BEST CYCLOTRON SYSTEMS INC.

CYCLONE VERS 10.13

Reference Guide

Bruce Milton

10/27/2014

The cyclotron orbit code CYCLONE is described, and the input parameters are defined. CYCLONE calculates orbits using both magnetic and electric field maps and is therefore very useful for central region design. Version 6.0 represents several major improvements over the original MSU version of the code. Version 7 has been modified to improve operation under UNIX. VERSION 8 offers a new input file structure and generalized spiral gaps. Version 8.4 must be compiled using a Fortran90 compiler as it includes dynamic allocation of arrays. Version 9 is PC only but includes stripping extraction and other new features. Version 10 now supports a MATLAB macro and calculation of inflector orbits.

Contents

1 Introduction

The orbit code CYCLONE has had a long history. It was originally developed in the mid 60's at MSU as part of the K50 cyclotron project[1]. It was later integrated with the program SPRGAP to allow for the calculation of cases involving spiral gap shapes. At a similar time multi-phase operation was also incorporated into the program[2]. This work was again done at MSU but for the K500 cyclotron project. The program has since been used to design the central regions for the MSU K1200, and K100 cyclotrons, and the Milan K800 cyclotron. In 1987 the program was brought to TRIUMF from MSU, and has been used in the design of the central region of the TR30 cyclotron. At TRIUMF the program has been modified to make it more flexible and to integrate it with the graphics program PHYSICA[4]. As well vertical motion has been added to part I, and the interface to the relaxation program RELAX3D[3] has been simplified.

CYCLONE is an orbit tracking code that can be used to study all regions of a cyclotron, but it is particularly suited to central region design work. Orbits are calculated using a Runga-Kutta[5] type integration in a variety of electric and magnetic field maps. The program is divided into three parts, each with a different integration approach. The first part integrates the equations of motion in the given electric and magnetic field maps using the RF time, τ, as the independent variable. This is useful near the centre of the cyclotron where the orbits may deviate a lot from a circle, it is however slow because of the need to look up the magnetic field values at each step. The second part again integrates the equations of motion in electric and magnetic field maps but the angle θ is taken as the independent variable. This is used for the first 5 or so turns in the cyclotron. The third part again uses θ as the independent variable, but the electric field is approximated by a delta function at the gaps. This is the fastest part and is used from about turn 5 to extraction. The program can step from one part to the next at the appropriate points. The program calculates median plane motion and vertical motion. In part III the vertical motion is linear. In Parts I and II the magnetic contribution is linear but the electric contribution can be either linear or non-linear depending on the electric field mesh used.

There is the ability to use separate electric field maps in Part I and Part II. These are known (for historical reasons) as the Small Field, and the Large Field, respectively. This allows a much finer mesh spacing to be used in Part I where the particle's energy is still quite low. Provision is provided to position these fields at arbitrary locations relative to the magnetic field. The field maps are stored in a format that is compatible with RELAX3D, so direct output from that code can be used as input. Both electric fields are stored in rectangular meshes. Version 9 now allows multiple z lanes in both Part I and Part II, so either the same field may be used or a different mesh spacing may be used between the two.

The magnetic field data is stored in a standard polar mesh. This is the same format as the files produced by the isochronous field generator, and are accepted as input to the equilibrium orbit code GENSPEO, and the tracking program GOBLIN. Equilibrium orbits can be computed by CYCLONE so that it can then calculate the x,px of the particle. There is a variety of configurations for EOs, so the x,px value can be calculated at either the gaps, at all angles, or along a fixed theta.

Various parameters can be used to set break points for switching from one part to the next. In some cases if the break points are not set then the program will chose it's own. Since the program can run either forward or backwards, this provides a variety of possible program flows. The orbit data is stored on unit 35 at all integration steps, and is printed on unit 6 at those positions that are requested. When switching from one part to another, unit 35 is maintained such that the orbit proceeds monotonically in the direction of motion.

2 UNITS

Although the units chosen for most parameters are obvious, it is necessary to point out the interaction between them. In general we define a length unit *'a'* and a field unit *'bcon'* as,

$a = c/\omega_0, \qquad bcon = m_0\omega_0/q.$

where c is the velocity of light, ω_0 is the orbital frequency (in radians/sec), m_0 is the rest mass, and q is the particle charge. The field unit should have the same dimensions as the magnetic field, generally (kG). The choice of units of momenta is not obvious, so to cover both possibilities we assume that,

$$Momentum\ unit = m_0\, c/a'$$

Where a' is some constant independent of energy. Thus in this unit,

$$p = a'\left[\frac{E}{m_0c^2}\left(2 + \frac{E}{m_0c^2}\right)\right]^{1/2},$$

where E is the kinetic energy of the particle. If $a' = 10^3$, then the units of momenta will be milli-radians. On the other hand if we take $a' = a$ then all momenta have length units. In CYCLONE 'a' is determined using the values of BCON, E0 and the length unit that are input, so there is no problem with inconsistencies. The program is such that all input must be made using the same length units, whether they be inches, centimeters or millimeters. The unit of energy is always assumed to be MeV. Internally all angles are in radians, but all IO is done in degrees.

3 EQUATIONS OF MOTION

3.1 Time

The independent variable in Parts II and III of this program is the particle angle θ. In these sections of the program an equation for τ, the RF time, is integrated along with the rest of the equations of motion. In Part I τ is the independent variable. The equation for τ uses the parameters (frequency error) and h (harmonic number), which define the RF frequency in terms of the orbital frequency, ν_0,

$$\nu_{rf} = h(1+\varepsilon)\nu_0 = h(1+\varepsilon)\,\omega_0/2\pi$$

The RF time τ is related to the dee locations by defining the energy gain at gap 1;

$$\text{FRAT} \equiv \text{h}(1+\varepsilon) = \nu_{rf}/\nu_0$$

$$\Delta E_1 = (q/e)V_o \sin(\tau) = FRAT \cdot \omega_0/2\pi$$

If dee unbalancing parameters have been included, the V_0 would be multiplied by the dee 1 scale factor, and the phase error would be added to τ. The phase of each dee is different from the preceding by $2\pi h/N_{dee}$; for instance, entering dee 2:

$$\Delta E_3 = (q/e)V_0 \sin(\tau - 2\pi h/N_{dee}).$$

A particle circulating at a frequency of ν_{RF}/h should cross the centreline of each dee just as the voltage in that dee goes through zero, to get the maximum energy gain per revolution. The phase value φ indicates whether this condition is satisfied. Phase error is given in RF degrees, and is positive when the RF leads the particle. Energy gain is approximately proportional to cos(φ). The phase value is always approximate in this program because the time centreline of the dees will not coincide with the geometrical centerline

3.2 Part 1

The equations used in Part I can be found using the normal equations for the Lorentz force,

$$\vec{F} = q\,(\vec{E} \times \vec{B}).$$

Using the usual substitutions,

$$\frac{d\vec{P}}{dt} = \vec{F}$$

$$\vec{v} = \frac{\vec{p}}{\gamma m_0}$$

we find,

$$\frac{d}{dt}p_x = qE_x + \frac{q}{\gamma m_0}\left(p_y B_z - p_z B_y\right)$$
$$\frac{d}{dt}p_y = qE_y + \frac{q}{\gamma m_0}\left(p_z x - p_x B_z\right)$$
$$\frac{d}{dt}p_z = qE_z + \frac{q}{\gamma m_0}\left(p_x B_y - p_y B_x\right)$$

