Introduction:

PARMELA is an electron linac counterpart of PARMILA, the name being derived from the phrase, "Phase and Radial Motion in Electron Linacs." Like PARMILA, it is a versatile multi-particle code in which the beam, represented by a collection of particles, may be transformed through a linac and/or transport system specified by the user.

Although PARMELA has the same structure as PARMILA, there are two basic differences between the two codes:

1. In electron linacs, the velocity of an electron can change significantly in the first few cells. Consequently, the particle motion must be integrated through these cells rather than being transformed by a gap transformation.

2. In order to treat the space-charge forces properly, PARMELA uses time (or phase angle) as the independent variable. Unfortunately, this complicates the logic of the code, as well as the input and output processes.

General Description:

The major goals in the design of PARMELA were flexibility and changeability. As a result, there are two portions of the code that serve as the skeleton, while most of the remainder consists of easily changeable or replaceable subroutines. The two portions are the main program, PARMELA, which responds to the input data and controls the logical flow of the program, and the subroutine, PARDYN, which controls the logic of the particle dynamics calculations.

The input data define the transport system, specify the control logic, and specify the input and output options. (Although the input is not usually on punched cards, each input record will be referred to as being a card, and must not contain more than 80 characters.) Each card recognized by PARMELA consists of an alphabetic label followed by a variable number of parameters. The label determines the action to be taken by PARMELA. The order of the data is not strictly prescribed, although certain rules must be obeyed.

For example, the elements in the transport system must be given in the proper sequential order. The label defines the type of element. Each element is defined by three or more parameters. The first three parameters have the same meaning for all elements: 1) The length of the element; 2) The aperture size at the downstream end of the element; 3) An output flag, indicating whether or not output is desired at the downstream end of the element. A complete list of all of the parameters for each element will be given later.

When an element is encountered in the input string, a sequence number is incremented, a "TYPE CODE" is set, and the parameters, including any internally generated ones, are stored in an array. A table of the longitudinal (Z) location of the
downstream end of each element is also created, with the first element beginning at \( z = 0 \). The next card is then read.

If the element encountered is a linac cell, then additional information concerning the field structure may be required. PARMELA allows for this possibility by calling on a subroutine to supply this information. For example, the subroutine may read an external data file to get a table of values for the fields. The necessary information would be stored in a labelled common block so that the subroutine which applies a cell impulse can get the proper field values. Strictly speaking, an element defined as a linac cell does not have to be a cell at all. It might be used for describing any arbitrary field structure over the length of the artificial cell.

The possibility of having a distributed element, or background field, is also included in PARMELA. An example would be the field produced by a set of helmholtz coils. This background field would be superimposed on the field due to any of the discrete elements. When a "COIL" card is encountered, a flag is set to indicate that a background field does exist, and a subroutine is called on to define this field so that another subroutine can apply the proper impulse to each particle at each integration step.

The electrons are each represented by six coordinates: The three position coordinates and the components of the momentum in these three directions. The initial coordinates are stored in a two-dimensional array, COR, dimensioned \((6, \text{imaa})\) (imaa is a parameter that is currently set equal to 2000). It is convenient to have a so-called "reference particle" having an initial \( z \)-coordinate and kinetic energy defined by a "RUN" card. The remaining four coordinates of the reference particle are set to zero, and the coordinates of the reference particle are stored in the first six locations in the array cor. The initial coordinates of the rest of the particles, if any, are generated in subroutine input. One or more "INPUT" cards may be used to define a variety of distributions. It is often convenient to generate the longitudinal coordinates by specifying their differences from those of the reference particle. This is one utility of the reference particle.

When the dynamics calculations are initiated (by encountering a "START" or a "RESTART" card), the coordinates are transferred from the COR array into the array, CORD, which is dimensioned \((7, \text{imaa})\). The seventh number for each particle is the sequence number of the element in which the particle is located. If the particle is upstream of the first element (\( z < 0 \)), then the sequence number is set to zero. In this case, the particle is assumed to be in a drift space until it reaches the first element. Particles downstream of the last specified element are also assumed to be in a drift space. In both cases, however, a background field may exist.

The total number of particles at the beginning of the calculation is stored in NPOINTS. Another variable, NGOOD, keeps track of the number of "GOOD" particles. All particles start as
good particles, and remain good until they are "LOST" for any of several reasons. When a particle is lost, its coordinates are swapped with the coordinates found in CORD(i,NGOOD), and NGOOD is reduced by one. This process keeps the good particles in the lower portion of the array and avoids the necessity of testing for lost particles. The initial coordinates (in array COR) are also swapped at the same time so that there will be a valid correspondence between the initial coordinates and the working coordinates. The calculations are aborted if the reference particle is lost, on the assumption that something drastic has gone wrong.

A particle can be lost for any of the following reasons: 1) its transverse coordinates are outside of the specified circular aperture in any element; 2) it arrives at a "CHOPPER" element at a phase such that it will be chopped from the beam; or 3) it drops behind the reference particle by more than a distance specified on the "ZLIMIT" card. (If no ZLIMIT card is included, then the default distance is a large value.) It is possible to set ZLIMIT to a negative value. When ZLIMIT is negative the particles are not lost but are kept in the calculation. However, they may not be included in the output. I.e. the buffer for a particular element is outputed when the reference particle is a distance ZLIMIT beyond the end of the element.

The beam dynamics calculations are started when a "START" card is encountered. An initial phase angle, WT 0 is specified on the start card, where the phase is related to time through a basic frequency specified on the run card. It is convenient to use WT 0 to shift the phase of the particles with respect to the elements. Otherwise, the phase of each element would have to be shifted. The other parameters on the start card are: DWT, the integration step size; NSTEPS, the number of steps to take at this step size; NSC, the number of steps between space-charge impulses; and NOUT, the number of steps between calling for output.

It is convenient to think of the independent variable, WT, as being the phase angle on a reference clock. Some elements in the system may have time-varying fields which may or may not oscillate at the reference frequency, and may have a phase shift with respect to the reference clock. One can think of these elements as having their own internal clocks which are set with respect to the reference clock. When a particle is located in one of these elements, the reference phase must be converted to a local phase for computing the proper field values.

At the beginning of each integration step, a test is made to see if a space-charge impulse should be applied. If so, a subroutine is called on to calculate the space-charge field and to apply the appropriate impulses to all of the particles. (Particles having Z < 0 are included in calculating the space-charge forces, but do not receive a space-charge impulse) Then each particle is transformed separately through the integration step, DWT. If a background field is indicated, its impulse is applied at the beginning of the step. The transformation for the
element in which the particle is located is applied next. If, during the step, a particle is located in more than one element, the step is divided into segments.

When a particle arrives at the end of an element, a check is made to see if output is desired (and permitted) at that point. If so, its coordinates are stored in the appropriate output buffer. After all of the "GOOD" particles have passed this boundary, subroutine OUTPUT is called on to process the information in that buffer.

The dynamics calculations are stopped after all particles have passed through the last element in the system, or after NSTEPS have been taken. More input cards are then processed. If a "CONTINUE" card is encountered, the calculations will be resumed using the values of DWT, NSTEPS, NSC, and NOUT specified on the continue card. One can also save the coordinates, using a "SAVE" card, for a possible restart (via a "RESTART" card) at these same conditions.

DETAILED DESCRIPTION OF MAIN PROGRAM:

After initializing some basic constants, and setting to zero the flags IFLD, NEL, and IP, PARMELA calls on the subroutine read to read the input file. Read returns an index which depends on the label on the card just read. The number of parameters associated with this card is stored in NN, and the parameters are stored in array VV. PARMELA recognizes the following labels:

<table>
<thead>
<tr>
<th>LABEL</th>
<th>parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN</td>
<td>IRUN, IP, FREQ, Z0, W0, LTYPE</td>
</tr>
<tr>
<td>DRIFT</td>
<td>L, APER, IOUT</td>
</tr>
<tr>
<td>CHOPPER</td>
<td>CELL, TANK, COIL, RUN</td>
</tr>
<tr>
<td>INPUT</td>
<td>OUTPUT, TITLE, SCHIEFF, ZOUT</td>
</tr>
<tr>
<td>ADJUST</td>
<td>START, RESTART, CONTINUE, SAVE</td>
</tr>
<tr>
<td>ZLIMIT</td>
<td>END, ERRORS, ROTATE, SBLOAD</td>
</tr>
<tr>
<td>TRWAVE</td>
<td>CHANGE, CFIELD, SOURCE, CATHODE</td>
</tr>
<tr>
<td>WIGGLER</td>
<td>SEXTUPOLE</td>
</tr>
</tbody>
</table>

These labels must appear on the data cards starting in column 1, and must be followed by a blank, comma, or equal sign. Following such a label, up to LMX parameters may be appended on the same or additional cards (LMX is specified in a parameter statement as 300 currently). Parameters are in free format, but in the proper order, and must be separated by spaces or commas. All alphabetic characters appearing after the label, except e, which indicates a decimal exponent, are ignored.

The parameters associated with each label are given below, followed by the parameter definitions. Parameters enclosed by parentheses are optional.
SOLENOID  L, APER, IOUT, B
QUAD      L  APER, IOUT BP
BEND      L  APER, IOUT, WR, ALPHA, BETA1, BETA2, PSI1, PSI2, R1, R2
BUNCHER   0., APER, IOUT, DWMAX, FREQB, PHIS, (WR)
CHOPPER   0., APER, IOUT, FREQC, PHIC, DPHIC
CELL      L, APER, IOUT, PHIO, E0, NC, DWMAX, FLAG, (0,0,B) (14 fourier coefficients of field) If fourier coefficients of cell are provided the will override the fourier coefficients given by the linac type. If you have several cells with the same NC the fourier coefficients of the last cell with NC will be used for all the cells. These may be the ones given by the linac type or specified explicitly on the cell card. If the fourier coefficients are given they must be the 12th through the 25 parameters on the card.
TANK      L, APER, IOUT, E0, NC, DWMAX, PHIO
COIL      Z, R, I, ZMIN, ZMAX this card applies a background solenoidal magnetic field. Z is position of coil, R is radius of coil, I is total current of coil, ZMIN and ZMAX are the limits the background field is applied over. ZMIN and ZMAX are required on the first coil card. There is no limit on the number of COIL cards used. A file called ‘bfield’ lists the magnetic field versus z if ZOUT is used.
INPUT     vv(1), ... vv(16) generalized input subroutine
          type 0, read np particles from file diout with probability of use based on particle charge in pc version. In cray version all particles are read in with the charge.
          type 1, np particles are generated randomly in three phase plane ellipses
          type 2, np particles are generated randomly in a six dimensional ellipse
          type 3, np particles are generated from output data from Egun data. Parameters are ntype, np, number of rays, phase spread, egun mesh size in cm. The file egun.dat contains the data. The format of the data is: ray#, r, rdot, zdot, tdot, r(cathode), i/r(cathode)/2pi. This is in PC version only.
          type 4, particle number np is positioned at six specified coordinates
          type 5, np particles are generated randomly in a four dimensional transverse hyperspace with random phase and energy spread within an ellipse.
          type 6, np particles are generated randomly in a four dimensional transverse hyperspace with uniform phase and random energy spread.
          type 7, np particles are generated in a kapchinski-vladimerskij distribution transversely with random phase and energy spread within an ellipse.
          type -n, random number generator reset to beginning
          type 8, np particles are generated randomly with uniform distribution in real space (x,y,z), then xp, yp, and zp are chosen from within each phase-plane ellipse. zp, zp are then converted to phi, w.
          type 9, gaussian distribution in x, and y and gaussian distribution in z, xp, yp, are zero and energy spread is zero. vv(1-6)=ntype, n, sigmar, rmax, sigmaz, zmax
sigmar and rmax are in units of cm and sigmaz and zmax are in degrees
this type used to be the following.
rectangular array in one of the three phase planes.
ntype,kind,hl,hr,nh,vb,vt,nv
type 10*n, as type n, but xp and yp are adjusted to
force the angular momentum to be zero.
valid for types 1, 2, 5, and 6.
vv(1)=type
vv(2)=no of particles or particle no
vv(3-8)=transverse ellipse parameters
vv(9+10)=phase and energy spread (degrees and MeV)
vv(11+16)=displacements of input beam (cm and m) phase spread or delta phase is entered in degrees

OUTPUT Use 5 if PARGRAF is used. Other wise it (1 to 4) depends on OUTPUT subroutines
TITLE next card image is read as an 80 character title
SCEFF depend on space-charge subroutine, SCEFF. the present one needs BEAMI, RMESH, ZMESH, NR, NZ, NIP, PL, OPT, FRM, RWAL
BEAMI =beam current in amperes.
RMESH =radial mesh size in cm.
ZMESH =longitudinal mesh size in cm.
NR =no. of radial mesh intervals (le 20)
NZ =no. of longitudinal mesh intervals (le 200)
NIP =no. of adjacent bunches
PL =distance between bunches. (if zero, pl=sce(3))
OPT =opt. if opt.lt.0, use only one ring. if opt=0,
use 2x2 array. if opt.gt.0, number of rings is determined by gaus depending on aspect ratio.
if opt even radial mesh intervals give equal volume rings.
if opt odd mesh intervals have equal dr
if opt.eq.4 or 5 assume metal wall at z=0. and calculate image charge.
FRM =frm, remeshing factor (default=1.5)
RWAL =rwall, radius of conducting wall. if zero, no wall
sce(11)=impact parameter, cm for point to point space-charge calculation. if zero use mesh space-charge calculation.
the code for the point to point calculation is commented out in the PC version. This calculation can take a lot of computer time if used. It is fully 3-d but it is noisy. It may underestimate the radial space charge force at high energy if the number of particles use is insufficient. If you use this 3d space charge calculation check its accuracy by comparing the radial expansion of the beam with either theory or the 2d scheff calculation
ZOUT none
ADJUST NP, do N1 through N2, *VALUE, NTYPE, ERR this card multiplies parameter NP by VALUE on all elements from N1 through N2. NTYPE AND ERR are optional. If NTYPE is specified it will only adjust parameter NP if it is an element identified by NTYPE. If ERR is specified the parameter will be multiplied by VALUE + a random number from -ERR to +ERR in % of VALUE. If VALUE is zero the the parameter will be multiplied by a random number from -ERR to +ERR.
--note-- this card does not apply to all parameters and no checking is done by the program to determine if you are modifying a valid parameter. ie the phase of a cell or tank is not a valid parameter but it field strength is a valid parameter. If NTYPE is 7 (ie. a cell), the phase is a valid parameter. In this case the phase will be increased by VALUE +(a random number -ERR to +ERR) the units are degrees in this case only.

