROUP),NMAX(IGROUP)
      IF (VR(I) .GT. TS2) TS2 = VR(I)
      IF (VR(I) .LT. TS1) TS1 = VR(I)
1106  CONTINUE
1105  CONTINUE

      YMIN = 1.3*AMIN1(SNGL(TS1),0.)
      YMAX = 1.3*AMAX1(0.,SNGL(TS2))
      CALL MAPS (0.,RMAX,C*YMIN,C*YMAX,
      )      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
      CALL MAP (0.,RMAX/(C*DELT),YMIN,YMAX,
      )      XOFF+.055,XOFF+.49,YOFF,YOFF+.3)

      DO 1110 IGROUP = 1,NGROUP
      IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 1110

      DO 1111 I = NMIN(IGROUP),NMAX(IGROUP)
      CALL POINT (R(I),VR(I))
1111  CONTINUE
1110  CONTINUE
C.....
      RETURN
      END

```

```

      SUBROUTINE VTHVR
      INCLUDE 'MAGIC2.STR'
C.....
C      THIS PLOTS VTHETA VERSUS VR

      CALL IPLOT

      TS1 = 0.
      TS2 = 0.
      TS3 = 0.
      TS4 = 0.

      CALL MAP (0...5,0...3,XOFF,XOFF+.5,YOFF,YOFF+.3)
      CALL SETLCH (0...075,1,0,0,1)
      WRITE (100,1200)
1200  FORMAT ('THETA VELOCITY (CM/SEC)')
      CALL SETLCH (.205,-.055,1,0,0,0)
      WRITE (100,1201)
1201  FORMAT ('RADIAL VELOCITY (CM/SEC)')

      TS1 = 1.E+25
      TS2 = -1.E+25
      TS3 = 1.E+25
      TS4 = -1.E+25

      DO 1210 IGROUP = 1,NGROUP
      IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 1210

```

```

DO 1211 I = NMIN(IGROUP),NMAX(IGROUP)
IF (VR(I) .GT. TS2) TS2 = VR(I)
IF (VR(I) .LT. TS1) TS1 = VR(I)
IF (VTHETA(I) .GT. TS4) TS4 = VTHETA(I)
IF (VTHETA(I) .LT. TS3) TS3 = VTHETA(I)
1211 CONTINUE

1210 CONTINUE

XMAX = 1.3*AMAX1(ABS(SNGL(TS1)),ABS(SNGL(TS2)))
YMAX = 1.3*AMAX1(ABS(SNGL(TS3)),ABS(SNGL(TS4)))
CALL MAPS (-C*XMAX,C*XMAX,-C*YMAX,C*YMAX,
           XOFF+.055,XOFF+.49,YOFF,YOFF+.3)
CALL MAP(-XMAX,XMAX,-YMAX,YMAX,
           XOFF+.055,XOFF+.49,YOFF,YOFF+.3)

DO 1215 IGROUP = 1,NGROUP
IF (NMAX(IGROUP) .LT. NMIN(IGROUP)) GO TO 1215

DO 1216 I = NMIN(IGROUP),NMAX(IGROUP)
CALL POINT (VR(I),VTHETA(I))
1216 CONTINUE

1215 CONTINUE

C.....

RETURN
END

SUBROUTINE MAGOOC
C GLOSSARY OF VARIABLE NAMES
C.....

C VARIABLES IN COMMON BLOCKS IN FILE MAGIC2.STR

C PARAMETERS
C NP MAXIMUM NUMBER OF PARTICLES ALLOWED
C NG NUMBER OF GRID POINTS TO BE USED
C NT MAXIMUM NUMBER OF TIME SAMPLES ALLOWED

C NG NUMBER OF GROUPS OF PARTICLES
C ( PRESENTLY 2, FOR HOT AND COLD )

C /INDATA/
C BD VACUUM MAGNETIC FIELD IN Z DIRECTION IN GAUSS
C DELT TIME STEP IN SECONDS
C RMAX RADIUS OF SYSTEM IN CENTIMETERS
C OENSI INITIAL DENSITY OF COLD PLASMA IN PARTICLES
C PER CUBIC CENTIMETER
C CURR CURRENT TO BE INJECTED IN AMPERES
C NUS DRAG COEFFICIENT
C ITRMAX NUMBER OF TIME STEPS TO BE DONE
C ITRPLT NUMBER OF TIME STEPS BETWEEN PLOTTING
C NINJ NUMBER OF PARTICLES TO BE INJECTED PER TIME STEP,
C PER INJECTION POINT, PER GYRO PHASE ANGLE
C RINJ RADIUS OF CENTER OF INJECTION IN CENTIMETERS

```

```

C      WINJ      WIDTH OF PARABOLIC INJECTION PROFILE IN CENTIMETERS
C      INJMOD    INJECTION MODE - 0. PULSE AT T = 0
C                      1. CONSTANT CURRENT INJECTION
C                      2. LINEAR RAMP CURRENT INJECTION
C      EINJ      ENERGY PER INJECTED PARTICLE IN KEV
C      NGYRO     NUMBER OF GYRO PHASE ANGLES PER INJECTION POINT
C      GYRRAN    RANGE OF GYRO PHASE ANGLES, FROM -GYRRAN TO +GYRRAN,
C                      GIVEN IN RADIANS
C      TEMPO     ENERGY PER COLD PARTICLE IN EV
C      TWALL     ENERGY GIVEN TO FAST PARTICLES WHICH ARE ABSORBED
C                      AND THEN RE-EMITTED BY THE WALL
C      ITEK      SWITCH WHICH TURNS ON PLOTTING OF OUTPUT TO
C                      TEKTRONICS TERMINAL