Converting from time to RF time, and momentum units to length units is accomplished using,

$$\frac{d}{d\tau} = \frac{1}{\omega_0 FRAT}\frac{d}{dt}$$
$$p \rightarrow \frac{a}{m_0 c}p,$$

and the magnetic field is divided by the central field (BCON) to given the final equations,

$$p_x = c_1 E_x + c_2\left(p_y B_z - p_z B_y\right)$$
$$p_y = c_1 E_y + c_2\left(p_z x - p_x B_z\right)$$
$$p_z = c_1 E_z + c_2\left(p_x B_y - p_y B_x\right)$$
$$x' = c_2 p_x$$
$$y' = c_2 p_y$$
$$z' = c_2 p_z$$

where the constants are defined as,

$$c_1 = \frac{qa^2}{E0 \times FRAT}$$
$$c_2 = \frac{1}{\gamma\, FRAT}$$

The particle energy is computed in two different fashions. One method uses the particle's momentum, and it can be shown that the usual relation for kinetic energy can be put in the form,

$$E_{kin} = \frac{{p^2}/{a^2}}{1 + \sqrt{1 + {p^2}/{a^2}}}$$

where p is again the momentum in length units. Since $p^2 = {p^2}_x + {p^2}_y + {p^2}_z$ and p_x, p_y and p_z are obtained by solving the equations of motion, this means that we are using the calculated values of the electric fields to compute the energy. Since computing the electric fields involves taking derivatives of the potential, a small fluctuation in the potential data may have pronounced effect on the kinetic energy calculated from the above equation.

It is therefore desirable to calculate kinetic energy by an alternate method which uses only the potential data, and not the associated fields. The kinetic energy of a particle can be written as,

$$E = (\gamma - 1)m_0 c^2$$

so that,

$$\frac{dE}{dt} = \frac{d}{dt}(\gamma m_0 c^2) = \gamma^3 m_0 v \frac{dv}{dt}$$

in the last step,

$$\frac{d(\gamma)}{dt} = \gamma^3 \frac{v}{c^2}\frac{dv}{dt}$$

was used. We also have the equations of motion,

$$\frac{d(\gamma_0 \vec{v})}{dt} = q\left(\vec{E} + \vec{v} \times \vec{B}\right).$$

If we take the dot product of v and the above equation we obtain,

$$\vec{v} \cdot \frac{d(\gamma m_0)}{dt} = q\vec{v} \cdot \vec{E},$$

Manipulating the left hand side results in,

$$\vec{v} \cdot \frac{d(\gamma m_0)}{dt} = \gamma^3 m_0 v \frac{dv}{dt},$$

Hence combining the above equations, we obtain,

$$\frac{dE}{dt} = q\vec{v} \cdot \vec{E}$$

If the potential is given by V = V(x,y,z), then

$$\frac{dV}{dt} = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x}\frac{dx}{dt} + \frac{\partial V}{\partial y}\frac{dy}{dt}$$

$$= \frac{\partial V}{\partial t} - \vec{E} \cdot \vec{v}.$$

So we arrive at,

$$\frac{dE}{dt} = q\left(\frac{\partial V}{\partial t} - \frac{dV}{dt}\right),$$

or if we put all the total derivatives on the left hand side, and divide through by ω_{RF} we get the result:

$$\frac{dJ}{\partial \tau} = q\,\frac{V(x,y,\tau)}{\partial \tau}$$

where,

$$J = E + qV(x,y,\tau).$$

The J-equation is just another equation that is integrated along with the equations of motion. The kinetic energy as calculated by the J-equation and using the more direct method are both provided in the standard output.

3.3 Part II and Part III

The equations of motion used in CYCLONE in Parts II and III can be derived using a Hamiltonian with θ as the independent variable. The relativistically correct Hamiltonian for the case with general electric and magnetic fields is given by;

$$H = -r(p^2 - p_r^2 - p_z^2)^{1/2} - q'rA_\theta,$$

where $q' = q\,a'/m_0c$, and $\vec{A}$ is the magnetic vector potential. In this case the conjugate pairs are (r,P_r), (z,P_z), and (t,W). The following equations allow the conjugate momenta to be exchanged for more useable variables.

$$p_r = P_r - q'A_r$$

$$p_z = P_z - q'A_z$$

$$p^2 = \frac{1}{c^2}(W - q\Phi)^2 = (-1)m_0^2c^2 = 2m_0E\left(1 + \frac{E}{2m_0c^2}\right).$$

Hamilton's canonical equations then lead to,

$$r' = \frac{dr}{d\theta} = \frac{r\,p_r}{\sqrt{p^2 - p_r^2 - p_z^2}}$$

$$p_r = \frac{p_r}{d\theta} = \sqrt{p^2 - p_r^2 - p_z^2} + q'(rB_z - zB_\theta) + qt'\mathcal{E}_r$$

$$z' = \frac{dz}{d\theta} = \frac{r\,p_z}{\sqrt{p^2 - p_r^2 - p_z^2}}$$

$$p_z = \frac{p_z}{d\theta} = q'(r'B_\theta - B_r) + qt'\mathcal{E}_z$$

$$t' = \frac{dt}{d\theta} = \frac{\gamma\, m_0\, r}{\sqrt{p^2 - p_r^2 - p_z^2}}$$

$$\frac{dE}{d\theta} = q(r'\mathcal{E}_r + r\mathcal{E}_\theta + z'\mathcal{E}_z)$$

where; $\vec{B}$ is the magnetic field, and $\vec{\mathcal{E}}$ is the electric field. In part II we again use the J equation (see Part I), with;

Insert J equation from TRI-DN-02-??

Since the magnetic field is only given in the median plane, the field must be expanded in z to give the values off the median plane. The appropriate expansions to first order in $z$are;

$$B_z(r,\theta,z) = B_z(r,\theta),$$

$$B_r(r,\theta,z) = z\frac{\partial B_z}{\partial r},$$

$$B_\theta(r,\theta,z) = \frac{z}{r}\frac{\partial B_z}{\partial \theta},$$

3.4 Transit Time Correction

In Part III there is the option to add a transit time correction factor to the delta function approximation for the energy gain. The two parameters that control this are wgap(length units) and awgap(degrees). The correction is based on the usual formula for transit time,

$$\Delta E = \Delta E_0 \times \frac{\sin U}{U},$$

where $(\Delta E)_0$ is the normal delta function energy gain for the gap, and

$$U \equiv \frac{1}{2}\omega_{RF}\Delta t = \frac{1}{2}\left(\frac{d\tau}{d\theta}\right)\Delta\theta$$

From the above equations of motion we know that,

$$\frac{d\tau}{d\theta} = \frac{h(1+\varepsilon)}{m_0}\frac{dt}{d\theta} = h(1+\varepsilon)\gamma\frac{r}{p_\theta}$$

If the gap had a constant linear width then using simple geometry it can be shown that the angular width of the gap would be,

$$\Delta\theta = 2\,tan^{-1}\left(\frac{wgap}{2r}\sqrt{1+(\alpha_s r)^2}\right).$$

Combining this width with the constant theta term we arrive at a gap width of,

$$\Delta\theta = AWGAP + 2\,tan^{-1}\left(\frac{wgap}{2r}\sqrt{1+(\alpha_s r)^2}\right).$$

Note that if both wgap and awgap gap are used simultaneously then their association with the actual gap dimensions is lost and they should be viewed as two parameters that are adjusted to best fit the actual energy gain as a function of energy.