START  WT0, DWT, NSTEPS, NSC, NOUT
RESTART DWT, NSTEPS, NSC, NOUT
CONTINUE DWT, NSTEPS, NSC, NOUT, DZ(change z of reference particle by DZ)
SAVE none (this saves all cordinates on disk, restart uses these cordinates.)
ZLIMIT ZLIM
ERRORS see comments in subroutine geners
ROTATE 0,APER,IOUT,DEG
SBLOAD 0,APER,0,ZLENGTH (Single bunch beam loading simulation based on LTYPE=2 at 1300 MHz) Frequency scaling is used.
TRWAVE L, APER, IOUT, PHIO, E0, NC, DWTMAX, FREQ, ICT, NMN, NMAX, PSHIFT, NW, NPRINT, Z1, Z2, R1, R2, DPHI, fourier coefficients (nmin=-5 to nmax=+5)
The first 9 paramaters must be on all trwave cards. nc is the traveling wave accelerator tank number. nc must be a number from one to seven each trwave card must have the same nc in a tank. each tank must have a unique number.
NW is the total number of trwave cards in a tank. each trwave card contains the data for one cell of the traveling wave accelerator. The first trwave card must have the first 13 parameters at least. If nprint is nonzero then Z1 through R2 must be defined. For example if NW equals 30 there must be 30 TRWAVE cards in sequence with no other element interwoven.
each cavity has gradient, beta, length, freq., phase and, in the middle, a gap. This code creates the E and B fields using the expansion of the field described in ‘LINEAR ACCELERATORS’ edited by Lopostolle and Septier pages 44 to 95
The fields are first calculated by superfish. The fields are input to the program by the fourier coefficients which are generated by a new version of efld called efldtr. these fourier coefficients are used to generate the fields with an expansion in Fourier-Bessel series. The fourier coefficients can be used on any trwave card. They must be parameters 20 to 30 and they will then be used on all subsequent trwave cards in that tank unless changed by another set of coefficients. ICT is the cell type. This must be a unique number for each type of cell. If ICT is positive the fields will be calculated from two standing waves which are stored in an array with index ICT. If ICT is negative the fields will be calculated from the fourier coefficients each time it is needed and the calculation will be much slower. The fields used will be identical in either case unless ICT is chosen wrong.
PSHIFT is in units of pi (e.g. 3/4pi per cell should have PSHIFT=.75, and 2/3pi per cell should have PSHIFT=.666667)
**IMPORTANT NOTE ABOUT TRWAVE**

The first cell of trwave actual starts in the center of the cell. The reason for this is to accurately simulate the fringe field into the traveling wave guide. The fringe field and the first half of the cell can be simulated with a cell card. The field of the cell card can be adjusted to provide a smooth transition from the cell to the traveling wave field. The phase of the cell card is exactly 90 degrees + the phase on the trwave card. The 8th parameter on the cell card must be +1. A default set of fourier coefficients for this cell card are:

```
0.1820485E+01,0.8322601E+00,0.1155573E+00,-.5371728E-01,
-.3360973E-01,-.7375432E-02,0.1037726E-02,0.1427306E-02,
0.4694677E-03,-.4060341E-05,-.6557005E-04,-.2415616E-04,
-.5552536E-06,0.2662174E-05
```

If these coefficients are used the field in the cell card is equal to the .76*field of the first trwave card. The length of the cell card is equal to the length of the first cell of the traveling wave accelerator. An example of a 3/4pi 1300MHz traveling wave accelerator is:

```
cell 8.65 2.54 1 90.000 5.728 1 5 1 0 0 0
0.1820485E+01,0.8322601E+00,0.1155573E+00,-.5371728E-01,
-.3360973E-01,-.7375432E-02,0.1037726E-02,0.1427306E-02,
0.4694677E-03,-.4060341E-05,-.6557005E-04,-.2415616E-04,
-.5552536E-06,0.2662174E-05
trwave 4.325 3.19 1 0.000 7.52 1 5 1300 5 -5 5 .75 35 0. 39.0 69. 30. 2. 60.
.00002 .000897 -.006674 -0.018529 .557918 .829216 .023465 -.013701 .000897
.000125 -.000026
trwave 8.651 3.19 0 0.000 7.42 1 5 1300 5
trwave 8.651 3.17 0 0.000 7.43 1 5 1300 5
```

The cell card in this example has 4.325cm of field in the beam tube and 4.325cm in the first cell of the traveling wave accelerator.

**CFIELD**

```
cell# Next card image contains name of file to read the fields for the cell# specified. The fields only need to be read in once for each cell# NC. The fields are in are in the format written by SFO7 on tape7. 40 increments(60 for the CRAY) in the z direction and 10 increments is the r direction. This can be changed if you want by changing the value specified in the PARAMETER statement in the source code of parmela the parameters are 'numz' and 'numr' which are currently 41 and 11 respectively in the FC version and 61 and 11 in the CRAY version. This card must be after a cell card that specifies the cell# and before the START or RESTART card.
```

**CHANGE**

```
NP,NE,VALUE  this changes the value of a parameter NP on element NE to VALUE. The length parameter is illegal.
```
not all parameters are legal and no checking is done.

**SOURCE**

Z,R,BGZ,BGR  this element puts one particle into the calculation at position (z,r) with momentum (bgz,bgr) at beginning of each time step

**WIGGLER**

L,APER,OUT,PERIODS,STEPS,B0,Ky/Kw,0,0,IPRIN

PERIODS = number of wiggler periods over the length L

B0 = wiggler field in gauss

Ky/Kw = ratio of Ky to Kw see equations below.

IPRIN = print flag if iprin .ne. 0 x,bgx,y,bgy,z,bgz is printed after each integration step.

Ideal wiggler. The magnetic field components are defined as follows:

\[
\begin{align*}
B_x &= B_0 \cdot \frac{K_x}{K_y} \cdot \sinh(K_x X) \cdot \sinh(K_y Y) \cdot \cos(K_w Z) \\
B_y &= B_0 \cdot \frac{K_y}{K_y} \cdot \cosh(K_x X) \cdot \cosh(K_y Y) \cdot \cos(K_w Z) \\
B_z &= B_0 \cdot \frac{K_w}{K_y} \cdot \cosh(K_x X) \cdot \sinh(K_y Y) \cdot \sin(K_w Z)
\end{align*}
\]

\[
K_w = \frac{2 \pi}{\text{lamda}(\text{wiggler})}
\]

\[
\text{lamda}(\text{wiggler}) = \frac{L}{\text{periods}}
\]

\[
K_x^2 + K_y^2 = K_w^2
\]

A wiggler with vertical focusing only, no horizontal focusing will have Ky/Kw=1. For equal focusing in both horizontal and vertical Ky/Kw=1/sqrt(2).

The equations of motion for the wiggler element are:

\[
\begin{align*}
\frac{dP}{dt} &= F = e \cdot (V) \times (B) \quad \text{ie} \quad e \cdot V \times B \\
\frac{m \cdot c \cdot d(\text{BETA} \cdot \gamma)}{dt} &= e \cdot c \cdot (\text{BETA}) \times (B)
\end{align*}
\]

\[
\begin{align*}
\delta(bgx) &= e/m \cdot (by \cdot Bz - bz \cdot By) \cdot \delta(t) \\
\delta(bgy) &= e/m \cdot (bz \cdot Bx - bx \cdot Bz) \cdot \delta(t) \\
\delta(bgz) &= e/m \cdot (bx \cdot By - by \cdot Bz) \cdot \delta(t)
\end{align*}
\]

\[
\text{constain abs}(P) = \text{constant} \quad \text{(momentum is conserved)}
\]

\[
P^2 = bgz^2 + bgy^2 + bgz^2 = \text{constant}
\]

The dynamics are done by drift-impulse-drift (leap frog method) but forcing the momentum to keep the same magnitude. Subroutine wiggler is sent a time interval dwtp (delta(t)) in degrees. the time step in the wiggler is the smaller of dwtp and that required by dwtp .lt. 360.*freq*delta(z)/(bz*c) where

\[
\delta(z) = \text{lamda}(\text{wiggler})/\text{steps}
\]

At each integration step.

\[
\text{drift:} \\
x = x + bx \cdot c \cdot \delta(t)/w \\
= x + bx \cdot \text{lamda}/360*\text{dwtp}/2.
\]
\[ y = y + by*\text{lamda}/360*\text{dwtp}/2. \]
\[ z = z + bz*\text{lamda}/360*\text{dwtp}/2 \text{ where } bx, by, \text{ and } bz \text{ are calculated by } bgx/\text{gamma}, bgy/\text{gamma}, bgz/\text{gamma}. \]

**impulse:**
\[ \text{delta(bgx)} = (by*Bz-bz*By)*e/(m*c)*\text{lamda}/360*\text{dwtp} \]
\[ \text{delta(bgy)} = (bz*Bx-bx*Bz)*e/(m*c)*\text{lamda}/360*\text{dwtp} \]
\[ \text{delta(bgz)} = (bx*By-by*Bx)*e/(m*c)*\text{lamda}/360*\text{dwtp} \]
\[ \text{bgx} = \text{bgx} + \text{delta(bgx)} \]
\[ \text{bgx} = \text{bgx} + \text{delta(bgy)} \]

to keep total momentum constant
\[ \text{bgz} = \sqrt{P^2 - \text{bgx}^2 - \text{bgx}^2} \]

**drift:**
another drift is applied as above.

end of integration step.

**SEXTUPOLE** L,APER,IOUT,Bpole/aper**2
Bpole/aper**2 is the strength of the sextupole in units of gauss/cm**2. It can be positive or negative.

**END** none end of input file.

definition of parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRUN</td>
<td>Identifying run number.</td>
</tr>
<tr>
<td>IP</td>
<td>Print flag. if IP is not zero, the input cards recognized by PARMELA are printed when read.</td>
</tr>
<tr>
<td>FREQ</td>
<td>Basic frequency of linac (Mhz).</td>
</tr>
<tr>
<td>Z0</td>
<td>Initial longitudinal position of reference particle (cm). The beginning of the first element is defined to be at z=0.</td>
</tr>
<tr>
<td>W0</td>
<td>Initial kinetic energy of reference particle (MeV).</td>
</tr>
<tr>
<td>LTYPE</td>
<td>Specifies type of linac cavities. ltype = 1 for disk-and-washer; 2 for side-coupled cavity; 6 for RTM side-coupled cavity.</td>
</tr>
<tr>
<td>L</td>
<td>Length of element (cm).</td>
</tr>
<tr>
<td>APER</td>
<td>Size of aperture at exit of element (cm).(radius)</td>
</tr>
<tr>
<td>IOUT</td>
<td>Output flag. if nonzero, output is desired at end of element.</td>
</tr>
<tr>
<td>B</td>
<td>Magnetic field (gauss).</td>
</tr>
<tr>
<td>DEG</td>
<td>Number of degrees coordinate system is rotated about</td>
</tr>
</tbody>
</table>
ZLENGTH  Distance centered on reference particle to apply single bunch beam loading energy correction.
BP      Magnetic field gradient (gauss/cm).
WR      Reference energy for this element (MeV).
ALPHA   Angle of bend for particle having energy WR (deg).
BETA1   Leading edge angle on bending magnet (deg).
BETA2   Trailing edge angle on bending magnet (deg).
PSI1    Fringe-field correction angle for leading edge of bending magnet (deg).
PSI2    Fringe-field correction angle for trailing edge of bending magnet (deg).
R1, R2  The radius of curvature of the leading and trailing pole face edges (cm). If zero straight edges are assumed.
DWMAX   Maximum energy gain in BUNCHER (MeV).
FREQB   Frequency of BUNCHER (MHz).
PHIB    Phase of BUNCHER (deg).
FREQC   Frequency of CHOPPER (MHz).
PHIC    Phase of CHOPPER (deg).
DPHIC   Half width of acceptable phase spread for CHOPPER (deg).
PHIO    Phase of CELL, TANK, or TRWAVE element (deg).
E0      Average axial electric field (MV/m) in CELL (or in the first cell in a tank) or in TRWAVE cell.
NC      Index to identify CELL number, or to identify TRWAVE tank number. There is no conflict between cell numbers and trwave numbers.
DWTMAX  Maximum integration step size in CELL or in TRWAVE (deg).
E0T     Average axial electric field times transit-time factor (MV/m).
FLAG    This is used to specify the configuration of the cell field. Normally FLAG=0 for symmetrical cells. Internally the fields are specified for the second half of the cell only. Then symmetry is used to provide the field for the first half of the cell. When fourier coefficients are used to determine the fields the coefficients are calculated on half a cell where z=0 is at the center. The fields should drop off to zero at L/2 where L is the length of the full cell.
         In cells where the fields are not symmetrical about the center, FLAG is set to +1 or -1 and the fields are specified for the full cavity. The fourier coefficients can still be used but are not as accurate. If FLAG is positive the orientation of the field are reversed from the direction they were originally calculated in SUPERFISH. If FLAG is negative they are not reversed! Special care must be taken to make sure the fields are oriented in the direction desired.
         The fields can be printed out if a TRWAVE card is used and NPRINT is non zero and Z1 and Z2 include the position of the cell card in question.
NCELLS  Number of identical cells in TANK.
NW      number of cells in traveling wave linac tank.
P       Phase shift between adjacent cells in TANK is p*pi
PSHIFT  phase shift between adjacent cells in traveling wave accelerator.

NPRINT  writes the fields at nprint locations from Z1 to Z2 at R1 and R2 to file 'trwdata'. The phase is incremented in units of DPHI from 0 to 360 degrees if DPHI is non zero.

Z1, Z2, R1, R2, DPHI see NPRINT.

BEAMI  Total current in the beam (amperes).

RMESH  Length of space-charge mesh in radial direction (cm).

ZMESH  Length of space-charge mesh in longitudinal direction (cm).

NR, NZ  Number of radial and longitudinal intervals in the space-charge mesh.

NIP  Number of "identical pulses" upstream and downstream of the particle bunch when calculating space-charge forces.

PL  pulse length, or repetition length, of the adjacent bunches for purpose of space-charge calculations.

OPT  If OPT < 0 use only one ring. If opt=0 use 2X2 array. If OPT > 0 number of rings is determined by GAUS depending on aspect ratio.

FRM  SCHEFF will remesh when reference particle energy is FRM times reference particle energy at last time mesh was generated.

RWAL  Radius of beam pipe for image charge. If zero, no wall.

WT0  Starting phase angle for dynamics calculations (deg).

DWT  Integration step size (deg).

NSTEPS  Number of integration steps to take.

NSC  Number of steps between space-charge impulses.

NOUT  Output is wanted after every NOUT steps.

ZLIM  If a particle falls more than ZLIM cm behind the reference particle, it is dropped from the calculation. if ZLIM is negative particle is not dropped from calculation but it won't be included in output.

The following action is taken after each type of data, after which another data card is read.

After RUN: The RUN card is normally the first card read. Subroutine read recognizes the RUN label and prints the run number, date, and time, followed by the frequency, starting location of the reference particle, and the energy of the reference particle. If IP is nonzero, then subsequent cards will be printed as they are read. The parameters FREQ, Z0, and W0 are stored, and COR(i,1) for i=1,2,3,4 are set to zero. Z0 is stored in COR(5,1) and the normalized momentum (p/mc) is calculated from W0 and stored in COR(6,1). NPOINTS is set to one, unless it is already greater than zero, and NGOOD is set equal to NPOINTS.

After TITLE: the next card is assumed to be a title and is
read with an 8a10 format.