C      /RDATA/
C      ITER      NUMBER OF TIME STEPS COMPLETED
C      NIN        NUMBER OF PARTICLES PER INJECTION POINT, PER
C                      GYRO PHASE ANGLE FOR THIS STEP
C      NLAST      TOTAL NUMBER OF PARTICLES UP TO THIS INJECTION
C      VINJS      VELOCITY OF INJECTED PARTICLE IN CODE UNITS
C      RINJS      RADIUS OF CENTER OF INJECTION PROFILE IN CODE UNITS
C      WINJS      WIDTH OF INJECTION PROFILE IN CODE UNITS
C      ENRTOT     ACCUMULATED ( OVER TIME ) TOTAL ENERGY IN ERGS
C      NHOT       TOTAL NUMBER OF HOT PARTICLES TO BE INJECTED
C      NCOLD      TOTAL NUMBER OF COLD PARTICLES TO BE PRESENT AT STARTUP
C      VTHERM     THERMAL VELOCITY OF COLD PARTICLES IN CODE UNITS
C      VWALL      VELOCITY OF FAST PARTICLE RE-EMITTED FROM WALL

C      /PRCLS/
C      R          RADII OF PARTICLES IN CODE UNITS
C      VR         RADIAL VELOCITIES OF PARTICLES IN CODE UNITS
C      PTHETA     CANONICAL ANGULAR MOMENTUM OF PARTICLES IN CODE UNITS
C      VTHETA     THETA COMPONENT OF VELOCITIES OF PARTICLES IN CODE UNITS

C      /GRID/
C      FLUX       THETA COMPONENT OF SELF MAGNETIC FLUX AS FUNCTION OF
C                      RADIUS IN CODE UNITS
C      JTHETA     THETA COMPONENT OF CURRENT AS FUNCTION OF RADIUS
C                      IN CODE UNITS
C      BZ         Z COMPONENT OF SELF MAGNETIC FIELD IN CODE UNITS
C      ETHETA     THETA COMPONENT OF SELF ELECTRIC FIELD IN CODE UNITS
C      CH         GEOMETRICAL FACTORS USED IN ACCUMULATION OF JTHETA
C      CZ
C      CP
C      TV1
C      TV2        TEMPORARY VECTORS USED IN GRID CALCULATIONS

C      /CONST/
C      PI         3.1415
C      C          SPEED OF LIGHT - 3E+10
C      DELTS      TIME STEP IN CODE UNITS
C      DELR       GRID POINT SPACING IN CENTIMETERS
C      DELRS      GRID POINT SPACING IN CODE UNITS
C      E          ELECTRONIC CHARGE - 4.803E-10
C      M          PROTON MASS - 3.343E-24
C      MC1        10N CYCLOTRON FREQUENCY

C      /SWITCH/
C      IVRR       TURNS ON AND OFF VR VS. R PLOT

```

```

C      IVTHR      □   □   □   □   VTHETA VS. R PLOT
C      IVTHVR     □   □   □   □   VTHETA VS. VR PLOT
C      IDENS      □   □   □   □   DENSITY VS. R PLOT
C      IBZS       □   □   □   □   SELF B FIELD - Z COMPONENT VS. R PLOT
C      IATHS      □   □   □   □   SELF VECTOR POTENTIAL - THETA COMPONENT
C                                     VS. R PLOT
C      IETHS      □   □   □   □   SELF ELECTRIC FIELD - THETA COMPONENT
C                                     VS. R PLOT
C      IJTH       □   □   □   □   CURRENT - THETA COMPONENT VS. R PLOT

C      /PLOTS/
C      JPLOT      KEEPS TRACK OF WHERE TO PUT THE PLOTS ON A PAGE
C      XOFF       HORIZONTAL OFFSET TO BE USED FOR CURRENT PLOT
C      YOFF       VERTICAL OFFSET TO BE USED WITH CURRENT PLOT
C      FNAME      NAME OF INPUT DATA FILE
C      KSAMPL     NUMBER OF TIME SAMPLES THAT HAVE BEEN TAKEN SO FAR
C      ISAMPL     NUMBER OF TIME STEPS PER TIME SAMPLE
C      KMAX       MAXIMUM NUMBER OF TIME SAMPLES THAT WILL BE TAKEN

C      /HISTRY/
C      XTIME      TIME VALUES FOR X - AXIS OF TIME PLOTS
C      BAVG       AVERAGE BETA VERSUS TIME VALUES
C      BMAX       MAXIMUM BETA VERSUS TIME VALUES
C      TENERC     CALCULATED TOTAL ENERGY VERSUS TIME VALUES
C      TENERI     TOTAL INJECTED ENERGY VERSUS TIME VALUES
C      ATTIME     SELF VECTOR POTENTIAL VERSUS TIME VALUES FOR 4
C                  DIFFERENT RADII
C      IPT1
C      IPT2
C      IPT3       GRID POINT NUMBERS FOR ATTIME SAMPLING
C      IPT4

C      /GROUP/
C      NMIN       KEEPS VALUE OF PARTICLE ARRAY INDEX FOR FIRST PARTICLE
C                  OF EACH GROUP
C      NMAX       KEEPS VALUE OF PARTICLE ARRAY INDEX FOR LAST PARTICLE
C                  OF EACH GROUP
C      BETA       KEEPS CURRENT WEIGHTING FACTOR FOR FIELD SOLVER FOR
C                  EACH PARTICLE
C      SUPN       NUMBER OF PARTICLES PER SUPERPARTICLE

C      /WAVES/
C      HALF       ALFVEN FREQUENCY
C      KWAVE      MAGNETOSONIC WAVE NUMBER
C      AWAVE      MAGNETOSONIC WAVE AMPLITUDE FOR SELF ELECTRIC
C                  FIELD IN VOLTS
C      .....
C      END

```