4 Electromagnetic & Gas Stripping

4.1 Lorentz Stripping

The second electron of H^- ion has a binding energy of 0.755eV, so it can easily be detached in a strong electric field. Of course relativistic motion in a magnetic field results in an electric field in the rest frame of the ion and the result is known as Lorentz stripping. The lifetime of the H^- ion in its own rest frame is of the form;

$$\tau(\varepsilon) = \frac{A_1(\varepsilon)}{E} e^{\frac{A_2(\varepsilon)}{E}},$$

where A_1 and A are only function of the binding energy ε. Experimental values are A_1=2.47 x 10^{-6} V s/m and A_2=4.49 x 10^9 V/m. The Lorentz transformation of the magnetic field in the lab frame to the electric in the rest frame of the ion is;

$$E[v/m] = \gamma\beta cB[T]$$

Where γ and β are the usual relativistic factors, c is the speed of light in m/s and B is the magnetic field in Tesla. The fraction of H^- surviving after a time t is given by;

$$f = e^{-t/\gamma\tau}$$

So the derivative with respect to θ is;

$$f' = \frac{-1}{\gamma\tau} f\, t'$$

The value of t' can be found above. If we renormalize as follows;

$$a_1 = \frac{A_1}{cB_0}, \qquad a_2 = \frac{A_2}{cB_0}, \qquad ef = \frac{E}{cB_0}$$

Then the time constant becomes,

$$\tau = \frac{a_1}{ef} e^{\frac{a_2}{ef}}$$

and,

$$f' = \frac{-1}{\omega_0 \tau} f \frac{r}{p_\theta}$$

4.2 Gas Stripping

Not yet implemented

5 Interpolation

Whenever interpolation is required (e.g. electric and magnetic fields etc.), a double three point

Lagrange interpolation is used. The procedure actually uses four points in each interpolation. For example, suppose V_1, V_2, V_3, V_4 are the function values at four equally spaced points x_1, x_2, x_3, x_4 respectively, and we wish to find the function value at $x_2 + f \cdot s$, here f is a fractional distance between x_2 and x_3, and s is the spacing between points. Then, using standard three point Lagrange interpolation, we fit a parabola to V_1, V_2, V_3, to find the value at $x_2 + f \cdot s$ is:

$$V_a(x_2 + f \cdot s) = \frac{f(f-1)}{2} V_1 + (1 - f^2)V_2 + \frac{f(f+1)}{2} V_3,$$

and it is easy to show that a three point Lagrange interpolation using V_2, V_3, V_4, to find the value at $x_2 + f \cdot s$:

$$V_b(x_2 + f \cdot s) = \frac{(1-f)(2-f)}{2} V_2 + f(2 - f)V_3 + \frac{f(f-1)}{2} V_4,$$

Now we take the weighted average between V_a and V_b

$$V(x_2 + f \cdot s) = (1 - f)\, V_a(x_2 + f \cdot s) + f V_b(x_2 + f \cdot s),$$

which can be written as,

$$V(x_2 + f \cdot s) = \sum_{i=1}^{4} c_i(f) V_i$$

where a little algebra shows that the $c_i(f)$ are given by,

$$c_1 = f^2 - 0.5(f^3 + f)$$

$$c_2 = 1.5f^3 - 2.5f^2 + 1$$

$$c_3 = 2f^2 - 1.5f^3 + 0.5f$$

$$c_4 = 0.5(f^3 - f^2)$$

The derivative of the function V can also be found in the same manner, where

$$\frac{\partial V}{\partial x} = V(x_2 + f \cdot s) = \sum_{i=1}^{4} c_{i+4}(f) V_i / s$$

where the new coefficients are,

$$c_5 = 2f - 1.5f^2 - 0.5$$

$$c_6 = 4.5f^2 - 0.5f$$

$$c_7 = 4f - 4.5f^2 + 0.5$$

$$c_8 = 1.5(f^2 - f)$$

The extension of this to two and three dimensions is simple. In two dimensions,

$$V(x_2 + f_x \cdot s, y_2 + f_y \cdot s) = \sum_{i=1}^{4}\sum_{j=1}^{4} c_i(f_x)c_j(f_y)V_{ij}$$

Hence, in two dimensions the interpolation involves 16 points and in three dimensions it involves 64 points.

6 Equilibrium Orbits

Equilibrium orbits are generated internally in manner similar to that used by the program GENSPEO, so that CYCLONE can compute x and p_x (displacements from the equilibrium orbit). The EO data can be stored in a file and read back in on future runs. The EO data is calculated at a fixed set of energies at the appropriate locations and then written to a file. When the data is in CYCLONE the radial values are stored as r^2 and the momentum values as p_r/p. During running, the program interpolates in the stored values as a function of energy, to arrive at an equilibrium orbit for the current energy. Then the values for x and p_x can be found. Several different angular locations are possible. In part III the EO data can be given at selected dee gaps (and centre lines) as can be done in SPRGAPZ. In part II the EO data can be given for a fixed angle or all angles in 1 sector. The basic EO results are written to the file EO.log for user reference.

7 Magnetic Field

A single magnetic field file is used by all three parts of CYCLONE. Basically this file contains an r-θ grid. The parameter HEADER tells CYCLONE how many lines to skip. A zero value indicates an ACSI file with no header (old standard), while a non-zero value indicates a file with a header. All files should be written using a format as given on the line following the file name (previous standard format was '(8F9.5)'), and with theta as the most rapidly varying.

8 Electric Fields

CYCLONE is setup to read the potential maps produced by the relaxation program RELAX3D. In Part II the electric field may be contained in multiple files to allow each dee to be stored separately and thus be independently phased. The routines for generating the input for RELAX3D generate a header file that accompanies the relaxation calculation. This file contains required data about the grid spacing and position that are required by CYCLONE. We have thus setup the command files to tie the two together using a run name and extension format. For each electric field CYCLONE will look for 2 files. They will be of the form file.EFLD and file.HEAD, where file is the file name entered on the line(s) following the EFIELD command (on unit 5). The file with the .EFLD extension is the RELAX3D output. The number of files is determined by the value of NTIME that is input. The position of the large and small electric fields relative to the magnetic field is determined by the data in the header files, and by the various shift and rotate commands. When using multiple files in Part II all files must have the same header data.

The number of vertical planes that are read in is determined by the value of NZ. If NZ=1 then the program assumes that only the median plane field has been given and then calculates the vertical electric field component using second derivatives of the potential. If only median plane motion is being studied a gain in speed can be made by setting NZ=1. This is not good for studying vertical motion, although it does make Part II fully linear in the vertical direction. If NZ is not equal to one and the particle exceeds the vertical size of the grid the particle is considered lost (just as when r=pr) and the program returns to reading input.

Since RELAX3D sets all fixed potential values to be negative we can check for collisions with the posts. If MAX HITS is non zero then the post checking is enabled. If a post point is used in the interpolation the absolute value of the potential is used to calculate the fields, and the hit count is

incremented. Generally since four points in each direction are used (i.e. 64 points for 3D maps), a single post point does not harm the calculation. In fact the calculation is correct if the post point is on the edge of the interpolation cube. The problem results if two neighboring points are both in the fixed potential, at this point the field calculation is bad. It is impossible to specify the exact conditions at which the calculation goes bad, but in 3D, 16 points can be inside a post and the results are still good. Therefore one often sets HITS to 17.