After any element (DRIFT, SOLENOID, QUAD, BEND, BUNCHER, CHOPPER, CELL, and TANK): If the number of parameters, NN, is more than 11 or less than 2, an error message is printed and a flag is set so that the read subroutine will skip the data until another run or end card is encountered. If the number of elements, NEL, is already (100 in the PC version or 1000 in the CRAY version), then an error message is printed and the job is aborted. Otherwise, NEL is incremented by one, the type code for the element is stored in NTYPE(NEL), and the NN parameters are stored in EL(N,NEL), N=1,NN. The longitudinal position of the downstream boundary of the element is computed by adding the length of the element to the location of the previous element, and this value is stored in ZLOC(NEL). Additional action is taken after the following elements:

After BEND: The angles ALPHA, BETA1, BETA2, PSI1, and PSI2 are converted from degrees to radians. The normalized momentum of the particle with energy WR is stored in EL(10,NEL). The radius of curvature of the bend is calculated and stored in EL(4,NEL), overwriting WR which is no longer needed.

After BUNCHER: The reference energy for the buncher is taken to be the energy of the reference particle, W0, unless the buncher card has 7 or more parameters, in which case the reference energy is the seventh parameter. This is used in calculating a normalized momentum, from which a value is computed for the wave number, k, which is stored in EL(8,NEL). this value is used in the buncher subroutine in calculating the modified bessel functions of kr.

After CELL: The average axial electric field strength, E0, is converted from MV/m to MV/cm. sin(PHI0) and cos(PHI0) are computed and stored in EL(9,NEL) and EL(10,NEL), respectively. If the seventh parameter, DWTMAX, is absent, it is set to 10 degrees. Subroutine CELLFLD is called on to set up the information about the field structure for this cell. This information will be used by subroutine CELLMIMP for applying the impulses in the cell.

After TANK: The length of the tank is divided by the number of cells to get the length of each cell, which is then stored in EL(8,NEL). E0T is converted from Mv/m to Mv/cm. The value for BETAW, the relative velocity of the electromagnetic wave in the tank, is computed and stored in EL(9,NEL). The square of the wave number is computed and stored in EL(10,NEL).

After COIL: The flag IFLD is set to one, and subroutine FIELD is called on to supply the information for the background field. The fields off axis are calculated to sixth order from the fields on axis. To sixth order the field expansions are

\[
B_z(z,r) = B_z(z,0) - r \left( \frac{d B}{d z} - \frac{d B}{d z} * r / 16 + \frac{d B}{d z} * r / 576 \right)/4
\]
\[
R_r(z,r) = -r\left(\frac{dB}{dz} - \frac{d}{dz}\left(\frac{r}{8} + \frac{B}{dz}\frac{r}{192}\right)\right)/2
\]

The derivatives are calculated and accumulated in the bfield array at the same time the fields on axis are calculated.

After ERRORS: Subroutine GENERS is called on to generate the appropriate errors.

After INPUT: Subroutine INPUT is called on to supply a set of input coordinates, after which the parameter NGOOD is set equal to NPOINTS.

After OUTPUT: The NN parameters on the card are stored in the array OPTCON.

After SCHEFF: The NN parameters are stored in array SCE, and subroutine SCHEFF is called on to initialize the space-charge tables. The first parameter on the card is assumed to be the beam current in amperes.

After ZOUT: Subroutine ZOUT is called on to print the location and type of each element in the system. The B field from the coil cards are also printed at the end of the element. If the element is a cell the amplitude and phase are printed also.

After ADJUST: Subroutine ADJUST is called. This allows the user to do special tasks by writing a subroutine called adjust. The current routine can adjust the value of most parameters on an element.

After ZLIMIT: ZLIM is set.

After START: The phase WT is set equal to the initial value, WT0. NBUF, the number of particles whose coordinates have been stored in the output buffer, is set to zero. The coordinates are loaded into array CORD from the array COR, and GAMMA is calculated and stored in the array GAM. The sequence number of the element in which each particle is located is stored as the seventh parameter in array CORD. The largest sequence number for all particles is stored in LEN. Subroutine PARDYN is called on to control the logic for the dynamics calculations.

After CONTINUE: DWT, NSTEPS, NSC, and NOUT are reset to the values specified, and subroutine PARDYN is called on to continue the dynamics calculations.

After SAVE: The parameter NGOOD is saved in NGS, and the coordinates in array CORD are stored in COR. The present value of the phase, WT, is stored in WTO. They are then written to disk.

After RESTART: The parameters DWT, NSTEPS, NSC, and NOUT are reset to the values specified, and NGOOD is set to NGS. The coordinates are read from disk that were written by the SAVE card and then same procedure described after start is then followed. This provides a means of changing the the beam line and starting
the calculation over just before the part that is changed without rerunning the whole problem over. It can save a lot of time.

After END: The program is terminated.

Detailed description of subroutine PARDYN:

The purpose of PARDYN is to control the logic involved in following the electron coordinates through the specified number (NSTEPS) of integration steps of size DWT, or until all particles are either lost or past the last element in the system.

At the beginning of each step, sin(WT) and cos(WT) are calculated and stored. If space-charge impulses are to be included, they are applied before the first step and after each NSC steps. Subroutine SCHEFF is called on to calculate and apply these impulses for a phase interval of NSC*DWT.

The parameter NP, which keeps track of the particle number, and the parameter NLE, which counts the number of particles that have passed the last element in the system, are both set to zero. NP is then incremented by one, and the coordinates are transferred from array CORD(i,NP), i=1,7, into the variables X, BGX, Y, BGY, Z, BGZ, and NE, and from GAM(NP) into GAMMA, all of which are located in a labelled common block, PCORD. WT and DWT are initialized to WT and DWT, respectively. These quantities keep track of the phase angle and the remaining phase step for each particle.

NE is the sequence number of the element in which the particle is located. If 0 < NE < NEL+1, the type code, NT, for the element is obtained from NTYPE(NE). If NT=7, the element is a linac cell, and subroutine CELLIMP is called on to calculate and apply the impulse due to the field in the cell. The impulse is multiplied by the phase interval, DWTP. However, if DWTP > DWTMAX, as specified for the cell (or the default value of 10 degrees), DWTP is set to DWTMAX by subroutine cellimp.

The longitudinal velocity is calculated along with the distance, DZ, that the drifting particle would travel in the phase interval, DWTP. If the particle would drift past the end of the element (or past Z = 0, if it is upstream of the first element), then DWTP and DZ are reduced so that the particle would just arrive at the end of the element (or Z = 0), and IEND is set to one. The particle then undergoes a coordinate transformation depending on the element in which it is located. A check is made to see if the particle is within the specified aperture for the element. If not, it is lost, and the coordinates for this particle are swapped with the coordinates of particle NGOOD, and NGOOD is reduced by one. The forward momentum is checked, and if it is zero or less, the particle is discarded. Otherwise, WTP is incremented by DWTP and a new value is computed for the remaining phase interval, DWTP.

If the particle is at the end of an element, indicated by IEND being nonzero, several checks are made. If NP = 1, then this is the reference particle, and its phase and energy are...
saved in the arrays PR and WR. These values are useful in describing the longitudinal phase space of the beam. If the reference particle is lost, a message is printed and PARDYN releases control.

If output is desired at the end of the element, and if the conditions described below are satisfied, then the phase-space coordinates of the particle are stored in array OUTCOR, along with NP. Because there is only a limited number of output buffers, however, it may not be permissible to save the coordinates of all the particles at every specified element. In the main program, after a start or restart card, the NEB and NBUF arrays are set to zero. NEB(n) will contain the element sequence number at which coordinated will be stored in the nth output buffer, and NBUF is the number of particles that have been stored in the nth output buffer, in OUTCOR. When the first particle arrives at the end of an element a buffer will be assigned to that element.

After all of the particles have been followed through the time step, DWT, subroutine output will be called if output is desired at this time step. Also, a check is made to see if there are any good particles that have not yet reached the end of each element specified in NEB array. IF all of the particles have passed any element in this array, then subroutine output is called on to process the coordinates in that buffer and NBUF is set to zero for that buffer.

GRAPHICS OUTPUT FOR PARMELA

Several graphics formats are now available for use with PARMELA. The mode of operation is similar to that used by PARMTEQ, the RFQ version of PARMILA. That is, the particle coordinates are written on a disk file (TAPE2 and/or TAPE4), and after PARMELA has finished running, this disk file may be processed by another program PARGRAF (executable file is PARGX). A small amount of data on a file SIMPLE is needed to direct PARGX.

To use this mode, the OUTPUT card used with PARMELA must be

OUTPUT 5

All but the first parameter on the output card are ignored. When all particles have passed the end of an element at which output is desired. The OUT 5 subroutine is called, and the particle coordinates are written on TAPE2 and/or TAPE4.

To process the data, run PARGX. Make a file called SIMPLE as follows.

SUBNUM n (n = 3, 5, 6, 7, 8, or 9)
OUTPUT l, i, k1, k2, k3 ...
OPTCON p1, p2, p3, ...
BEGIN
END

The n on the subnum card denotes which type of graphics output is desired. The integer 6, 7, 8, and 9 are used for
compatibility with our other PARMILA type processors. Each of these types will be explained in detail later.

The first parameter on the output card, \( l \), must be either 1 or 2. A "1" means that output is desired at the ends of elements \( k_1, k_2, k_3, \ldots \). The second parameter \( i \), should be either 0 or 1. A "1" means that output is wanted at these elements, and a "0" means that output should be suppressed at these elements. When \( l=2 \), then \( k_1, k_2, \) and \( k_3 \) are treated like indices in a "DO LOOP". Output is wanted (if \( i=1 \)) at elements \( k_1 \) through \( k_2 \) in steps of \( k_3 \). If \( k_3 \) is absent it is assumed to be 1.

The parameters on the OPTCON (OutPuT CONstants) card control the graphics boundaries, etc. and will be explained below, because the depend on the type of output that is called for.

PARGRAF recognizes one more label 'FAST'. FAST turns off the emittance calculations. These calculations can take a considerable amount of computer time for a large number of particles.

SUBNUM 3

This is a new option that is available. Four plots are produced by this subroutine. They are:
1) a plot of the phase of each particle at end of current element vs phase of the particle at \( z=0 \).
2) a plot of the energy of each particle at end of current element vs phase of the particle at \( z=0 \).
3) a plot phase-energy phase space projection at end of current element.
4) a plot of \( xp \) vs \( x \) at end of first element selected for display by this subroutine of particles still good at end of current element. This is useful for determining the transverse acceptance of an accelerator or beam line.

The optcon parameters are:
- phase-in max = optcon(1)
- phase-out max = optcon(2)
- energy-out max= optcon(3) (in kw, for plotting ease needs no decimal places)
  - optcon(4) to optcon(7) are \( x_{max}, kx, x_{max}, kxp \) (\( x_{max} \) in units of cm, \( kx \) is number of decimal places, \( x_{max} \) in units of milliradians, and \( kxp \) is number of decimal places).

SUBNUM 5

Plot 2 profiles; \( x \) vs \( z \) and \( r \) vs \( z \).
args in OPTCON are \( z_1, z_2, nz, x_{scale}, kx, r_{scale}, kr. \)
optionally followed by: \( npair, (z_1,r_1), \ldots (z_{npair},r_{npair}) \)
\( nz \) is number of intervals for pip marks and labels. \( x_{scale}, \) and \( r_{scale} \) are the 2 vertical limits.
\( kx \) and \( kr \) are the number of decimal places in the scales.
\( (z_1,r_1) \) to \( (z_{npair},r_{npair}) \) define endpoints of lines drawn to show geometry of beam tube or accelerator.

SUBNUM 6
Plot input and output phase-space projections on the x-x', y-y', and deltaPHI-deltaW planes. There are 16 parameters on the OPTCON card. The first 8 concern the input plots and the last 8 are for the output plots. The parameters are Xmax, Kx, X'max, Kx', and deltaPHI, Kphi, delataW, Kw.

Xmax denotes the graph boundary for X and Y.

X'max denotes the graph boundary for X' and Y'.
deltaPHImax denotes the graph boundary for deltaPHI.
deltaWmax denotes the graph boundary for deltaW.

Kx is the format code for the X and Y scales, and denotes the number of decimal places to use in the scale (each scale has 4 divisions between + Xmax).

Kx', Kphi, and Kw are format codes for the other axes.

SUBNUM 7

Makes 4 plots on a page according to ntype. The OPTCON parameters are:

OPTCON ntype, Xmax, Kx, X'max, deltaPHI, Kphi, deltaW, Kw

ntype= 1, plot deltaPHI-deltaW, phase spectrum, energy

ntype= 2, plot X-X', Y-Y', deltaPHI-deltaW, X-Y


SUBNUM 8

plot 2 coordinates as a function of distance along transport system. The quantities are designated by 2 integers IQ1 and IQ2, having the following meanings:

IQ     quantity to be plotted
1     X coordinate of all particles
2     Y coordinate of all particles
3     deltaPHI coordinate of all particles
4     deltaW coordinate of all particles
5     Xbar and Xmax
6     Ybar and Ymax
7     deltaPHIbar and deltaPHI
8     deltaWbar and deltaW
9     X' coordinate of all particles
10    Y' coordinate of all particles

The OPTCON parameters are:

OPTCON IQ1, IQ2, Z1, Z2, Nz, Kz, X1max, Kx1, X2max, Kx2 where

X1max and X2max refer to the maximum values of the 2 quantities being plotted; Kx1 and Kx2 are their respective formats. Z1 and Z2 are the beginning and ending distances along the system in cm, and Nz is the number of intervals for the Z-scale, Kz is the Z-scale format.

SUBNUM 9

plot 3 quantities versus distance along the system X, deltaPHI, and deltaW.

OPTCON Z1, Z2, Nz, Xmax, Kx, deltaPHI, Kphi, deltaWmax, Kw
Abstract

Note: This document is intended to be read in conjunction with Kirk MacDonald’s notes on his changes to the program Parmela, and in conjunction with the original notes describing Parmela. Where the three conflict, this document supersedes the Kirk MacDonald notes, which in turn supersedes the original notes.

This version of PARMELA has been modified to permit modelling of space-charge correlation corrected electron guns (with diagnostics specifically for this purpose) with or without axisymmetry (through 3-D field maps for RF cavity fields and several fully 3-D space charge calculating algorithms) with provision for approximating wakefield effects in structures (by applying kicks in a non-self-consistent fashion derived from another code, such as ABCI/Xwake/W3WAK, etc. Solenoidal focussing fields may be modelled either using one-dimensional maps (and employing Busch’s Theorem) derived from a collection of Amperean loops, from a map of Bz(φ), from an axisymmetric map of [Br(φ,ρ),Bz(φ,ρ)], or from a comprehensive map of [Bx(x,y,z),By(x,y,z),Bz(x,y,z)].

In addition, two bug fixes from the canonical LLoyd Young version of PARMELA are in place: (1) space charge fields on the cathode are not subject to the curious weighting function present in the original, which caused underestimation of the longitudinal decelerating field on the cathode, and hence lead to overestimates of the quantity of charge extractable, and underestimates of the energy spread; and (2) numerical (completely non-physical) bunching of the particles in the longitudinal direction has been eliminated by introducing mixing (by dithering the timestep value) to avoid coherent excitation of any longitudinal bunch harmonics.

Longitudinal and transverse wakefield modelling has been added by applying a kick to each particle at the exit of specified elements derived from wake potential tables developed by, for example, ABCI, Xwake or TBCI. The approach is clearly not self-consistent and meant only for applications where wakefield effects are small compared to space charge and RF effects.