There are commands to rotate and shift the electric fields. The interaction of these is as follows;

$$x_m = r\,cos(\theta)\;\; y_m = r\,sin(\theta)$$

$$x' = x_m - x_{shift} - x_{rot}$$

$$y' = y_m - y_{shift} - y_{rot}$$

$$x_e = x'c_{11} + y'c_{12} + x_{rot}$$

$$y_e = x'c_{21} + y'c_{22} + y_{rot}$$

$$c = \begin{bmatrix} \cos(\theta_{rot}) & \sin(\theta_{rot}) \\ -\sin(\theta_{rot}) & \cos(\theta_{rot}) \end{bmatrix}$$

The 'm' subscript denotes magnetic coordinates and the 'e' subscript denotes the electric field coordinate, in other words the x and y values used to located the field in the grid file provided. R and θ are the orbit coordinates. See the shift and rotate commands for the other variables.

9 ROI

A *region of interest* (ROI) is a user defined window, that allows the user to modify the magnetic and electric fields in a given region. This is particularly suited to the addition of special elements in the cyclotron such as deflectors, magnetic channels, peelers etc., where the specifications of the elements are very user dependent. There can be any number of ROIs input.

The ROI specification is provided in a file given as a parameter to the ROI command. This file should contain NROI lines. On each line there should be the following values;

itype,var,theta1,theta2,r1,r2,nt

where itype is 1,2 or 3, var is a real number passed to the ROI subroutine, theta1 and theta2 define the initial and final angles of the ROI, r1 and r2 define the inner and outer radii of the ROI, and nt is the turn at which the roi starts.

The itype value specifies if the the ROI is a simple radial electric field (itype=1) with Er=var in kV/length, a simple additional magnetic field (itype=2) with Bz=BZ+var/BCON, or a user defined routine (itype=3).

For itype=3 the user defines the characteristics of the ROI in the user supplied subroutine 'Custom Roi'. The program library contains a dummy ROI routine which will be linked in should the user not provide his/her own routine. The subroutine argument list is;

Subroutine Custom Roi(theta,r,z,tau,nturn,iroi)

where, THETA,R,Z1 specify the current position of the particle, TIME is the proper RF time (τ), the number of the current ROI is IROI, and NTURN is the current turn number. The variables THETA and TIME are in radians, while R and Z1 are in units of length (inches, cm, etc.). The sample file indicates which variables are available for the user to modify in the ROI routine.

If you are using the distributed CYCLONE script then typing "cyclone make my file" where your code is in the file "my file.f90" will generate a cyclone.exe file in the current directory. Subsequent executions of the CYCLONE script will run the local cyclone.exe file (if it is present).

10 Gap Factors

Voltage variation as a function of radius can be accommodated in part3 using a gap factor table. The gap factor table is located in a file specified by the GAPFACTOR command. The first line of this file should contain;

npoints,first point,delta

where NPOINTS is the number of points in the file, FIRST POINT is the radius of the first point and DELTA is the radial increment. The second line should be a valid format for reading the NPPOINT values eg. "(F10.5)" which would read one value per line. (Format should be less than or equal to 20 characters.)

Gap factors should be normalized to 1 since the accelerating voltage will be given by CHG*VOLTAGE*gapfact*cos(φ). The derivative of the gap factor is used to compute the momentum compaction. Setting the 4th debug variable on will give a printout of the various components of the energy gain at the gap.

A voltage that varies will give a phase compression [6]. This comes about from a magnetic field in the gap that adds an additional momentum kick to the particle. The kick is given by;

$$\partial p_r = con \times \frac{d}{dr} \times \sin\varphi$$

with;

$$con = V_{dee} \times a^2/EO$$

and the energy gain at the gap is given by;

$$\Delta V = V_{dee} \times \cos\varphi$$

11 Files

The program manipulates data contained in many different files, most of which are optional. These files are briefly listed below.

Electric Fields(45 & 46) There can be many different electric field files involved in a given CYCLONE run. Details of these files are given in section 8.

Magnetic Field(44) There is one magnetic field used in any given CYCLONE run; details of the file structure are given in section 7.

Data Base(34) This is an output file that contains the basic run input data and file names. This file is used by various macros for post-processing the CYCLONE output.

EO(26) One file containing equilibrium orbit data can be saved and read back in. The nature of this file is discussed in section 6.

Orbit Data(35) This is the principle output file other that the screen data on unit 6. This is a binary (or formatted) file with the orbit coordinates stored with one record per step. This file is correctly maintained so that it proceeds monotonically in the orbit direction. This file is used by macros to plot the orbits.

Centre of curvature (36) data is written if ILOG36 is set. Each line contains: nturn,ip,th,radius,pr,e,rc,psi

Collimator (10) Particle position at inflector collimator (nrun,xc,yc,xp,yp)

Theta Data (16) This is an ASCI file that contains the orbit coordinates at the fixed theta printout locations (parts II & III only). This file uses unit 16 and is controlled by ILOG16. The format mirrors that found on unit 6.

Gap Data (47) This is an ASCI file that contains the orbit coordinates at the gap crossings (part III

only). This file uses unit 47 and is controlled by ILOG47.

Part2 Efield Output Actual values of the electric fields used along the orbit in part 2 are output on unit 50 if ILOG50 is non-zero. This file has one record per integration step. Values output are;

$$Nturn,\ \theta,\ E,\ \tau,\ E_r,\ E_\theta,\ E_z$$

Part1 Efield Output Actual values of the electric fields used along the orbit in part 1 are output on unit 51 if ILOG51 is non-zero. This file has one record per integration step. Values output are;

$$Nturn,\ \theta,\ R,\ z,\ E\ ,\tau,\ V,\ V\ sin(t),\ dV/dT,\ E_x,\ E_y\ E_z$$

Input Data (5) The basic run parameters are read in from unit 5. The information that is required on this unit is described in section 11.

End Code (15) Contains the condition that stopped each particle run.

Final Conditions (17) This ASCI file contains one line for each particle that successfully reaches the desired turn number. The line contains the same information as the printout on unit 6. This file uses unit 17.

Wall Data (18) This ASCI file contains the accumulated Lorentz stripping losses on the wall. This file is written when the LORENTZ,PRINT command is read. It contains three columns, θ, Loss, Power. Loss is the total % loss in that angular location, and Power is E*%. To obtain actual power in watts in the theta patch, multiply the number in this column by 10 times the beam current in mA.

Stripped Conditions(56) This ASCI file contains 2 lines per particle. The first line is the coordinates at the stripper (θ, R, Pr, Tau, E, Z, Pz) and the second is the values at the cross over point (θ, R, Pr, P_θ, E, Z, Pz, Matrix), where the matrix is the transfer matrix values if this option is enabled.

Axial_all(53) Very detailed E field data from part zero

Foil Data(56)

Cross-Over Data(57)

Restart Data(58) In stripper runs the restart point is stored on this file, in a format that can be used as input to a future CYCLONE run.

Foil Frame(59)

eAxial (61) Part Zero E field (`efld,amp(efld),test`)

bAxial(81) Part Zero magnetic field data (Tau, x, y, z, Bx, By, Bz)

Lost (85) Record of all particles lost in parts 0, I or II and location.

Spiral Data This file is required if the DEE command specifies an input file. The first line of the file is a header that specifies;

$$npoint,\ r0gap,\ drgap,\ nsets$$

which are the number of gap data points to follow, the radius of the initial gap point and the radial spacing of the points. The next npoint lines consist of 2*nsets angles per line. The first angle is the angle of the first dee entrance at that radius, and the second the angle of the first dee exit. The other spiral lines are found by symmetry. The program then assumes a linear spiral between pairs of points given in the gap table. npoint must be at greater than or equal to 2.

12 INPUT to CYCLONE VERS 9

CYCLONE Version 9 uses a command driven input structure. The commands are shown in tables 1 and 2 with their parameters. Commands and parameters can be separated by spaces, commas, an equal sign or blanks. Multiple commands can exist on the same input line, but parameters of a command cannot be split across lines. Command names should be given in capital letters. Usually commands can be abbreviated, however at least 4 characters maybe required to ensure command uniqueness (unless the full command itself is less than 4 characters). An exclamation mark can be used to indicate a comment line.

The RUN command starts the computation, so both a BFIELD and a PARTICLE command must be given before the first RUN command. Normally the order in which commands appear before a RUN is executed is not important. However in the special case of an ELLIPSE command, if R or PR values follow the ELLIPSE command then the new values are used to offset the orbits from the EO, while values of R or PR input before the ELLIPSE command are used as initial guess for the EO search. Each time a command is given, the new parameter values replace those previously given. Normally blank lines are ignored. The CONTINUE command can be used to overwrite the starting particle parameters with the final values of the preceding particle run.

The BFIELD, EODATA, BUMP, CONE, ROI and EFIELD commands all read file names from the next line of the input file. The DEE command may also read a file name. When this information is required there must not be a blank line or comment following the command.

Data types of the parameters given in parenthesis after each parameter name. Valid types are real (R), integer (I), character (C) and logical (L). Logical values can be specified using TRUE/FALSE, YES/NO, or ON/OFF. Character parameters should be specified in uppercase. Parameter order is important. In many cases parameters that are not specified will default to reasonable values.

Command	Parameters	Description
ACCEL	ispr on	disables gaps
BFIELD**	nsc ntheta nr r0 Δr bcon bump	Reads in magnetic field
BOUNDS	1/2 bnd	true/false condition for bnd_1 or bnd_2
BUMP**	bump amp,bump ang,harm	Input harmonic field
CENTRE	R,Pr,Z,Pz	Offset coordinates for an ellipse
COLLIMATOR	EXISTS z x y theta	Inflector collimator
CONE**	cone amp	Input cone field
CONTINUE		start from final values of previous orbit
DEBUG	Idebug, Enable	debug numbers 1 thru 6
DEE	ndee file dee width first gap spiral angle	
ELLIPSE	X/Z/Both ε_1 (ε_2) normalized fill npart	Generate Eigen-ellipse
EFIELD*(ntime)	1/2 scale nz ntime	Reads in electric fields
ERROR	idee verr(idee) perr(idee)	
END		End of job
EODATA*	NEO EOI DEO THEO IEO NTHEO	Reads in EO data
EPS	Eps	mutually exclusive with FREQUENCY
FILE*		Start reading input from a file
FLAG	Active, Theta, Rmin, Rmax	Stop particle on flag
FOIL	Active, Cross, Energy, Opt, Hold, Position, Transfer,Size	Controls stripping foil behaviour
FREQUENCY	RF frequency	mutually exclusive with EPS
GAPFACTOR*	Iharm	voltage variation with radius
HARMONIC	Harmonic	harmonic number
INFLECTOR	zint, th_match, r_match	Part zero setup
ISOC	r_isoch HoldFreq Write	Isochronize magnetic field
LOG	ilog on/off formatted(true/false) spaces(i)	controls log files ilog=35/36/47/50
LORENTZ	ON/OFF/WALL, Rwall, Print	Control Lorentz striping calculation
NOCENTRE		Turn off the ellipse centre
NOTIME	no time time notime	
OFFSET	1/2 delta v vscale	E field positioning
PART	E0 CHG	This line must exist
POSTS	0/1/2 max hits no posts allowed post log	Post checking in electric fields
PRINT	0/1/2/3/4 np th0 dth, lsect, nsect	Fixed angle print definition (4=spiral)
PULLER	H_{puller}, epull	
ROI*	epull(true/false) nroi	Region of interest

ROTATE	1/2 theta xc yc	rotate small(1) or large(2) field
RUN	nturn kcy	Starts orbit
SHIFT	1/2 x y	move small(1) or large(2) field
START	kcy(0/1/2/3)	Initial Part
THIRD	idee verr3(idee) pherr3(idee)	enables third harmonic
TIME	wgap awgap	gap correction factors
TRANSFER	0/1/2/3 n θ	turn number and angle for transfer between parts
TSTEP	istep	Part 1 tau step size(negative for < 1)
TURNS	Nturn	Maximum turn number
VARYTAU	Enb(L), TauMin, TauRange	Vary Tau for ellipse cases
VOLTAGE	Voltage	dee voltage in kV
UNITS	length unit, field unit	(cm, mm, m, inch) (kG, T, mT)

Table 1: Input commands for CYCLONE. In the parameters column the "/" indicates "or", e.g. 1/2 means 1 or 2. An * after a command indicates that it will read one or more lines from the input file following the command's line.

Command	**Parameters**	**Description**
THETA	Θ, isw	starting angle and switch
R	R	starting r
PR	Pr	starting p_r
Z	Z	starting z
PZ	Pz	starting p_z
PHI	Phi	starting φ
TAU	Tau	starting τ
ENERGY	E	starting energy
ALPHA	α	starting α
ETA	η	starting η
XI	ξ	starting ξ
PETA	P_η	Starting py
PXI	P_ξ	Starting py

Table 2: Orbit coordinate inputs for CYCLONE.

ACCEL

ispr (I) Spiral line number of the gap. Accelerating gaps will have numbers 3,5,7... 2*ndee+1 with 3 and 7 being dee entrances and 5 and 9 being dee exits.

enable (L) Enable or disable voltage gain at the specified gap.

BAXIAL

th_rot (R) The angle to rotate the electric field in degrees.

The line following the EAXIAL command is assumed to contain the file name without file extension.

BFIELD

nsc (I) The number of sectors. The magnetic field is assumed to cover the angular range $0 \le \theta < 360/nsc$.

ntheta (I) The number of theta values in the magnetic field

nr (I) The number of radius values in the magnetic field.

r0 (R) The radius of the first value in the magnetic field.

Δr (R) The spacing between radius values in the magnetic field.

BCON (R) The isochronous field value in kG.

Header (I) Number of header lines to skip in field file. If zero then there is no file header. If non-zero then there is a standard ASCI header on the file that is terminated with an EOF. The program will read up to Header lines looking for the EOF character. (Note: Originally this was a logical value, with False being the same as zero, and True the same as 50.)

Normally the line following the BFIELD command is assumed to contain the file name. There are two special names; BFLAT and GAMMA that cause a field to be generated internally by CYCLONE. In these two cases no more lines are read by the BFIELD command. Following the file name is a format specification for the data. It should be a normal FORTRAN format with the opening and closing parentheses e.g. (8F9.5).

BOUNDS

itype (C) Indicate the SMALL or LARGE electric field.

bnd (L) A switch to allow running off the edge of the small electric field file. When false the particle is allowed to run outside the electric field file until the alternate stop condition is reached.

BUMP

Bamp (R) Amplitude of the field will be used to scale the input data.

Bang (R) Angle of the harmonic field (deg).

Harm(I) Harmonic of field (e.g. 1 or 2).

The following tw0 input lines must contain the file name and the file format information.