*E-mail address: ecolby@calvin.fnal.gov, 43064:ECOLBY
Graphics output from this version of PARMELA is nil. All output is in the form of ASCII data files. A large number of macros (> 200) have been written for analyzing and displaying the data from all of the various diagnostics using the program Matlab. A small number (20) macros also exist for Mathematica, but are neither maintained nor complete.

The traditional 'restart' capability of PARMELA has been extended to allow a running simulation to be stopped at any time, with a restart file and the required change to the input deck being automatically generated. To stop a Parmela run at any time, type:

```
% kill -15 <pid>
```

at the UNIX prompt, once you have established the process id for the running PARMELA process. For large jobs, PARMELA may take up to a minute to terminate. To resume execution, simply type the command 'parmela'. The restart file, named 'parmrstrt' contains a complete description of the phase space, as well as other parameters needed for PARMELA to resume execution. The parmin000 file is automatically modified to include the required 'restart' card after the 'run' card. Of course, if you desire to kill the process outright, do so with the usual `kill -9 <pid>`.

Existing PARMELA command cards that have been modified and new PARMELA command cards are listed below, both in summary form with changes in bold print, and in detailed form, explaining the meaning, units, and default values for the associated parameters.

Existing commands cards that have been modified

CONTINUE DWT NSTEPS NSC NOUT NSLICE
INPUT 11 NP SIGMAT MAXT SIGMAR MAXR W0 DW0 DWT HBX HBY HBPHI HBW0 ASPECT XOFF YOFF ANOISE FNOISE
RUN IRUN IP FREQ Z0 W0 LTYPE INTERP THRESH
SCHEFF BEAMI DRMESH DZMESH NR NZ NIP PL OPT REMESH RVAL POINT SOL NCH LINEL
START PHI0 DWT NSTEPS NSC NOUT NSLICE NSPT

New command cards

ASPECT AR
AUTOBUCK RI,RO,ZL,ZU
BFSETUP
BSHIFT XSHIFT,YSHIFT,XP SHIFT,YP SHIFT
BSOLENOID RI,RO,ZL,ZU,J
BZMAP SCAL EF
CCELL LX LY LZ APERX APERY IQUT PHI1 PHI3 E1 E3 NC DWTMAX SYM CFREQ
CTYPE BZ NFC NFC2
COLLIM TYPE LOW HIGH
FFKICK
DFILES TYPE1 NTYPE1 PAR1 PAR2 PAR3 ... TYPE2 NTYPE2 PAR1 PAR2 PAR3 ...

2
This card will allow approximation of elliptic cavity modes by scaling the RF field components of an axisymmetric cavity (generated with a preceding CELL card) by 1/AR in the horizontal direction and by AR in the vertical direction.

**AUTOBUCK** RI, RO, ZL, ZU

This card declares the physical dimensions and location of a bucking solenoid placed behind the cathode. Coordinates assume the point (r=0, z=0) to be the center of the photocathode. This card must follow the BSOLENOID cards of the solenoids whose fields are to be compensated. The required current density needed to yield Bz=0 on the photocathode is automatically calculated and applied. Up to 5 background solenoids can be handled by this routine.

RI - Inside radius of bucking coil
RO - Outside radius of bucking coil
ZL - Lower z coordinate of coil
ZU - Upper z coordinate of coil

**BFSETUP**

This card, when placed on the first or second line of the input file sets the number of particles to 2 and disables space charge calculations to allow rapid background solenoid field adjustment without the need to modify the SCHEFF or INPUT cards.

**BSHIFT** XSHIFT, YSHIFT, XPSHIFT, YPSHIFT
This card will shift the magnetic center of the background solenoid field by the vector \((x_{\text{shift}}, y_{\text{shift}})\) and the orientation of the solenoid by angles \((X_{\text{PSHIFT}}, Y_{\text{PSHIFT}})\) to allow modelling of alignment errors.

**BSOLENOID** \(RI, RO, ZL, ZU, J, \{Z_{\text{low}}, Z_{\text{high}}\}\)

This card defines a set of coils that fill the prescribed \(r\) and \(z\) coordinates with a conductor carrying a current density \(J\). This card defines a background solenoidal field ranging from \(Z_{\text{low}}\) to \(Z_{\text{high}}\).

- **RI** - Inside radius of background solenoid coil
- **RO** - Outside radius of coil
- **ZL** - Lower \(z\) coordinate of coil
- **ZU** - Upper \(z\) coordinate of coil
- **J** - Current density averaged over the entire cross section (in \(\text{A/cm}^2\))
- **Z_{low}** - (Declare with first BSOLENOID card only) lowest \(z\) location to have non-zero background field (must be > \(1\times10^{-13}\))
- **Z_{high}** - (Declare with first BSOLENOID card only) highest \(z\) location to have non-zero background field

**Notes:** This card causes an array to be loaded with 1000 on-axis B-field values ranging from \(Z_{\text{low}}\) to \(Z_{\text{high}}\). Setting \(Z_{\text{high}}\) too far away will adversely affect the accuracy of the B-field by making the mesh too coarse.

**BZMAP** **SCALEF** **LOWZ** **HIGHZ**

FILENAME

This card provides for direct reading of the background magnetic field strength from an external file named FILENAME.

- **SCALEF** - Scale factor to rescale fields with. (If none entered, defaults to 1.0 => use field map exactly as read in.)
- **LOWZ, HIGHZ** - Defines the region of field information to read in from the POISSON file, in POISSON coordinates.

**Notes:** Poisson must be run to produce the output file OUTPOI in a certain format. In particular, choose your coordinates such that the \(z\) axis corresponds to the \(L\) index (in the ‘‘y’’ direction) of Poisson, then use the following output option controlling CON
values (enter when running Poisson):
*42 1 1 1 LMAX
where LMAX is the upper mesh index of the problem mesh. (Look in file OUTLAT for this information.)

Make sure to declare the appropriate symmetry:
*19 1 *46 1
is usually the correct choice for solenoids. See the Poisson/Superfish reference manual for details.

CCELL LX LY LZ APERX APERY IOUT PHI1 PHI3 E1 E3 NC DWTMAX SYM CFREQ CTYPE FLAG NFC NFC2
FILENAME [FILENAME2]

This card is intended to introduce fully 3-D RF cavity fields by one of two methods: by the reading of a field map produced by another code, such as Superfish or HFSS, or by building the field up from a triple Fourier series expansion. Both the internal and external field maps are expressed in Cartesian coordinates. Up to 4 maps may be stored in memory for use in simulations.

LX,LY,LZ - X,Y, and Z dimensions (in cm) of a rectangular cavity (Lx and Ly are only used when using Fourier series for the fields)
APERX,APERY - X and Y dimensions (in cm) of the (elliptic) exit aperture (viz. the distance from the beam axis to the nearest obstruction in the given direction; not the full width of the opening)

Important note: the aperture values also declare the transverse dimensions of the internal field mesh. Choosing unreasonably large values of APERX and APERY to neglect particle loss at the exit will adversely affect the field accuracy.

IOUT - Print status (Vestigial option from CELL card)
PHI1 - Phase of lowest mode of cavity
PHI3 - Phase of next harmonic of cavity (Defaults to PHI1) [Note: this is NOT the relative phase of the first and third harmonics]
E1 - Peak Electric Field strength on axis [Ez(0,z)] (in MV/m) for
fundamental mode

E3  - Peak Electric Field strength on axis [Ez(0,z)] (in MV/m) for next excited harmonic. (Defaults to E1/3)

NC  - Cell identifier number (other cells with this number will derive their electric field structure from the same Fourier coefficients)

DWTMAX - Maximum temporal step in this cell (Defaults to 10 degrees)

SYM  - Symmetry: 1=>upstream half, -1=>downstream half, 0=>full cavity

CFREQ - Cell frequency in MHz (Defaults to FREQ as set on RUN card)
  Setting CFREQ to 0 causes a DC field of value E1 to be applied.

CTYPE - Cell type code. CTYPE=1, code will assume an ‘ideal’ rectangular cavity with only the lowest (TM000) mode excited. Type CTYPE=2 assumes an ‘ideal’ rectangular cavity with the first and third (TM000 and TM222, respectively) modes excited in a E1:E3 amplitude ratio. CTYPE=10 (resp. 20) derive field structures for one mode (resp. two modes) from the Fourier coefficients listed in FILENAME (resp. FILENAME for fundamental, FILENAME2 for next harmonic)

CTYPE = -99,-199 => read full field map from external file
FILENAME for fundamental, and (if ctype=-199) file FILENAME2 for next harmonic. External files are overwritten on the first reading with properly interpolated field maps.

The field maps should cover the last half of the cell. For a half cell, this means a region of dimensions APERX x APERY x Lz, while for a full cell, the region APERX x APERY x Lz/2. The mesh of the external field map should not exceed the declared resolution of the internal map (40x40x100) in any dimension. The mapped fields should be normalized. (Peak field in entire map should have an absolute magnitude no greater than 1.0)

FLAG  - Determines which field components to turn off during the run. (Useful for separating out the contributions of RF and space charge in the emittance and examining the effects of ponderomotive focussing). Defaults to 0.
  0  - Use all components (Ex,Ey,Ez,Hx,Hy,Hz)
  1  - Electric focussing (Ex,Ey,Ez,0,0,0)
  2  - Magnetic focussing (0,0,Ez,Hx,Hy,0)
  3  - Accelerate only (0,0,Ez,0,0,0)

NFC  - Number of Fourier coefficients to be found in file FILENAME
- (Max: 10x30x30=9000). The filename (16 characters or less) must be written on the line immediately following the CCELL card.

**NFC**
- Number of Fourier coefficients to be found in file FILENAME2
- (Max: 10x30x30=9000). The filename (16 characters or less) must be written on the line immediately following the FILENAME card.

**FILENAME**
- The file FILENAME should be in the same directory as the Parmela executable, and should list the (normalized) Fourier coefficients in the following record format:

\[ IJKA_{ijk} \]

Note that since the indices of each Fourier coefficient accompany it on the same line that the ordering of the coefficients in the file need not be sequential.

The triple Fourier Series used is:

\[
E_x(\vec{r}, t) = \sum_{i=0}^{30} \sum_{j=0}^{30} \sum_{k=1}^{10} A_{ijk} \sin k_x x \cos k_y y \sin k_z z
\]

\[
E_y(\vec{r}, t) = \sum_{i=0}^{30} \sum_{j=0}^{30} \sum_{k=1}^{10} A_{ijk} \cos k_x x \sin k_y y \sin k_z z
\]

\[
E_z(\vec{r}, t) = \sum_{i=0}^{30} \sum_{j=0}^{30} \sum_{k=1}^{10} A_{ijk} \cos k_x x \cos k_y y \cos k_z z
\]

\[
k_x = \frac{2\pi (i + 1/2)}{L_x} \quad k_y = \frac{2\pi (j+1/2)}{L_y} \quad k_z = \frac{\pi k}{L_z}
\]

The origin of coordinates has been taken to be on the beam axis in the upstream opening of the cavity.

**FILENAME2**
- This file is required only when CTYPE=20 is chosen, and should have the same structure as the other Fourier coefficient (or
FIELD MAP INFORMATION: the field map is generated and stored in common /c3flds/ for the downstream half of the cavity only, on a 40x40x100 point mesh.

The field map file should cover the downstream half of the RF cell over the following physical dimensions: -APERX to APERX, -APERY to APERY, Lz/2 to Lz for a full cell (SYM=0), or 0 to Lz for a half cell. (SYM=-1 or +1). Note that even if you are using the a +1 symmetry half cell, the field map must still cover the downstream half RF cell, the appropriate reflection of coordinates being done by PARMELA. The origin of coordinates is on axis in the upstream side of the map.

The ASCII file should be in the following tab- or space-delimited format:

Low-x High-x Step-x !Omit these three header lines
Low-y High-y Step-y !for the binary file
Low-z High-z Step-z
Ex(1,1,1) Ey(1,1,1) Ez(1,1,1) Hx(1,1,1) Hy(1,1,1) Hz(1,1,1)
Ex(2,1,1) Ey(2,1,1) Ez(2,1,1) Hx(2,1,1) Hy(2,1,1) Hz(2,1,1)
... ... ... ... ...

The first three lines give the physical mesh bounds (in [cm]) and the mesh size. (also in [cm]). PARMELA will truncate meshes larger than its own internal mesh and will interpolate meshes that are coarser than its internal mesh, but will return an error if the loaded mesh covers a physical region smaller than the internal mesh, or is finer than the internal mesh. (recompile changing the parameters NMESHX, NMESHY, and NMESHZ if you desire a finer mesh.) The E-field values must be listed at the time when the e-field is a maximum, and the H-field values must be listed at the time when the H-field is a maximum, one quarter period later. Note that the indices permute in standard FORTRAN ordering, with the leftmost index (labelling x mesh planes) incrementing the fastest, and the rightmost (labelling z mesh planes), the slowest.

SUPERFISH /
OUTSHY files: The conversion program ‘`poi2par’’ will read an unedited OUTSHY file to produce a PARMELA binary field map file. The OUTSHY file must cover a radial region ~5\% larger than the intended region to be used in PARMELA, and must cover one half of a cell. (ie. If you’re mapping a full RF cavity, use OUTSHY to produce two separate files, one covering the upstream half, the other the downstream half, then process separately and include them in the PARMELA run with two separate CCELL cards.) For optimal performance, choose the Superfish mesh resolution to match the PARMELA mesh resolution: 40 x 40 x 100 mesh points.

HFSS Calculator
Field Dumps: The conversion program ‘`hfsss’’ (still under development) is designed to read an HFSS space calculator dump of the E and H fields over the irregular tetrahedral mesh and produce a PARMELA binary field map. As with OUTSHY maps, full RF cells must be mapped into two separate files, and entered on two contiguous CCELL cards.

URMEL files: A conversion program is in the works, but is not yet finished.

MAFIA files: MAFIA output files are unsupported at this time.

SOPRANO : Output from Soprano is unsupported at this time.

ARGUS : Output from Argus is unsupported at this time.

COLLIM TYPE LOW HIGH

This card provides for custom spatial filtering of the beam, as is accomplished with a collimator.

TYPE - Code to describe what type of collimator is used:
   1 - vertical slit (collimates in horizontal plane)
   2 - horizontal slit (collimates in vertical plane)
   3 - user defined (modify the routine COLLIM)

LOW - Lower coordinate bound, in [cm].
HIGH - Upper coordinate bound, in [cm].

EQUAD ZSTART ZEND K XFACT YFACT
This card is designed to allow an external quad to be placed around an RF cavity. This card works only with the CCELL card.

ZSTART - Starting z location of the quadrupole field, in global coords. (with the surface of the photocathode taken to be z=0) [cm].

ZEND - Ending z location of the applied quadrupole field [cm]

K - Gradient, k>0 => horizontally focusing, [Gauss/cm]

XFACT - Multiplier for quad strength in horizontal plane

YFACT - Multiplier for quad strength in vertical plane

FFKICK

Place this following a CCELL card to cause the transverse kick due to the fringing fields in the previous RF cavity to be calculated by a method analogous to Busch’s theorem, using Rosenzweig’s form for the integrated transverse kick:

\[ \Delta(\gamma,\beta_y) = y\left(\frac{eE_0}{m_e c^2}\right) \cos\left(\frac{\pi x}{L_x}\right) \sin(\omega t + \phi_0) \]

INPUT 11 NP SIGMAT MAXT SIGMAR MAXR W0 DW0 DWT HBX HBY HBPHI HBW0 ASPECT XOFF YOFF ANOISE FNOISE TANOISE TFNOISE FILENAME

ITYPE - ITYPE=11 invokes Jerome Gonichon’s "Quiet Start"

NP - Number of particles

SIGMAT - Temporal sigma (in picoseconds). If SIGMAT < 0, then a user-defined subroutine called LASER will be called to produce the temporal behavior of the laser pulse. See preamble to subroutine for specific instructions.

NOTE: If ANOISE is nonzero, then this is the risetime of the laser pulse in picoseconds.

MAXT - Cut on temporal distribution (in picoseconds), half-length of flat top if ANOISE is nonzero.

SIGMAR - Spatial sigma (in cm) = horizontal dimension of beam

If SIGMAR is negative, a user defined subroutine XSECTN will be called to produce a custom cross section shape. If SIGMAR=-666, then (X,Y) distribution will be generated by the rejection method from the image file FILENAME.

MAXR - Cut on spatial distribution (also in cm) If SIGMAR=-666, then RMAX specifies the scaling factor to use in adjusting the magnification of the input. Defaults to 1.0.
WO - Mean emission energy (in MeV)
DWO - Energy spread (in MeV)
DWT - Clock step size, same as DWT on START card (Degrees)
HBX - Hammersley sequence base for X dist. generation (choose: 1)
HBY - Hammersley sequence base for Y dist. generation (choose: 2)
HBPHI - Hammersley sequence base for PHASE dist. gen. (choose: 3 for Gaussian, -3 for uniform distribution. Choosing a negative base number overrides the SIGMAT setting above)
HBWO - Hammersley sequence base for ENERGY dist. gen. (choose: 5)

Notes: In general, the base choices suggested here {1,2,3,5} can be substituted by any four prime numbers. Higher base numbers result in larger scale structure (i.e. "stripes" in the distribution) and should be avoided. DO NOT use the same base number twice, or a perfectly correlated distribution in the two parameters will result.

ASPECT - Ratio of vertical to horizontal beam dimensions.
(assumed 1 if omitted) If ASPECT < 0, then beam cross section is rectangular, not elliptic.
XOFF - X distance to displace laser spot from symmetry axis, [cm]
YOFF - Y distance to displace laser spot from symmetry axis, [cm]
ANOISE - Amplitude of intensity noise on laser pulse. (0.0 to 1.0)
FNOISE - Harmonic number of intensity noise modulation applied to laser (eq. if FNOISE=10, the intensity modulation will have ten periods over the bunch length.
TANOISE - Amplitude of transverse intensity noise on laser pulse. (0.0 to 1.0)
(This option induces sinusoidal variations in the x-directions)
TFNOISE - Harmonic number of transverse intensity noise modulation applied to laser (eq. if TFNOISE=10, the intensity modulation will have ten periods over the bunch length.)

MAKERS
A restart file of the type generated on receiving signal 15 (SIGTERM) is generated when this card is reached, but unlike the SAVE command, execution continues.

NOWAKES CHARGE
Added anywhere before the START card, this card will suppress any wakefields (all WAKE cards will be ignored), unless the optional argument CHARGE is supplied.

**CHARGE** - Number of fundamental charges in bunch (total) = \( \frac{Q_{\text{tot}}}{e} \). Used when diagnosing wakefield effects with space charge off. This value will override the value set on the SCHEFF card(s).

**POISSON ZOFFSET RMULT [ZMAPL ZMAPH] FILENAME**

This card causes a POISSON field map to be read into the static B-field arrays RBFLD and ZBFLD with an offset in z given by ZOFFSET, and with a scale factor RMULT applied to every field component value. [rmult defaults to 1.0, zoffset defaults to 0.0] First POISSON card must carry start and stop coordinates of map ZMAPL, ZMAPH.

The program POI2PAR is available to read POISSON output files and interpolate onto the standard PARMELA mesh.

**RFSETUP**

This card, when placed on the first or second line of the input file sets the number of particles to 2 and disables space charge calculations to allow rapid RF phase adjustment runs without the need to modify the SCHEFF or INPUT cards.

**SCHEFF** BEAMI DRMESH DZMESH NR NZ NIP PL OPT REMESH RWAL POINT SOL NCH RMACRO

**BEAMI** - Beam current in amps if BEAMI > 0, in number of charges otherwise

**RMESH** - Initial mesh radius [cm] (Defaults to 5.0 cm)

**ZMESH** - Initial mesh length [cm] (total, not half length of mesh) (Defaults to 5.0 cm)

**NR** - Starting number of mesh points in radial direction. (Max: 40, default: +20) If NR is negative, remeshing will only change the mesh dimensions, not the number of mesh points.

**NZ** - Starting number of mesh points in longitudinal direction. (Max: 80, default: +20) If NZ is negative, remeshing will only change the
mesh dimensions, not the number of mesh points.

NIP - Number of adjacent bunches (typically just set to zero)
PL - Distance between adjacent bunches
(also simply zero for single bunch case)
OPT - Number of rings to use in Gaussian quadrature (typically set = 0)
If OPT>0, the BNL mesh method is used, if OPT<0, the old method is used. If OPT=4 or 5, then image charges on a metallic cathode are included. If OPT is an even number, binning occurs in equal volume rings, if odd, in equal thickness rings.

REMESH - Remeshing criterion: REMESH > 0 => remesh when either the rms beam radius or rms beam length changes by a factor of REMESH.
REMESH < 0 => remesh when either the maximum beam radius or maximum beam length changes by a factor of REMESH.
(Defaults to 0.05 implying a remesh when rms Rbeam or rms Zbeam changes by 5 percent.)
REMESH = 0 => never remesh.

RWAL - Radius of conducting wall. (=0 means no wall)

POINT - Radius of Cloud of charge used in Point-by-Point method
If point is negative, images on cathode are included in calc.
When using ellipsoidal macroparticles, this is the distance expressed as a multiple of the longest semi-major axis length of the ellipsoids at which the fields are assumed to become the same as those of spherical charge distribution. Defaults to 3.

SOL - Point-by-point algorithm: 0=uses spheres, 1=uses "rods" composed of NCH discrete charges of charge Q/NCH 4=use Carlsten line-by-point method with retardation effects
RZ-mesh method algorithm: 0=use LANL image charge method, 1=use LANL/UCLA hybrid method

NCH - Number of charges to subdivide each charge into (obsolete)

RMACRO - Size of macroparticle, expressed as a multiple of the of the corresponding interparticle spacing. Defaults to 2.

START PHIO DWT NSTEPS NSC NOUT NSLICE NSPT

PHIO - Launch phase of bunch w.r.t. RF [Degrees] (If set to -99, program will use gradient of first RF cell and Kim’s criteria to calculate the proper phase [Kim’s criteria yields two roots
DWT  - Step size in degrees
NSTEPS  - Total number of time steps to take
NSC  - Mod on timestep for space charge calculation
NOUT  - Mod on timestep for output of RMS beam quantities; controls also
                   the timesteps when slice emittances and trace files can be written.
NSLICE  - Mod on timestep for output of slice emittance data
NSPT  - Number of slices (even!) to take at each sample time
          (Default is 14 slices)

FWHME  XCUT1 YCUT1 ZCUT1 XCUT2 YCUT2 ZCUT2 ...

XCUTn  - Percentage of particles to include in horizontal FWHM emittance
          calculation. Halo is symmetrically cut off using this criterion.
          Specify up to five triplets of values, ending with a -999.
YCUTn  - Percentage of particles to include in vertical FWHM emittance
          calculation.
ZCUTn  - Percentage of particles to include in horizontal FWHM emittance
          calculation.

NOTE: The third set of cut values (XCUT3 YCUT3 ZCUT3) are used
      for the slice emittance calculation as well. (if nslice!=0)

          Default: calculate 90%, 80% and 70% FWHM emittances.

CONTINUE  DWT NSTEPS NSC NOUT NSLICE

DWT  - Step size in degrees
NSTEPS  - Total number of time steps to take
NSC  - Mod on timestep for space charge calculation
NOUT  - Mod on timestep for output of RMS beam quantities
NSLICE  - Mod on timestep for output of slice emittance data

DFILES  TYPE1 NTYPE1 PAR1 PAR2 PAR3 ... TYPE2 NTYPE2 PAR1 PAR2 PAR3 ... TYPE3
          NTYPE3 PAR1 PAR2 PAR3 ... -999

          The DFILES card should follow right after the RUN card to
work properly.

**TYPEn** - Type code for file to produce:

<table>
<thead>
<tr>
<th>Type Code</th>
<th>Description</th>
<th>Parameters</th>
<th>Filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Space charge impulse file</td>
<td>Timestep number(s)</td>
<td>scrz_d</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sckm_d</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sccc_d</td>
</tr>
<tr>
<td>2</td>
<td>Phase space dump</td>
<td>Timestep number(s)</td>
<td>psd_d</td>
</tr>
<tr>
<td>3</td>
<td>RZ mesh size</td>
<td>none</td>
<td>remesh_d</td>
</tr>
<tr>
<td>4</td>
<td>$\beta(z), \gamma(z)$</td>
<td>none</td>
<td>betag_d</td>
</tr>
<tr>
<td>5</td>
<td>Initial phase space</td>
<td>none</td>
<td>initps_d</td>
</tr>
<tr>
<td>6</td>
<td>E,B fields in RF cell</td>
<td>Cell number</td>
<td>rfc_h_d</td>
</tr>
<tr>
<td>7</td>
<td>RF cell impulse dump</td>
<td>Timestep number(s)</td>
<td>rfi_d</td>
</tr>
<tr>
<td>8</td>
<td>Static B-field map $B(r,z)$</td>
<td>none</td>
<td>bzmap2_d</td>
</tr>
</tbody>
</table>

**NTYPEn** - Total number of files of type TYPEn to produce (Limit: 10)

**PAR1...PARn** - Parameter values for plots. (e.g. timestep numbers when phase space dumps are done.)

-999 - The input line is of variable length, so the -999 terminator must be included

**RUN IRUN IP FREQ Z0 W0 LTYPE INTERP THRESH**

**IRUN** - User defined run number (no internal significance)

**IP** - Print status

**FREQ** - Global frequency declaration; is overridden by local frequency declarations by any element [MHz]

**Z0** - Staring z coordinate for reference particle (set to 0 for photocathode sources [cm]

**W0** - Energy of reference particle [MeV]

**LTYPE** - Linac type:
1 => disk-and-washer structure
2 => side-coupled cavity
3 => RTM side-coupled cavity

**INTERP** - Interpolation order for E,B fields: 0-NONE, 1-linear, 2-quadratic, 3-cubic. [Defaults to linear interpolation]

**THRESH** - Acceptable particle loss threshold. If the number of particles remaining in the simulation drops below this percentage of the
starting value, the simulation will stop.
(Default=0.50, or 50% particles must stay 'alive' to continue)

TRACE X,X',Y,Y',Φ,E

The trace card is used to introduce a particle that will have
a complete 6-dimensional history written to a file for subsequent
analysis. Coordinate dumps occur on the same timesteps that output
occurs. (See the START and CONTINUE cards for details.) This card
works only with input type 11. (Photocathode with quiet start)

Up to 15 particles may be specified. The file(s) will
be ptra___d with the particle number being the suffix of the
file name. The coordinates and energy must be NORMALIZED to the
parameters of the rest of the distribution, while the
divergences are specified in milliradians directly.

To trace a particle at the horizontal edge of the beam in the
y=0 plane that is centered longitudinally, which has no
transverse momentum, and the same energy as the reference
particle, enter:
TRACE 1 0 0 0 0 1

The trace card(s) must precede the INPUT card.

VERBOSE

Putting this card on the first line (ahead of the title card)
causes Parmela to print information about the current task
on the screen. This has limited usefulness in debugging, and
should not be used in non-interactive jobs.

WAKE DUMPFLAG XOFFSET YOFFSET TSCALEF LSCALEF FILENAME

The WAKE card allows for an after-the-fact kick to be applied
to the beam using data from the file FILENAME. In principle, it
can be used to model any perturbative force with a dependence on
the z-coordinate alone. All wake elements may be disabled by
adding a NOWAKES card to the start of the input deck.

DUMPFLAG
1-Dump map of wakefield kicks as seen by beam
0-no dump [the default] Dump file is named wakeNN.d,
with NN defined as the element number.

XOFFSET
Amount to offset the beam horizontally on entering the structure
in [cm].

YOFFSET
Amount to offset the beam vertically on entering the structure
in [cm].

TSCALEFAC
Scale transverse wake kicks by TSCALEFAC times the values loaded
from the ABCI file. Defaults to +1.0 .

LSCALEFAC
Scale longitudinal wake kicks by TSCALEFAC times the values
loaded from the ABCI file. Defaults to +1.0 .

FILENAME
Name of file containing digested ABCI data. Format expected is:

line 1: Comments (column labels)
line 2: Comments (column units)
line 3: Z W0par_1 W1perp_1 W1par_1
    ...
line N: Z W0par_N W1perp_N W1par_N

Where the columns are:
Z , the ABCI Z-coordinate in [m]
W0par , the monopole longitudinal wake in [V/pC]
W1perp , the dipole transverse wake in [V/pC/m]
W1par , the dipole longitudinal wake in [V/pC/m^2]
Tips for Debugging a PARMELA Input Deck

In the text that follows, each complete PARMELA command line, such as the command to start a PARMELA run:

```
RUN 1 3 2856.0 0.0 1.0E-06 1 2 0.80
```

is called a "card" and is composed of a label ("RUN") and a mixture of required and optional parameters.

1. Make sure to observe the required ordering for the cards. In addition to the actual beamline description cards, which must be in order from the first element to the last, there are several other card orderings to observe. These are summarized below in an abbreviated notation with (card1,card2,card3,...) meaning that card1 must precede card2, which in turn must precede card3, and so on. (run,dfiles); (run,restart); (laser,input); (trace,input); ({all cards except save, end, continue}start)

2. Be sure to set values to all "required" parameters, observing their order of listing on the input line. Optional parameters which fall on the input line between required parameters (such as occurs for the SCHEFF card) must be entered to preserve the ordering and placement of the values on the input line. It is a good idea to declare values for optional parameters, even if the default value is acceptable, to remind you in the future of all the parameters that control your run.

3. All cards must end with a carriage return. Parameters entries which wrap onto subsequent lines should have only one carriage return at the end of the list.

4. Many cards are scanned for minor errors, and diagnostic messages will be displayed telling you briefly what the error is, and suggesting corrective action.

5. Make sure no characters except "e" or "E" which flag the exponent of a real value, occur after the card label. If you wish to enter comments into the input deck, enter them following the "END" card.

6. Input all parameters, unless otherwise specifically instructed, in cgsa units. Some exceptions to this rule include: MV/m for RF cavity peak fields, picoseconds for bunch lengths, MeV for beam energy and energy spread, etc.