CENTRE

R (R) Radial or x coordinate of ellipse centre

Pr (R) Pr or px coordinate of ellipse centre

Z (R) Vertical or y coordinate of ellipse centre

Pz (R) Pz or py coordinate of ellipse centre

COLLIMATOR

Exists (L) Turn collimator checking on/off

zCol (R) Vertical location of the collimator

xCol (R) Half Width of aperture in x' direction

yCol (R) Half Width of aperture in y' direction

thCol (R) Orientation of aperture. This is the angle between the x axis and the collimator x' axis. Positive angles are CCW from x axis.

CONE

Bscale (R) Amplitude of the field. This will be used to scale the input data.

The following two input lines must contain the file name and the file format information.

CONTINUE Set the initial particle coordinates equal to the final particle coordinates of the preceding run. Only valid after at least one RUN command has been executed.

DEE

file (L) If true then gap data is read from a file, in which case the next input line must specify the file name, and no other parameters are required for the DEE command.

Δ_{dee} (R) The angular width of the dees. The second gap crossing (the exit of the first dee) is given by $\theta_2 = \theta_1 + \Delta_{dee}$.

ndee (I) The number of dees (max = 4). The dees are evenly spaced around the machine, so that the second dee entrance is given by, $\theta_3 = \theta_1 + 360/ndee$

θ_{g1} (R) The angle of the first gap at r=0. The first spiral is given by $\theta_1 = \theta_{g1} + \alpha_s * r$. This is the gap at which the turn counter is updated in Part III. This must be an entrance gap.

α_s (R) The spiral constant in degrees per unit length.

DEBUG

idbg (I) The debug type, which is an integer between 1 and 8

State(L) On/Off status of the indicated debug type

1- Initial Conditions
2- Transfer and setup values
3- Part 1 details and stripping
4- Part 3 details including delta E
5- Part 1 RRK data
6- Ellipse data
7- Input parsing
8- Magnetic field information

EAXIAL

inf_voltage (R) The inflector voltage in kV.

th_rot (R) The angle to rotate the electric field in degrees.

The line following the EAXIAL command is assumed to contain the file name without file extension.

ERROR

idee (I) The number of the dee for which error information is being given (should be between 1 and ndee).

verr (R) The relative voltage error of the dee (Vdee = voltage*verr).

perr (R) The phase error of the dee in degrees (should be between 180 and -180 degrees).

EFIELD

itype (C) Indicates the SMALL or LARGE electric field.

scale (R) A scale factor on the dimensions in the grid's header file

nz (I) The number of vertical planes to be used.

ntime (I) The number of independent dees. The program will expect a RELAX3D file for each independent dee. Not used if itype=small.

The line following this command must contain the electric field file name(s) without extension (ntime lines).

Ellipse

Populate an ellipse with particles. The definition of the ellipse shape and the type of fill is determined by the parameters chosen.

Examples:

ELLIPSE,XE,ε_x,T,EDGE,9
ELLIPSE,BOTHE,ε_x,ε_z,F,EDGE,9,T
ELLIPSE,ZI,S33,S44,S34,GAUSS,900
ELLIPSE,BOTHI,S11,S22,S12,S33,S44,S34,EDGE,9
ELLIPSE,CORR,S13,S14,S23,S24,FILL,nPart

type(C)

XE The r-pr eigen ellipse is used
ZE The z-pz eigen ellipse is used
BOTHE Both eigen ellipses are used
XI Sigma values are input to define r-pr ellipse
ZI Sigma values are input to define z-pz ellipse
BOTHI Sigma values are input to define both ellipses
CORR Sigma correlation values between the two ellipses are being defined. Only valid if BOTHI proceeds this.

If type is eigen then the following are read;

ε(R) Ellipse area in mm-mrad. (Note: if type=BOTH then two emittance values are expected)

normalized(L) Determines if the emittance is normalized or un-normalized.

Else if sigma input

Sigma(R) Sigma matrix values σ_{11}, σ_{22}, σ_{12}, σ_{33}, σ_{44}, σ_{34} in that order. If x or z type then input only those 3 values that apply.

Else if CORR

Sigma(R) Sigma matrix values σ_{13}, σ_{14}, σ_{23}, σ_{24} in that order.

And this is followed by;

Fill(C) Type of particle distribution to be used. This is one of EDGE, GAUSSIAN, or UNIFORM

nPart(I) Number of particles in the distribution. The default is 9 which for edge fill gives the traditional 8 articles on the boundary of the ellipse. For edge fill, other values will place nPart-1 particles around the ellipse spaced equally in angle. For Edge and Uniform when both is selected,

then nPart refers to the number in each plane. For these cases there are nPart values in each plane that are combined in all combinations to create $nPart^2$ rays. For uniform type it is best to select a value of nPart that is the square of an integer.

The values of R and PR at the time of the ELLIPSE command are used as initial guess for the EO search. At present if eigen is selected then the command only generates coordinates at zero degrees. The continue command can be used to force the end conditions of a run to be used to set the energy and R,Pr guess. If either R or Pr are set after the ellipse command then these values are used as the ellipse centroid. If an EODATA,START command is related to this run, then the ellipse is centre on the EO, plus any x,px offset applied in that command.

Bound(R) Truncates a Gaussian distribution at a given number of sigma (remember the number of dimensions affects the meaning). If set to zero (default) there is no bound.

END Same as an EOF marker. Input stops at this point and all subsequent lines are ignored, does not execute a run.

EODATA

READ Read EO data from a previously created file. The file name must be on the next line following the EODATA, READ command.

WRITE Write EO data to a file. The file name must be on the next line following the EODATA,WRITE command. This file can then be input in future runs using the read sub-command above. (Note: EO data is scaled to improve interpolation.)

MAKE Generate the EOs. This is followed by a number of variables.

Neo (I) The number of equilibrium orbits in the file.

Eoi (R) The energy of the first EO value.

Deo (R) The spacing between EO values.

rguess (R) An initial guess for the radius of EO number 1.

prguess (R) An initial guess for the pr value for EO number 1.

ieo type (I) An indicator specifying the x-px calculation location (between 2 & 5). If ieo type = 2 the calculation is done at a fixed theta value ($\theta = theo$), and only in Part II. If ieo type = 3 then the x-px calculation is done at the dee crossings just as is done in SPRGAPZ. For ieo type=4 then x-px calculations are done at all print angles. Ieo_type=5 will only output the EO data at R(0) and should not be used for computing x-px information. Default value is 3.

theo (R) The angle that the EO data is given at. (For type 2 only.)

START Start the next ray on the EO at theta=0 deg.

x Radial offset from the EO

px Radial Momentum offset from the EO

ERROR Per dee voltage difference

ERROR3 Per dee voltage difference for third harmonic

EPS (R) The frequency error, i.e. $\omega_{rf} = h\,(1+EPS)\,\omega_{orb}$.

FILE Switch to reading commands from a file (mostly for MATLAB implementation). File name must be located as the next line in the input and will be the last item read prior to opening the specified file.

FLAG

Active (L) Turn flag on/off

Theta (R) Angle for the flag

Rmin (R) Minimum radius of the flag

Rmax (R) Minimum radius of the flag

Particle is stopped if the radius falls between Rmin and Rmax at the specified radius

if the flag is active

FOIL

ACTIVE Optimize the stripper location

FoilActive (L) Activate/Deactivate the stripper foil

nStripperSteps (I) Number of steps to integrate after the stripper.

StrippedStepSize (R) Step size for integration after the stripper, units is path length.

CROSS Set the cross over position

CrossOverRadius (R) Location of the cross over point, will be used optimize foil when "Cross" type optimization is selected. Also orbit values will be interpolated at these values.

CrossOverTheta (R) Angluar position of the cross over point (see notes above).

ENERGY Set the foil energy parameters

FoilEnergy (R) The desired stripping energy (central ray)

FoilDeltaE (R) Energy loss in the foil in MeV, the default is 0.0

StrippedCharge(R) The particle's charge after stripping. The default is 1.00109 which is appropriate for H^- to proton. (The normal orbit charge is usually positive, which is solely a convention. In the case of the stripped charge positive is the opposite of the circulating charge.)

OPT Optimize the stripper location

FoilType (C) Selects the type of optimization of the stripper location

FIXED Position fixed by position setting, no optimization

CROSS Vary angle to achieve cross over point

EFIXED Strip at the defined energy, keeping the angle fixed.

EOPT Strip at the defined energy and optimize the angle to achieve the crossover

MaxIter (I) Maximum number of iterations to search for the optimization

Tolerance (R) Accuracy of the achieved search value (e.g. cross-over point)

HOLD Hold the previously obtained stripper location. Useful if multiple particles are being run and the first ray is a position optimization.

nStripperSteps (I) Number of step to integrate after the foil

Tolerance (R) Accuracy of the cross-over position to obtain

POSITION Define the foil location, and activate extraction by stripping

FoilRadius (R) Radial location of the foil (initial position if fitting).

FoilTheta (R) Angular position of the foil (initial value if fitting).

FoilTilt (R) Angle that the foil forms with a radial line. Foil is angled from the defined tip position. Positive angle means larger radial portions of the foil are downstream of the tip.

TRANSFER Turn on transfer matrix calculations

TransferMatrices (L) Enable/Disable transfer matrix calculations.

LogTransfer (L) Create a log file containing the transfer matrix elements from stripper to cross over point.

SIZE Trap particles outside the foil size

CheckSize (L) Enable/Disable checking the foil size

FoilWidth (R) Foil width, distance in r from the foil position that is okay

FoilHeight (R) Maximum vertical value to remain in foil

FREQUENCY (R) The RF frequency in Mhz. Need only be specified if there is not an exact integer relationship between the orbital frequency and the RF frequency. This is mutually exclusive with EPS.

GAPFACTOR

iharm(I) Specifies the harmonic number associated with the table. Valid values are -3,-1,1,3 where the negative values are used to disable the gap factors.

The line immediately following this command must contain the file name of the gap factor file.

HARMONIC (I) The harmonic number (i.e. the integer ratio between the orbital frequency and the RF frequency).

INFLECTOR

zint (R) The initial z value at which to start tracking in part zero. If positive tracking will start down, while if negative tracking will start going up.

th_match (R) The angle to end part zero.

r_match (R) The radius to start checking for th_match – will only check once this radius is exceeded.

ISOC

R_isoch (R) Maximum radius to isochonize, and match field at this radius

HoldFeq (L) If true then maintain B0, else adjust B0 to match R_isoch, default is false.

Save (L) If true then write the resulting field to file. If true then the file name must follow on the next line.

LOG

ilog (I) Unit number of an output file. Valid values are 34,36,47,50, and 51.

state (L) Whether the unit ILOG is on (writing) or off.

formatted (L) Whether the unit is binary (unformatted=F) or ASCI (formatted=T). Default depends on the unit number.

spaces (I) Number of blank lines to leave between runs. A value of -1 indicates EOF is written.

LORENTZ Calculation of Lorentz stripping – only part 3

TYPE (C)

ON Enable calculation of Lorentz stripping. Will print out fraction remain on unit 6. This will also print z motion on unit 6.

OFF Disable calculation of Lorentz stripping.

Wall Enable summing of looses on a surface located at Rwall. This command also resets the accumulated losses

Rwall (R) Radius of the loss surface – losses are projected straight forward from each integration step to this radius.

PRINT Write the accumulated losses on unit 18 now (as soon as command is processed)

NOTIME

notime (L) If notime is true, then the time dependence of the voltage will be turned off in Part I (ie. a DC field). At all steps the voltage will be equal to $voltage * vscale * sin(time)$.

time (R) The value of the RF time used if notime is true.

OFFSET

itype (C) Indicate the SMALL or LARGE electric field.

ΔV_1 **(R)** An offset to the input potential (RELAX3D file). This value is subtracted from the potential value found in the electric field file. That is to say the V = abs(V_{relax}) - ΔV_1.

vscale (R) Only valid for the SMALL field. It gives a scale factor on the voltage.

PARTICLE

E0 (R) The particle mass in MeV.

q (R) The particle charge in units of e. May be fractional. (sign???)

POSTS

itype (C) Indicate the AXIAL, SMALL or LARGE electric field.

max hits (I) The number of times a point inside a post (fixed potential region) can be used during the field calculation at any one point. If HITS is zero then the post checking is disabled. When working in just the median plane an appropriate value is 5.

no posts allowed (L) If true then stop particle when max hits is reached.

post log (L) If false then no printout is provided until the particle is stopped, otherwise all post HITS are logged on unit 6

PRINT

iprt (I) Integer between 1 and 4 indicating what printing information is being specified. Values of 0, 1, 2 or 3 indicate that the specification is for normal integral steps in parts 0, I, II or II respectively. For iprt=1 only the parameter np is used. For iprt=4 then the specification is for prints on spiral lines in part III.

np (I) Number of prints per turn. (In part Iit is the number of steps per print.)

th0 (R) Angle (or gap number) of first print on each turn.

dth (R) Step size in degrees (or gaps) between prints.

lsect (L) Make prints modulo nsect

nsect (I) When repeating each sector this defines the sector (can be different from magnetic field)

For iprt=0

nprint (I) Number of tau steps between prints.

nstep (I) Tau steps size in degrees. If less then 0 then step size is 1/nstep.

max_step (i) Maximum steps to run before stopping if th_match not found

PULLER

ηpuller (R) The η value at which the puller is located. Part I will interpolate all integration and printout variables at this location, and the integration will proceed from this point.

epull (L) If epull true, then the electric field will be turned off at the puller ($\eta > \eta_{puller}$) and the integration will proceed to θ_{pt2}, whether or not θ_{pt2} lies inside the small electric field.

ROI Implements region of interests as specified in a file.

NROI(I) Number of ROI regions.

The line following this command must contain the ROI file name.

ROTATE

itype (C) Indicates the SMALL or LARGE electric field.

θ **(R)** Angle of rotation of small field (positive is ccw)

$\mathbf{xc_{small}}$ **(R)** X position of centre of rotation for small field

$\mathbf{yc_{small}}$ **(R)** Y position of centre of rotation for small field

RUN Orbit calculations begin immediately after this command is given.

SHIFT – shift the electric field relative to the magnetic field. Shift is applied before rotate.

itype (C) Indicates the SMALL or LARGE electric field.

xsmall (R) Move small field in x direction.

ysmall (R) Move small field in y direction.

START (I) Specifies whether to start in part 1,2 or 3.

THIRD

idee (I) The number of the dee for which the third harmonic information is being given (should be between 1 and ndee).

verr (R) The relative voltage error of the dee (Vdee = voltage*verr).

perr (R) The phase error of the dee 3rd harmonic in degrees (should be between 180 and -180 degrees).

TIME

wgap The width the dee gap (in length units) - used for transit time correction.

awgap The angular width of the dee gap (in degrees)- used for transit time correction.