7. Make extensive use of the DFILES diagnostic file output to find the cause of unphysical simulations. Consult the "Modification to PARMELA, Revisited" guide for information on using this card.
Running Unix Poisson/Superfish
And UCLA Parmela on Guiness

This technote is meant to provide tips and tricks on the use of the unix version of
Poisson and Superfish implemented on guiness. It is not meant to be an
introduction to the codes themselves, but to show the user some (sometimes-
undocumented) relevant features.

Poisson

1. **Input decks**
The input deck for unix Poisson is virtually the same as that of the PC version
of the code. There are a few important exceptions to keep in mind.

1. Not all CONs can be input with the input deck. The most notable are:
   - CON(19)=ICYL tells Poisson to use cylindrical symmetry
   - CON(21)-CON(24) boundary conditions on the upper, lower, right, and
     left boundaries of the problem.

   • To indicate a Poisson problem instead of a Superfish problem, the first
     character in the input deck must be a space (for Superfish problems the
     first character is a 1).

   • Be careful importing input decks from PC files. The non-printing ^M end of
     line characters can confuse the programs and give unpredictable results.

2. **Running the codes**
The codes must be run in this order: automesh, lattice, poisson, and psfplot
(optional).

   1. Automesh: straightforward to run. Asks for input file and produces outputs
      outaut and tape73 used by lattice.

   2. Lattice: again straightforward. When prompted for CONs, input in this
      fashion: *con# value *con# value ... s

      Where s denotes the end of con inputs.

      Example: *19 1 *21 0 0 0 0 s

   • Poisson: similar to automesh and lattice. Usually no CON inputs are
     required. When asked which “Dump number” to use, answer 0. The
     output file outpoi will contain CON values, run information and a listing of
     field values at certain r and z locations. The CONs to control which field
     values are produced are CON(42)-CON(45). The first two control the start
     and stop logical coordinates in r and the last two in z. To produce an
     outpoi file suitable for input into Parmela, set these values to KMIN=TOP=1
     LMIN=1 LTOP=LMAX where LMAX is given by CON(3) in any of the output
     files.
The following is an example of a Poisson run used to produce a solenoid field map for the Parmela bzmap input card.

Trying 128.97.23.154...
Connected to guiness.
Escape character is '^]'.

Red Hat Linux release 5.1 (Manhattan)
Kernel 2.0.34 on guiness
login: anderson
Password:
Last login: Thu Oct 22 21:01:10 from guiness
You have mail.
9:07pm up 2 days, 11:50, 4 users, load average: 0.07, 0.03, 0.01
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
anderson tty1 5:38pm 3:28m 1.31s 0.01s sh /usr/X11R6/bin/startx
anderson ttyp0 :0.0 8:29pm 0.00s 0.10s 0.01s telnet guiness
pietro ttyp2 stout 8:00pm 51:06 0.58s 0.50s emacs scharge.c
anderson ttypl guiness 9:07pm 0.00s 0.11s 0.03s w

guiness ~ [1] cd sftest/exercise/
guiness ~/sftest/exercise [2] cat sol

neptune solenoid
$reg nreg=3,xmax=20.00,ymax=50.00,ktop=5,ltop=50,dx=0.25,dy=0.25,npoint=5$
$po x=0.000,y=0.000$
$po x=20.000,y=0.000$
$po x=20.000,y=50.000$
$po x=0.000,y=50.000$
$po x=0.000,y=0.000$
$reg mat=1, cur=30000, npoint=5$
$po x=5.0,y=11.35$
$po x=14.6,y=11.35$
$po x=14.6,y=18.72$
$po x=5.0,y=18.72$
$po x=5.0,y=11.35$
$reg mat=2, npoint=9$
$po x=8.89,y=11.35$
$po x=14.6,y=11.35$
$po x=14.6,y=18.72$
$po x=5.0,y=18.72$
$po x=5.0,y=21.26$
$po x=17.78,y=21.26$
$po x=17.78,y=8.81$
$po x=8.89,y=8.81$
$po x=8.89,y=11.35$

*******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California  *
*                                                                             *
*******************************************************************************
*                                                                             *
*   Neither the Government nor the University makes any warranty, expressed  *
* or implied, or assumes any liability or responsibility for use of this     *
* software.                                                                  *
*******************************************************************************

?type input file name

sol

REGION NO. 1
OK

REGION NO. 2
OK
REGION NO.  3
OK
guiness ~/sftest/exercise [4] lattice
******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California *
******************************************************************************
*                                                                             *
*   Neither the Government nor the University makes any warranty, expressed   *
*   or implied, or assumes any liability or responsibility for use of this    *
*   software.                                                                 *
******************************************************************************

?type input file name (s for default)
s
BEGINNING OF LATTICE EXECUTION
DUMP 0 WILL BE SET UP FOR POISSON
  neptune solenoid

?TYPE INPUT VALUES FOR CON(?)
*19 1 *21 0 0 0 0 s

ELAPSED TIME  =  0.2 SEC.
0ITERATION CONVERGED

ELAPSED TIME  =  0.4 SEC.

GENERATION COMPLETED
DUMP NUMBER 0 HAS BEEN WRITTEN ON TAPE35.

DUMP NUMBER 0 HAS BEEN WRITTEN ON TAPE36.
guiness ~/sftest/exercise [5] cat outlat | grep LMAX
  (  3) =        201, LMAX

DUMP NUMBER 0 HAS BEEN WRITTEN ON TAPE36.
guiness ~/sftest/exercise [6] poisson
******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California *
******************************************************************************
*                                                                             *
*   Neither the Government nor the University makes any warranty, expressed   *
*   or implied, or assumes any liability or responsibility for use of this    *
*   software.                                                                 *
******************************************************************************

?type "tty" or input file name 
tty

?TYPE INPUT VALUE FOR DUMP NUM
0
BEGINNING OF POISSON EXECUTION FROM DUMP NUMBER   0
  PROB. NAME = neptune solenoid

?TYPE INPUT VALUES FOR CON(?)
*42 1 1 1 201 s

ELAPSED TIME  =  0.2 SEC.
0  CYCLE  AMIN  AMAX  RESIDUAL-AIR  ETA-AIR  RHOAIR  XJFACT
0  0  0.0000E+00  0.0000E+00  1.0000E+00  1.0000  1.9000  1.0000
SOLUTION CONVERGED IN 350 ITERATIONS

ELAPSED TIME = 1.2 SEC.

DUMP NUMBER 1 HAS BEEN WRITTEN ON TAPE35.

DUMP NUMBER 1 HAS BEEN WRITTEN ON TAPE36.

?TYPE INPUT VALUE FOR DUMP NUM

-I

THIS CODE MAKES A POSTSCRIPT OUTPUT FILE, PLOT.PS

?TYPE INPUT DATA NUM, ITRI, NPHI, INAP, NSWXY,

1 0 40 s

INPUT DATA
NUM= 1 ITRI= 0 NPHI= 40 INAP= 0 NSWXY= 0

PLOTTING PROB. NAME = neptune solenoid CYCLE = 350

?TYPE INPUT DATA XMIN, XMAX, YMIN, YMAX,
s

INPUT DATA
XMIN= 0.000 XMAX= 20.000 YMIN= 0.000 YMAX= 50.000

?TYPE GO OR NO

go

?TYPE INPUT DATA NUM, ITRI, NPHI, INAP, NSWXY,

-I

This code makes a postscript output file, plot.ps

\texttt{guiness ~/sftest/exercise [7] psfplot}

\texttt{********************** POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12) **********************}
\texttt{* Copyright (c) 1965,1991,1992,1993 Regents of the University of California *}
\texttt{*******************************************************************************}
\texttt{THIS CODE MAKES A POSTSCRIPT OUTPUT FILE, PLOT.PS}

\texttt{guiness ~/sftest/exercise [8] ls -l plot.ps}
\texttt{-rw-r----- 1 anderson users 148307 Oct 22 21:09 plot.ps}
\texttt{guiness ~/sftest/exercise [9] lpr plot.ps}
\texttt{guiness ~/sftest/exercise [10] logout}
Superfish

1. Input decks
   Superfish input decks have all the same rules as Poisson input decks, except that the first character in the file must be 1.

2. Running the codes
   Just as with Poisson runs, execute automesh, lattice, superfish, psfplot. All the CONs that were important for Poisson are the same for Superfish. When running Superfish CON(65) must be specified. This is the initial guess of the cavity resonant frequency.

Here is an example of a typical Superfish run.

```
anderson
Password: 
Last login: Fri Oct 23 09:24:08 from murphys
You have mail.
  9:27am  up 3 days, 10 min,  5 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
pietro   tty1                       9:05am 21:37  0.71s  0.01s  sh /usr/X11R6/b
pietro   ttyt0 :0.0            9:06am  0.00s  0.19s  0.14s  pine
pietro   ttyt2 :0.0            9:25am  2:01  0.65s  0.62s  emacs scharge.c
anderson ttyt1  murphys        9:24am  0.11s  0.06s  scharge.c
anderson ttyt3  murphys        9:27am  0.11s  0.03s  w

pietro   [1] cd sftest/exercise/
pietro   [2] cat ucla_bgu
superfish brookhaven gun
&reg xmax=15.0,ymax=4.2082,dx=0.04,npoint=18 &
&PO X=0.000,Y=0.000 &
&PO X=0.000,Y=4.1587 &
&PO X=2.27584,Y=4.1587 &
&PO X=2.27584,Y=2.20218 &
&PO NT=2,X0=3.22834,Y0=2.20218,R=0.9525,THETA=270.0 &
&PO X=3.2766,Y=1.24968 &
&PO X=3.3782,Y=1.24968 &
&PO X=3.4798,Y=1.24968 &
&PO X=3.52806,Y=1.24968 &
&PO NT=2,X0=3.52806,Y0=2.20218,R=0.9525,THETA=0.0 &
&PO X=4.48056,Y=4.2082 &
&PO X=7.73000,Y=4.2082 &
&PO X=7.73000,Y=2.20218 &
&PO NT=2,X0=8.68250,Y0=2.20218,R=0.9525,THETA=270.0 &
&PO X=9.62406,Y=1.24968 &
&PO X=15.0000,Y=1.24968 &
&PO X=15.0000,Y=0.0000 &
&PO X=0.0000,Y=0.0000 &

pietro   [3] automesh
*******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California *
*******************************************************************************
*                                                                             *
*   Neither the Government nor the University makes any warranty, expressed   *
*   or implied, or assumes any liability or responsibility for use of this    *
*   software.                                                                 *
*******************************************************************************

?type input file name
ucla_bgu
```
REGION NO. 1
LOGICAL BOUNDARY SEGMENT END POINTS

<table>
<thead>
<tr>
<th>ISEG</th>
<th>KB</th>
<th>LB</th>
<th>KD</th>
<th>LD</th>
<th>KE</th>
<th>LE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>121</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>121</td>
<td>1</td>
<td>0</td>
<td>58</td>
<td>121</td>
</tr>
<tr>
<td>3</td>
<td>58</td>
<td>121</td>
<td>0</td>
<td>-1</td>
<td>58</td>
<td>64</td>
</tr>
<tr>
<td>4</td>
<td>58</td>
<td>64</td>
<td>0</td>
<td>-1</td>
<td>81</td>
<td>37</td>
</tr>
<tr>
<td>5</td>
<td>81</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>83</td>
<td>37</td>
</tr>
<tr>
<td>6</td>
<td>83</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>85</td>
<td>37</td>
</tr>
<tr>
<td>7</td>
<td>85</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>88</td>
<td>37</td>
</tr>
<tr>
<td>8</td>
<td>88</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>89</td>
<td>37</td>
</tr>
<tr>
<td>9</td>
<td>89</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>113</td>
<td>64</td>
</tr>
<tr>
<td>10</td>
<td>113</td>
<td>64</td>
<td>0</td>
<td>1</td>
<td>113</td>
<td>122</td>
</tr>
<tr>
<td>11</td>
<td>113</td>
<td>122</td>
<td>1</td>
<td>0</td>
<td>195</td>
<td>122</td>
</tr>
<tr>
<td>12</td>
<td>195</td>
<td>122</td>
<td>-1</td>
<td>-1</td>
<td>195</td>
<td>64</td>
</tr>
<tr>
<td>13</td>
<td>195</td>
<td>64</td>
<td>-1</td>
<td>-1</td>
<td>218</td>
<td>37</td>
</tr>
<tr>
<td>14</td>
<td>218</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>241</td>
<td>37</td>
</tr>
<tr>
<td>15</td>
<td>241</td>
<td>37</td>
<td>1</td>
<td>0</td>
<td>376</td>
<td>37</td>
</tr>
<tr>
<td>16</td>
<td>376</td>
<td>37</td>
<td>0</td>
<td>-1</td>
<td>376</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>376</td>
<td>1</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

--- WARNING --- USER PROVIDED DRIVE POINT COULD BE BETTER THAN THE DEFAULT BEST GUESS DRIVE POINT.

guiness ~/sftest/exercise [4] lattice

*******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California  *
*******************************************************************************

?type input file name (s for default)
s

BEGINNING OF LATTICE EXECUTION

DUMP 0 WILL BE SET UP FOR SUPERFIS

lsuperfish brookhaven gun

?type input VALUES FOR CON(?)
*19 1 *21 1 0 0 1 s

ELAPSED TIME = 0.6 SEC.
ITERATION CONVERGED

ELAPSED TIME = 0.5 SEC.

GENERATION COMPLETED

DUMP NUMBER 0 HAS BEEN WRITTEN ON TAPE35.

DUMP NUMBER 0 HAS BEEN WRITTEN ON TAPE36.
guiness ~/sftest/exercise [5] superfish

*******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California  *
*******************************************************************************

*
?type "tty" or input file name

tty

?TYPE INPUT VALUE FOR DUMP NUM

0

BEGINNING OF SUPERFISH EXECUTION FROM DUMP NUMBER 0

PROB. NAME = superfish brookhaven gun

?TYPE INPUT VALUES FOR CON(?)

*65 2856. s

ELAPSED TIME = 0.9 SEC.

<table>
<thead>
<tr>
<th>CYCLE</th>
<th>DEL(k**2)</th>
<th>FREQ[MHz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-3.4622E-05</td>
<td>2855.86206</td>
</tr>
<tr>
<td>2</td>
<td>1.9968E-06</td>
<td>2855.86987</td>
</tr>
<tr>
<td>3</td>
<td>-2.1473E-10</td>
<td>2855.86987</td>
</tr>
</tbody>
</table>

SOLUTION CONVERGED IN 3 ITERATIONS

ELAPSED TIME = 0.1 SEC.

DUMP NUMBER 1 HAS BEENWRITTEN.

?TYPE INPUT VALUE FOR DUMP NUM

-1

guiness ~/sftest/exercise [6] psfplot

*******************************************************************************
* POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)                           *
* Copyright (c) 1965,1991,1992,1993 Regents of the University of California *
*******************************************************************************
* Neither the Government nor the University makes any warranty, expressed    *
* or implied, or assumes any liability or responsibility for use of this    *
* software.                                                                 *
*******************************************************************************

THIS CODE MAKES A POSTSCRIPT OUTPUT FILE, PLOT.PS

?TYPE INPUT DATA- NUM, ITRI, NPHI, INAP, NSWXY,

1 0 35 s

INPUT DATA
NUM= 1 ITRI= 0 NPHI= 35 INAP= 0 NSWXY= 0

PLOTTING PROB. NAME = superfish brookhaven gun
CYCLE = 3

?TYPE INPUT DATA- XMIN, XMAX, YMIN, YMAX,

s

INPUT DATA
XMIN= 0.000 XMAX= 15.000 YMIN= 0.000 YMAX= 4.208

ARROW PLOT?: ENTER Y FOR YES OR N FOR NO

n

?TYPE GO OR NO

go

?TYPE INPUT DATA- NUM, ITRI, NPHI, INAP, NSWXY,

-1 s

Creating Parmela RF Field Maps from Superfish Output

Once you have produced the tape35 and tape36 files from the process described above, you can extract the information about the fields and use it as Parmela input. This is the procedure to follow:

1. Run SFO1
   The primary purpose of SFO1 is to renormalize the fields in the binary output files to some reasonable value. Unfortunately, it renormalizes so that the average axial $E_z$ is 1.00 MV/m. Parmela expects the input file to be normalized so that the peak axial field is 1.00 MV/m. I will indicate how one can compensate for this below. Running SFO1 is straightforward. No new CONs need to be entered. Take note of the value for CON(74) ASCALE that the output specifies.

2. Run Shy
   The shy command is the unix equivalent to running sf7 with the tape command in the PC codes. It produces the file outshy, which contains field values in a range of (r,z) specified by the input CONs 54-57. Also, be sure to specify CON(74) (this is the AScale value computed by sfo1). At this point one could enter a value for CON(74) which produces the correct value of 1.00 MV/m peak axial field, either by intelligent guessing or by an iterative process.

3. Run Fish2par
   Fish2par takes the outshy file produced above and creates a binary file that Parmela can read as the input for a ccell card. Running is again straightforward. Take note of the parameters XAPER, YAPER, and SYM that fish2par produces and use these values in the Parmela ccell card.

4. Repeat the process of running Shy and Fish2par for each halfcell you want to have input fields for.

Below is an example of producing Parmela input files for each half cell of the 1.625 cell gun simulated in the Superfish run shown above.
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
anderson tty1                       1:25pm 24:26   0.90s  0.02s  sh /usr/X11R6/b
anderson ttyp0 :0.0              1:26pm 2:26   0.12s  0.12s  bash
anderson tty1 :0.0              1:42pm 7:24   0.08s  0.01s  vi parmin.full
anderson ttyp2 murphys           1:50pm 0.00s  0.10s  0.03s  w

 Usuario anderson en tty1...

Ficheros... 

 Shell...

POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)
Copyright (c) 1965,1991,1992,1993 Regents of the University of California

Neither the Government nor the University makes any warranty, expressed
or implied, or assumes any liability or responsibility for use of this
software.

?type "tty" or input file name
	tty

?TYPE INPUT VALUE FOR DUMP NUM
	1
BEGINNING OF SFO1 EXECUTION FROM DUMP NUMBER 1
PROB. NAME = superfish brookhaven gun

?TYPE INPUT VALUES FOR CON(?)
	s

?TYPE GO FOR OUTPUT SUMMARY AT TERMINAL

go

Total number of points =   46872  NCELL =  1  ZCTR =       0.0000 cm
LINT =  1  Average r =   0.0348 cm  ZBEG =   0.0000 cm  ZEND =  15.0000 cm
Integral [Ez(z,r=    0.03 cm)sin(Kz)dz] =    -72358.83 V
Integral [Ez(z,r=    0.03 cm)cos(Kz)dz] =     30435.78 V

Problem name =superfish brookhaven gun

SUPERFISH calculates the frequency [f] to at most six place accuracy
depending on the input mesh spacing.

Full cavity length [2L]  =   30.0000 cm         Diameter =    8.4164 cm
Mesh problem length [L]  =   15.0000 cm
Full drift-tube gap [2g] =    0.0000 cm
Frequency [f] (starting value =2856.000) =               2855.869873 MHz
Eo normalization factor (CON(74)=ASCALE) for  1.000 MV/m =   13774.5
Stored energy [U] for mesh problem only =                    2.79097 mJ
Power dissipation [P] for mesh problem only =                   0.00 W
Magnetic field on outer wall =                                     0 A/m
Hmax for wall and stem segments at z=  0.00,r=  0.00 cm =          0 A/m
Emax for wall and stem segments at z=  0.00,r=  0.00 cm =      0.000 MV/m

Beta        T        Tp       S        Sp       g/L         Z/L
2.85784650  0.20291 -0.11692 -0.48239 -0.01017  0.000000    0.000000

?TYPE INPUT VALUE FOR DUMP NUM
	-1
usbiness ~/sftest/exercise/ [3] shy
		
Neither the Government nor the University makes any warranty, expressed or implied, or assumes any liability or responsibility for use of this software.

*******************************************************************************

?type "tty" or input file name
tty

?TYPE INPUT VALUE FOR DUMP NUM
1
1
BEGINNING OF SHY EXECUTION FROM DUMP NUMBER 1
PROB. NAME = superfish brookhaven gun

?TYPE INPUT VALUES FOR CON(7)
*74 13774.5 *54 0.00 3.3782 0.00 1.25 s

?TYPE INPUT VALUE FOR DUMP NUM
-1

guiness ~/sftest/exercise [4] cp outshy halfcell

guiness ~/sftest/exercise [5] fish2par

*SuperFish field map to PARMELA field map converter*

Enter the Superfish/SHY field map filename: halfcell
Enter 1 for upstream half of cavity, -1 for downstream half: -1

Mesh dimensions:
Logical Nr x Nz : 37 x 85
Physical r x z : 1.25 x 3.37800002
Total mesh points: 3145

Loading fields...

Done reading in Superfish/SHY file.

Enter dipole offset of field in Y-direction [cm]: 0.00

PARMELA mesh: 39 x 39 x 99
Total points: 150579

Mapped field region: (In PARMELA coordinates)
Xmin= -1.1875 Xmax= 1.1875 Xstep= 0.0625
Ymin= -1.1875 Ymax= 1.1875 Ystep= 0.0625
Zmin= 0. Zmax= 3.37800002 Zstep= 0.0344693884

Make sure CCELL card entries are:
XAPER = 1.1875
YAPER = 1.1875
SYM = -1

Beginning interpolation of data, and construction of field map files...

Percent completed/to go: 10 20 30 40 50 60 70 80 90

POISSON to PARMELA field map conversion complete.
Input file: halfcell

>>> Binary output file: halfcell.bfm
>>> ASCII output file : halfcell.asc

Enter the name of the binary field map on the line following the CCELL card, complete with the .bfm extension. The ASCII version of the file should be used for verifying the interpolation only, and then
deleted. (It requires 13.6 MB of storage)

guiness ~/sftest/exercise [6] rm halfcell.asc

******************************************************************************
*   POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)                       *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California *
******************************************************************************

* Neither the Government nor the University makes any warranty, expressed *
* or implied, or assumes any liability or responsibility for use of this *
* software.                                                              *

?type "tty" or input file name

tty

?TYPE INPUT VALUE FOR DUMP NUM
1
1
BEGINNING OF SHY EXECUTION FROM DUMP NUMBER 1
PROB. NAME = superfish brookhaven gun

?TYPE INPUT VALUES FOR CON(?)
*74 13774.5 *54 3.3782 6.1052 0.00 1.25 s

?TYPE INPUT VALUE FOR DUMP NUM
-1

guiness ~/sftest/exercise [8] mv outshy upfull

guiness ~/sftest/exercise [9] fish2par

*SuperFish field map to PARMELA field map converter*
Enter the Superfish/SHY field map filename: upfull
Enter 1 for upstream half of cavity, -1 for downstream half: 1

Mesh dimensions:
Logical Nr x Nz : 37 x 70
Physical r x z  :  1.25 x  2.727
Total mesh points: 2590

Loading fields...
25 -0.640380025 0.

Done reading in Superfish/SHY file.
Enter dipole offset of field in Y-direction [cm]: 0.00

PARMELA mesh: 39 x 39 x 99
Total points: 150579

Mapped field region: (In PARMELA coordinates)
Xmin= -1.1875 Xmax=  1.1875 Xstep=  0.0625
Ymin= -1.1875 Ymax=  1.1875 Ystep=  0.0625
Zmin=  0. Zmax=  2.727 Zstep=  0.0278265309

Make sure CCELL card entries are:
XAPER =  1.1875
YAPER =  1.1875
SYM = 1

Beginning interpolation of data, and construction of field map files...
Percent completed/to go:  90  80  70  60  50  40  30  20  10
POISSON to PARMELA field map conversion complete.
Input file:  upfull

>>> Binary output file:  upfull.bfm
>>> ASCII output file :  upfull.asc

Enter the name of the binary field map on the line
following the CCELL card, complete with the .bfm
extension. The ASCII version of the file should be
used for verifying the interpolation only, and then
deleted. (It requires 13.6 MB of storage)

guiness ~/sftest/exercise [10] rm upfull.asc

*******************************************************************************
*               POISSON/SUPERFISH GROUP OF CODES (RELEASE 4.12)               *
*  Copyright (c) 1965,1991,1992,1993 Regents of the University of California  *
*******************************************************************************
*                                                                             *
*   Neither the Government nor the University makes any warranty, expressed   *
*   or implied, or assumes any liability or responsibility for use of this    *
*   software.                                                                 *
*******************************************************************************

?type "tty" or input file name
tty

?type INPUT VALUE FOR DUMP NUM
1
1
BEGINNING OF SHY EXECUTION FROM DUMP NUMBER   1
PROB. NAME = superfish brookhaven gun

?type INPUT VALUES FOR CON(?)
*74 13774.5 *54 6.1052 9.5005 0.00 1.25 s

?type INPUT VALUE FOR DUMP NUM
-1

*SuperFish field map to PARMELA field map converter*

Enter the Superfish/SHY field map filename: outshy
Enter 1 for upstream half of cavity, -1 for downstream half:  -1

Mesh dimensions:
Logical Nr x Nz :  37 x 85
Physical r x z  :   1.25 x   3.39499998
Total mesh points:  3145

Loading fields...

Done reading in Superfish/SHY file.

Enter dipole offset of field in Y-direction [cm]:
0.00

PARMELA mesh:  39 x  39 x  99
Total points:  150579

Mapped field region: (In PARMELA coordinates)
Ymin=  -1.1875 Xmax=   1.1875 Xstep=  0.0625
Ymin=  -1.1875 Ymax=   1.1875 Ystep=  0.0625
Zmin=   0. Zmax=   3.39499998 Zstep=  0.0346428566
Make sure CCELL card entries are:
XAPER = 1.1875
YAPER = 1.1875
SYM = -1

Beginning interpolation of data, and construction of field map files...

Percent completed/to go: 10 20 30 40 50 60 70 80 90
POISSON to PARMELA field map conversion complete.
Input file: outshy

>>> Binary output file: outshy.bfm
>>> ASCII output file: outshy.asc

Enter the name of the binary field map on the line following the CCELL card, complete with the .bfm extension. The ASCII version of the file should be used for verifying the interpolation only, and then deleted. (It requires 13.6 MB of storage)

guiness ~/sftest/exercise [13] rm outshy.asc
guiness ~/sftest/exercise [14] mv outshy.bfm xithalf.bfm
5906 -rw-r---- 1 anderson users 6023160 Oct 24 13:54 halfcell.bfm
5906 -rw-r---- 1 anderson users 6023160 Oct 24 13:57 upfull.bfm
5906 -rw-r---- 1 anderson users 6023160 Oct 24 13:59 xithalf.bfm

Parmela

Eric Colby has written a thorough description of the modified version of Parmela running on guiness. I recommend reading this document in addition to the documentation for the original version before continuing with these notes. Here I will just highlight some of the important differences.

Input Files

1. parmin6000 This is the main Parmela input deck. It is set up very similarly to the PC version of the code. Some important card are:
   - bzmap Similar to the poisson card in PC Parmela, but the input file placed on the next line is the outpoi file produced by the procedure given above. In cases where you use boundary conditions to model a bucking soleniod, (where $B_z(z=0) = 0$ in poisson units) the arguments of the bzmap card are just: scalef lowz highz. In cases where the field null is not at the origin in poisson coordinates, the full options are: scalef lowz_poi highz_poi lowz_poi highz_poi zmin zmax. Where lowz_poi and highz_poi are the start and stop positions of the input file in poisson coordinates, lowz_par and highz_par are the start and stop positions of the input file in Parmela coordinates, and zmin and zmax are the Parmela coordinates over which to use the input fields.
   - bfsetup Using this card on the first or second line of the file makes Parmela run up to the point of loading the background soleniodal fields,
and writing the output file `bzmap.d`. This is useful for check that you have the correct field map.

- **rfsetup** When this card is in the first or second line of the input deck, Parmela will run with 2 particles and no space charge.
- **ccell** Colby’s write-up explains this pretty well. Use `ctype = -99` for bfm input files from superfish.
- **input** Again, look at Colby’s documentation. Note that you have to use `itype 11` and the indicated Hammersley sequence bases.

## Output Files

1. **parmout6000** This is the main output file. It contains lines with this format:
   
   \[ N_{\text{step}}, Z_{\text{ave}}, N_{\text{part}}, \gamma_{\text{ave}}, X_{\text{rms}}, X_{\text{fwhm,i}} (i=1,5), \epsilon_{x,\text{rms}}, \epsilon_{x,\text{fwhm,i}} (i=1,5), \gamma_{\text{rms}}, Y_{\text{rms}}, Y_{\text{fwhm,i}} (i=1,5), \epsilon_{y,\text{rms}}, \epsilon_{y,\text{fwhm,i}} (i=1,5), Z_{\text{rms}}, Z_{\text{fwhm,i}} (i=1,5), \epsilon_{z,\text{rms}}, \epsilon_{z,\text{fwhm,i}} (i=1,5) \]

2. **moments.d** Contains the moments of the beam distribution in this format:
   
   \[ <x>, <\beta_x>, <y>, <\beta_y>, <z>, <\beta_z> [\text{mrad}], \sigma_x, \beta_{x,\text{rms}}, \sigma_y, \beta_{y,\text{rms}}, \sigma_z, \beta_{z,\text{rms}} [\text{mrad}] \]

3. **twiss.d** Contains the RMS Twiss parameters of the beam in this format:
   
   \[ Z_{\text{ave}}, \alpha_x, \beta_x, \alpha_y, \beta_y, \alpha_z, \beta_z \]

4. Other diagnostic files. These are files with the extension `.d` that are produced based on the `dfiles` input card used. Examples are `bzmap.d`, `betag.d`, and `psd___d.d`.