TRANSFER This command sets up the various transfer points between the various Parts of CYCLONE. Since the default is to transfer to the next part when a particle goes off of an electric field file, these transfer points will only affect the actual transfer if they occur before the particle goes off the field or the default is overridden with the BOUNDS command. There are four possible transfer points; 0 = before Part I, 1 = between Part I and Part II, 2 = between Part II and Part III, and 3 = after Part 3.

itype (I) Indicates which transfer is being specified (0,1,2,or 3).

nt (I) Turn number of the transfer. In part I nt is the number of turns in part 1, which is important when tracking backwards (i.e. for itype=0, nt=0 is the most common entry).

theta (R) Angle of the transfer. This should be a valid printing angle in Parts II or III. In Part 1 the orbit is interpolated at this point. In Part I the orbit need not enclose the machine centre so some angles may not traversed.

stop (L) Determines if the orbit should be stopped at the transfer point. This is only valid if itype is 0, 1 or 2.

TSTEP

tstep (I) Adjusts the Tau step size in part one. If TSTEP is positive then the step size in degrees is TSTEP*NH. If TSTEP is negative then the step size is NH/TSTEP.

TURNS (I) Number of turns before stopping the run. The turn number is incremented at zero degrees in Part II and at the first gap crossing in Part III. A negative value for turn indicates a backwards run.

VARYTAU to be used with the ellipse command otherwise has no effect

Enable(L) – Enable random tau values when running ellipse cases

TauMin(R) – Minimum value of tau (degrees)

TauRange(R) – Range of tau values in degrees. Tau will be uniformly distributed between TauMin and TauMin+TauRange

VOLTAGE (R) The dee voltage in kV.

UNITS

length unit (C) Valid entries are; 'CM','MM','METER', or 'INCH'

field unit (C) Valid entries are; 'KG','TESLA','MTESLA', or 'MILLITESLA'

If starting in PART 1 then the coordinates may be specified as either (THETA,R,PR) or (XI,ETA,ALPHA), however you cannot mix the two coordinate systems. CYCLONE tries to use the set of coordinates associated with the last occurrence of one of these six commands. Using (XI,ETA,ALPHA) when starting in parts 2 or 3 makes no sense. In Part I the turn number is incremented at the starting angle. In Part II the turn number is incremented at zero degrees, while in Part III the turn number is incremented at the entrance to the first dee.

The following commands give the beam coordinates and only have a single parameter:

R The initial radius in length units.

PR The initial value of the radial momentum in length units.

Z The initial vertical position in length units.

PZ The initial value of the vertical momentum in length units.

ENERGY The starting value of the energy in MeV.

TAU The starting time in degrees. The last value of PHI or TAU before the RUN determines the actual initial condition used.

PHI The starting phase in degrees. The last value of PHI or TAU before the RUN determines the actual initial condition used.

XI ξ is the x coordinate of the particle in the small electric field system. Should only be used when starting in part 1.

PXI is the momentum in the ξ (x) direction (part zero)

ETA η is the y coordinate of the particle in the small electric field system. Should only be used when starting in part 1.

PETA is the momentum in the η (y) direction (part zero)

ALPHA α is the angle between the initial particle direction and the η axis, for starting in Part I. A positive value of α is a clockwise rotation of the trajectory. Should only be used when starting in part 1.

THETA θ is the starting angle in the magnetic field coordinate system. A second parameter following the angle can be either ANGLE or GAP. The gap parameter only has meaning in PART III where it allows starting on a gap. In this case θ should be a half integer. (i.e. .5,1.,1.5,2 .. 2*ndee).

13 EO, Ellipses & Starting Conditions

The ellipse command can be used to generate a set of starting conditions based on an eigen-ellipse. The number of particles and the distribution type is determined by the user selections on the ellipse command line.

If the continue command is issued following the ellipse command then the results of the last run are used to position the ellipse. (Caution to ensure that the previous run ended at zero degrees.)

If the EOData,Start combination is used then the orbit begins on the EO, at zero degrees. Offsets for x and px, can be added to this command.

As of version 10.12 updating the position coordinates no longer sets the ellipse centre. Use the CENTRE command instead.

14 Contents of dbs (unit 34) file

All Orbit runs

- Magnetic field file name
- Δr, r0, nr, ntheta, bcon, E0

First orbit run only

- 1 line with file name of the small electric field (='none' is no field read)
- 1 line with ntime, scale1, scale, lines of offset= 5, lines per run = 13
- Ntime lines with the file names for the electric fields

All Orbit runs

- 10 lines that mimic the old card data entry

```
0,x_small,y_small,x_rot_small,y_rot_small,th_rot_small,rhit1,
dv_small,rbnd1
1,thpt2,etapul,epull,float(nt),f50,f51,conv,acon
2,theta_in,float(ith_switch),rin,prin,zin,pzin,0.,0.
3,tau_in,e,pot,time,t2,theta_pt2,f47,f36
4,epsi,float(nh),voltage,vscale,rout(notime),t3,theta_pt3,0.0
5,gap1*tcon,as,dd*tcon,float(ndee),0.0,wgap,agap
6,Foilactive,Ftype,XC_rad,XC_th,F_rad,F_th,F_tilt,F_E
7,float(nprnt2),tt2,td2,float(nprnt3),tt3,td3,float(nump),f35
8,x_large,y_large,x_rot_large,y_rot_large,
th_rot_large,rhit2,dv_large,rbnd2
9,turns,float(kcy),rout(debug1),
```

- 1 line with line counts for files 35,36,50,51

15 File Naming

The assignment of IO units to specific file names is done using a data file. This file must be named "extension.dat" and contains the unit number followed by the file name extension. The extension supplied is appended to the run name which in the MatLab version is supplied as a parameter and in DOS is entered on the command line. The extensions.dat file is first looked for in the current running folder. If not found then the program looks for the environment variable "Cyclone$dir" which should contain a path to the folder that contains the extensions.dat file. If no file is found then the default FORTRAN file names (FORT.nn) are used.

Sample contents of an extensions.dat file follow;

```
6,.log
```

```
8,_sposts.log
9,_lposts.log
10,_collimator.log
14,_inf.dat
15,_ecode.log
16,_TH.log
17,_end.log
18,_gob.log
19,.rlog
25,_etest.dat
26,.EO
35,.orb
34,.dbs
36,_cent.log
45,_strip.log
47,_gap.log
50,.fld
51,.fld1
52,_ell.log
53,_axial_all.fld
55,_flag.log
56,_foil.log
57,_XC.log
58,_fstart.dat
59,_frame.log
61,_ISO.log
62,_eAxial.fld
67,_test.dat
77,_f77.log
80,_bAxial.fld
85,_lost.dat
```

References

[1] T.I. Arnette, “program CYCLONE”, Michigan State University internal report (1966).

[2] ”Revised CYCLONE Program for Central Region Orbit Calculations”, Michigan State University Cyclotron Lab. internal report (1978).

[3] C.J. Kost and F.W. Jones, “RELAX3D User’s Guide and Reference Manual”, TRI-CD-88-01, May 1988.

[4] J.L. Chuma, “PHYSICA Reference Manual”, TRI-CD-93-01, v1.3 (Jan 1998).

[5] S. Gill, “A Process for Step-by-Step Integration of Differential Equations in an Automatic Digital Computing Machine”, Proc. Cambridge Philos. Soc., 47, 96(1951).

[6] W. Joho, “Application of the Phase Compression - Expansion Effect for Isochronous Storage Rings”, Particle Accelerators, 6, 41 (1974).