Here is an example of a Parmela simulation of the RF gun modeled with the superfish files generated above and a background solenoid field of the Poisson run shown above.
input 11 9999 3.0 3.0 0.23 0.23 9.0e-08 3.0e-08 0.10
    -1 -2 -3 1 1 1 1 1 -1 2856. -99 0
ccell 1.1875 1.1875 3.3782 1.1875 1.1875 0 0. 0.85. 0 1 1 -1 2856. -99 0
halfcell.bfm
cell 1.1875 1.1875 2.727 1.1875 1.1875 0 180. 0. 85. 0 2 1 1 2856. -99 0
upfull.bfm
cell 1.1875 1.1875 3.39427 1.1875 1.1875 0 180. 0. 85. 0 3 1 -1 2856. -99 0
xithalf.bfm
bxmap 0.897 0.00 50.0 0.00 50.00 0.00 50.00 0.00 50.00 0.00 50.00 0.00 50.00
sol.poi
drift 91.0 4.1
drift 20.49 4.1
scheff -6.25E+9 0.7 0.7 39 199 0 0 5 1.5 0
continue 0.1 5000 10 10 10
scheff -6.25E+9 0.8 0.8 39 199 0 0 5 1.5 0
continue 0.1 10000 5 32 32
scheff -6.25E+9 0.75 0.75 39 199 0 0 5 1.5 0
continue 0.1 5000 10 32 32
scheff -6.25E+9 0.65 0.65 39 199 0 0 5 1.5 0
continue 0.2 4000 10 32 32
scheff -6.25E+9 0.55 0.55 39 199 0 0 5 1.5 0
continue 0.2 4000 10 32 32
scheff -6.25E+9 0.475 0.475 39 199 0 0 5 1.5 0
continue 0.2 12000 10 32 32
makers
scheff -6.25E+9 0.4 0.4 39 199 0 0 5 1.5 0
continue 0.2 8000 10 32 32
scheff -6.25E+9 0.35 0.35 39 199 0 0 5 1.5 0
continue 0.2 4000 10 32 32
scheff -6.25E+9 0.3 0.3 39 199 0 0 5 1.5 0
continue 0.2 54000 10 32 32
makers
save
end

>>>PARMAIN: Using verbose mode.

>>>CREAD: card being processed: run
>>>CREAD: card being processed: output
>>>CREAD: card being processed: dfiles
>>>CREAD: card being processed: input

Generating 9999. particles from a photocathode
Quiet start with Hammersley sequence

sigma of laser pulse = 3.00 psec
sigma of laser pulse = 3.08 degrees
tmax of laser pulse = 3.00 psec
sigma of laser spot = 0.230 cm
rmax of photocathode = 0.230 cm
mean electron energy at cathode = 9.000E-02 ev
sigma of energy spread at cathode = 3.000E-02 ev
integration step size = 0.100 degrees
clock offset for earliest particle = 3.10 degrees
X centroid offset on cathode = 0.00 cm
Y centroid offset on cathode = 0.00 cm
bit reversal base for x = -1
bit reversal base for y = -2
bit reversal base for phi = -3
bit reversal base for W0 = 5

>>>CREAD: card being processed: ccell

Loading RF Cavity field from BINARY file: halfcell.bfm
Done.

>>>CREAD: card being processed: ccell

Loading RF Cavity field from BINARY file: upfull.bfm
Done.
>>> CREAD: card being processed: ccell
Loading RF Cavity field from BINARY file: xithalf.bfm
Done.

>>> CREAD: card being processed: bmap
Loading Background B-field data from file: sol.poi
Be sure to look at the output field dump file bmap.d to ensure correct reading/interpolation of the field map.

>>> CREAD: card being processed: drift
>>> CREAD: card being processed: drift
>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions
>>> rmax = 0.699999988 Nr= 39  dr= 17.9487171
>>> dzmax = 0.770000041 Nz= 199  dz= 3.8693471

Using LANL RZ mesh method, with images

>>> CREAD: card being processed: start
Creating dump of Bz(z) in file bmap.d

Integral B.dl for background field is : 31731.0273

Using launch phase: 45.0

<table>
<thead>
<tr>
<th>Step mt</th>
<th>Z</th>
<th>Ngood</th>
</tr>
</thead>
<tbody>
<tr>
<td>-30</td>
<td>0.000</td>
<td>10000</td>
</tr>
<tr>
<td>-20</td>
<td>0.000</td>
<td>10000</td>
</tr>
<tr>
<td>-10</td>
<td>0.001</td>
<td>10000</td>
</tr>
<tr>
<td>0</td>
<td>0.003</td>
<td>10000</td>
</tr>
<tr>
<td>10</td>
<td>0.006</td>
<td>10000</td>
</tr>
<tr>
<td>20</td>
<td>0.011</td>
<td>10000</td>
</tr>
<tr>
<td>30</td>
<td>0.016</td>
<td>10000</td>
</tr>
<tr>
<td>40</td>
<td>0.021</td>
<td>10000</td>
</tr>
<tr>
<td>50</td>
<td>0.029</td>
<td>10000</td>
</tr>
<tr>
<td>60</td>
<td>0.035</td>
<td>10000</td>
</tr>
<tr>
<td>70</td>
<td>0.045</td>
<td>10000</td>
</tr>
<tr>
<td>80</td>
<td>0.054</td>
<td>10000</td>
</tr>
<tr>
<td>90</td>
<td>0.066</td>
<td>10000</td>
</tr>
<tr>
<td>100</td>
<td>0.077</td>
<td>10000</td>
</tr>
<tr>
<td>200</td>
<td>0.236</td>
<td>10000</td>
</tr>
</tbody>
</table>

>>> SCHEFF: Calculating new Green Functions
>>> rmax = 0.699999988 Nr= 39  dr= 17.9487171
>>> dzmax = 1.15500021 Nz= 199  dz= 5.80402136

Step mt = 300  Z = 0.435  Ngood = 10000
Step mt = 400  Z = 0.681  Ngood = 10000

>>> SCHEFF: Calculating new Green Functions
>>> rmax = 0.699999988 Nr= 39  dr= 17.9487171
>>> dzmax = 1.73250031 Nz= 199  dz= 8.7060318

Step mt = 500  Z = 0.934  Ngood = 10000
Step mt = 600  Z = 1.196  Ngood = 10000
Step mt = 700  Z = 1.470  Ngood = 10000

>>> SCHEFF: Calculating new Green Functions
>>> rmax = 0.699999988 Nr= 39  dr= 17.9487171
>>> dzmax = 2.59875059 Nz= 199  dz= 13.0590477

Step mt = 800  Z = 1.748  Ngood = 10000
Step mt = 900  Z = 2.046  Ngood = 10000
Step mt = 1000 Z = 2.315  Ngood = 10000

Creating phase space dump file at mt = 1000
Step mt = 2000 Z = 5.155  Ngood = 10000

Creating phase space dump file at mt = 2000

>>> SCHEFF: Calculating new Green Functions
>>> rmax = 0.699999988 Nr= 39  dr= 17.9487171
>>> dzmax = 3.89812565 Nz= 199  dz= 19.5885715

>>> SCHEFF: Calculating new Green Functions
>>> rmax = 0.699999988 Nr= 39  dr= 17.9487171
>>> dzmax = 5.84718895 Nz= 199  dz= 29.3828583
Step \( mt \) = 3000 \( Z \) = 8.025 \( N_{\text{good}} \) = 10000
Step \( mt \) = 4000 \( Z \) = 10.898 \( N_{\text{good}} \) = 10000
Step \( mt \) = 5000 \( Z \) = 13.751 \( N_{\text{good}} \) = 10000

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.800000012 \( Nr \) = 39 \( dr \) = 20.5128212

>>> \( dz_{\text{max}} \) = 7.56099224 \( Nz \) = 199 \( dz \) = 37.9949379

>>> CREAD: card being processed: continue

Step \( mt \) = 6000 \( Z \) = 16.624 \( N_{\text{good}} \) = 10000
Step \( mt \) = 7000 \( Z \) = 19.540 \( N_{\text{good}} \) = 10000

D> Creating phase space dump file at \( mt \) = 7000

Step \( mt \) = 8000 \( Z \) = 22.457 \( N_{\text{good}} \) = 10000
Step \( mt \) = 9000 \( Z \) = 25.347 \( N_{\text{good}} \) = 10000
Step \( mt \) = 10000 \( Z \) = 28.290 \( N_{\text{good}} \) = 10000
Step \( mt \) = 11000 \( Z \) = 31.166 \( N_{\text{good}} \) = 10000
Step \( mt \) = 12000 \( Z \) = 34.066 \( N_{\text{good}} \) = 10000
Step \( mt \) = 13000 \( Z \) = 36.959 \( N_{\text{good}} \) = 10000
Step \( mt \) = 14000 \( Z \) = 39.826 \( N_{\text{good}} \) = 10000
Step \( mt \) = 15000 \( Z \) = 42.706 \( N_{\text{good}} \) = 10000

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.75 \( Nr \) = 39 \( dr \) = 19.2307701

>>> \( dz_{\text{max}} \) = 6.14501667 \( Nz \) = 199 \( dz \) = 30.8794804

>>> CREAD: card being processed: continue

Step \( mt \) = 16000 \( Z \) = 45.626 \( N_{\text{good}} \) = 10000
Step \( mt \) = 17000 \( Z \) = 48.471 \( N_{\text{good}} \) = 10000
Step \( mt \) = 18000 \( Z \) = 51.379 \( N_{\text{good}} \) = 10000
Step \( mt \) = 19000 \( Z \) = 54.269 \( N_{\text{good}} \) = 10000
Step \( mt \) = 20000 \( Z \) = 57.145 \( N_{\text{good}} \) = 10000

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.650000036 \( Nr \) = 39 \( dr \) = 16.6666679

>>> \( dz_{\text{max}} \) = 5.19865942 \( Nz \) = 199 \( dz \) = 22.5632973

>>> CREAD: card being processed: continue

Step \( mt \) = 21000 \( Z \) = 61.743 \( N_{\text{good}} \) = 10000
Step \( mt \) = 22000 \( Z \) = 67.536 \( N_{\text{good}} \) = 10000
Step \( mt \) = 23000 \( Z \) = 73.290 \( N_{\text{good}} \) = 10000
Step \( mt \) = 24000 \( Z \) = 79.058 \( N_{\text{good}} \) = 10000

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.550000012 \( Nr \) = 39 \( dr \) = 14.1025639

>>> \( dz_{\text{max}} \) = 4.19865942 \( Nz \) = 199 \( dz \) = 26.1239166

>>> CREAD: card being processed: continue

Step \( mt \) = 25000 \( Z \) = 84.730 \( N_{\text{good}} \) = 10000
Step \( mt \) = 26000 \( Z \) = 90.526 \( N_{\text{good}} \) = 10000
Step \( mt \) = 27000 \( Z \) = 96.293 \( N_{\text{good}} \) = 10000
Step \( mt \) = 28000 \( Z \) = 102.009 \( N_{\text{good}} \) = 10000

D> Creating phase space dump file at \( mt \) = 28200

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.475000024 \( Nr \) = 39 \( dr \) = 12.1794872

>>> \( dz_{\text{max}} \) = 3.49009609 \( Nz \) = 199 \( dz \) = 22.5632973

>>> CREAD: card being processed: continue

Step \( mt \) = 29000 \( Z \) = 107.575 \( N_{\text{good}} \) = 10000
Step \( mt \) = 30000 \( Z \) = 113.549 \( N_{\text{good}} \) = 10000
Step \( mt \) = 31000 \( Z \) = 119.230 \( N_{\text{good}} \) = 10000

>>> CREAD: card being processed: makers

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.400000006 \( Nr \) = 39 \( dr \) = 10.2564106

>>> \( dz_{\text{max}} \) = 3.78182697 \( Nz \) = 199 \( dz \) = 19.0041542

>>> CREAD: card being processed: continue

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.350000024 \( Nr \) = 39 \( dr \) = 8.97435951

>>> \( dz_{\text{max}} \) = 3.30909896 \( Nz \) = 199 \( dz \) = 16.6286373

>>> CREAD: card being processed: continue

>>> CREAD: card being processed: scheff

>>> SCHEFF: Calculating new Green Functions

>>> \( r_{\text{max}} \) = 0.300000012 \( Nr \) = 39 \( dr \) = 7.69230795

>>> \( dz_{\text{max}} \) = 2.83637261 \( Nz \) = 199 \( dz \) = 14.2531281

>>> CREAD: card being processed: continue

>>> CREAD: card being processed: makers
Here is $\sigma_x$ and $\varepsilon_x$ produced by this run.
Changes to the **INPUT** Card: Thermal Emittance
Salime Boucher, UCLA PBPL, 10/18/00

**INPUT** 11 NP SIGMAT MAXT SIGMAR MAXR W0 DW0 DWT HBX HBY HBPHI HBW0 ASPECT XOFF YOFF ANOISE FNOISE TANOISE TFNOISE ELASER EWORK FILENAME

ITYPE - ITYPE=11 invokes Jerome Gonichon's "Quiet Start", unless ELASER and EWORK are given, in which case the thermal emittance distribution is used.

NP - Number of particles

SIGMAT - Temporal sigma (in picoseconds). If SIGMAT < 0, then a user-defined subroutine called LASER will be called to produce the temporal behavior of the laser pulse. See preamble to subroutine for specific instructions.
NOTE: If ANOISE is nonzero, then this is the risetime of the laser pulse in picoseconds.

MAXT - Cut on temporal distribution (in picoseconds), half-length of flat top if ANOISE is nonzero.

SIGMAR - Spatial sigma (in cm) = horizontal dimension of beam
If SIGMAR is negative, a user defined subroutine XSECTN will be called to produce a custom cross section shape. If SIGMAR=-666, then (X,Y) distribution will be generated by the rejection method from the image file FILENAME.

MAXR - Cut on spatial distribution (also in cm) If SIGMAR=-666, then RMAX specifies the scaling factor to use in adjusting the magnification of the input. Defaults to 1.0.

WO - Mean emission energy (in MeV)

DWO - Energy spread (in MeV)

DWT - Clock step size, same as DWT on START card (Degrees)

HBX - Hammersley sequence base for X dist. generation (choose: 1)

HBY - Hammersley sequence base for Y dist. generation (choose: 2)

HBPHI - Hammersley sequence base for PHASE dist. gen. (choose: 3 for Gaussian, -3 for uniform distribution. Choosing a negative base number overrides the SIGMAT setting above)

HBW0 - Hammersley sequence base for ENERGY dist. gen. (choose: 5)
Notes: In general, the base choices suggested here {1,2,3,5} can be substituted by any four prime numbers. Higher base numbers result in larger scale structure (i.e. "stripes" in the distribution) and should be avoided. DO NOT use the same base number twice, or a perfectly correlated distribution in the two parameters will result.
ASPECT - Ratio of vertical to horizontal beam dimensions. (Assumed 1 if omitted) If ASPECT < 0, then beam cross section is rectangular, not elliptic.

XOFF - X distance to displace laser spot from symmetry axis, [cm]

YOFF - Y distance to displace laser spot from symmetry axis, [cm]

ANOISE - Amplitude of intensity noise on laser pulse. (0.0 to 1.0)

FNOISE - Harmonic number of intensity noise modulation applied to laser (eq. if FNOISE=10, the intensity modulation will have ten periods over the bunch length.

TANOISE - Amplitude of transverse intensity noise on laser pulse. (0.0 to 1.0) (This option induces sinusoidal variations in the x-directions)

TFNOISE - Harmonic number of transverse intensity noise modulation applied to laser (eq. if TFNOISE=10, the intensity modulation will have ten periods over the bunch length.)

ELASER - The energy of the laser photons. If this is given, the thermal emittance distribution is used.

EWORK - The work function of the cathode.