

ICOOL User's Guide

Version 1.7
26 March 2010

R.C. Fernow
Brookhaven National Laboratory

Contents

1. Introduction

- 1.1 file structure
- 1.2 basic job structure

2. Basic concepts

- 2.1 comments
- 2.2 beam definition
- 2.3 regions
- 2.4 repeats
- 2.5 cells
- 2.6 reference particle
- 2.7 reference orbit
- 2.8 material
- 2.9 postprocessor output file
- 2.10 restarting a job
- 2.11 name substitution
- 2.12 key input during execution

3. Control variables

- 3.1 global simulation
- 3.2 analysis & diagnostics
- 3.3 print
- 3.4 beam restart
- 3.5 postprocessor output
- 3.6 accuracy
- 3.7 moving grid
- 3.8 spin
- 3.9 neutrino production
- 3.10 miscellaneous

4. Interaction variables

- 4.1 global controls
- 4.2 energy loss
- 4.3 multiple scattering
- 4.4 correlated scattering and straggling
- 4.5 decays
- 4.6 nuclear interactions
- 4.7 space charge

5. Internal diagnostics

- 5.1 region summary table
- 5.2 data print out
- 5.3 histogram
- 5.4 scatterplot

- [5.5 z-history](#)
- [5.6 r-history](#)
- [5.7 emittance](#)
- [5.8 covariance](#)
- [5.9 magnetic field map](#)
- [5.10 RF diagnostics](#)

[6. Pseudoregion commands](#)

- [6.1 simulation](#)
- [6.2 postprocessor file](#)
- [6.3 field grid](#)
- [6.4 error analysis](#)
- [6.5 background fields](#)
- [6.6 miscellaneous](#)

[7. Field description](#)

- [7.1 hard edge](#)
- [7.2 linear longitudinal variation](#)
- [7.3 soft edge](#)
- [7.4 current elements](#)
- [7.5 on-axis field](#)
- [7.6 grids](#)
- [7.7 other static fields](#)
- [7.8 time-varying fields](#)

[8. ICOOL utilities](#)

- [8.1 ERRSUM2](#)
- [8.2 ECALC9F](#)
- [8.3 ENDOF9](#)
- [8.4 NOTEND9](#)
- [8.5 EXTPAR9](#)
- [8.6 EXTREG9](#)
- [8.7 BUNCH9](#)

[9. Other useful utilities](#)

- [9.1 HistoRoot](#)

[10. References](#)

1. Introduction

[ICOOL](#) is a 3-dimensional tracking program that was originally written to study ionization cooling of muon beams [1,2]. The program simulates particle-by-particle propagation through materials and electromagnetic fields. Particles are tracked and regions are described using "accelerator" coordinates. The program was written with low energy (1 MeV/c -- 1 GeV/c) muons in mind, but tracking of electrons, pions, kaons, and protons is also possible. The physics processes available include decays, delta rays, multiple scattering, energy loss and straggling.

Information is input to the code and written out via ASCII data files.

The incident beam particles can be generated from uniform or Gaussian distributions or read from an input file. The code can read its own output, so simulations can be staged. The particles are tracked through a sequence of fixed-length longitudinal regions. In general a region is cylindrical in shape and may be subdivided radially. Every region has a specified material and field type associated with it. Groups of regions can be grouped in cells and a separate cell field can be superimposed over the region fields when tracking is done. In general the program takes user-defined steps along a reference axis. For each step it updates the particle position, time and momentum, taking into account the local field, and corrects the particle's momentum for energy loss and multiple scattering in the step. There is an option for letting the program make adaptive step sizes.

ICOOL uses analytic and other methods to compute field strengths at a given location. There are in general several model levels for each field type, which determine the approximation used to calculate the field.

The program always generates an output log file. In addition, depending on control variable settings, it may generate several other output files. A number of internal diagnostics, such as histograms and scatterplots, are conveniently available. There are options for saving information about each particle after every region (or step) in a postprocessor file.

This guide gives an overview of the main features of ICOOL from a functional point of view. Many examples have been included. However, not all capabilities of the code are discussed here. A complete description of all commands can be found in the ICOOL Reference Manual, [icoolman.pdf](#).

1.1 File structure

The source code for ICOOL is broken up logically into the following files.

file	description
icommon.inc	common blocks
icool.for	Monte Carlo superstructure
ifld.for	field routines
iint.for	interaction routines
idiag.for	internal diagnostics
imath.for	mathematical routines
imsoft.for	Windows-specific code

When running a job ICOOL uses the file naming convention for0xx.dat. File numbers 01 to 19 are reserved by the program. Some common execution files are listed below.

file	type	description
for001.dat	input	command file
for002.dat	output	log file
for003.dat	input	beam file
for004.dat	output	beam file
for007.dat	output	region summary table
for008.dat	internal	overflow particle file
for009.dat	output	postprocessor file
for011.dat	output	beam envelope file

Every job must have at least a command file for001.dat. Sections 2-7 of this guide are mainly concerned with a description of the contents of this file. Every job produces at least a log file for002.dat. Section 8 of this guide describes a number of useful utility programs that can be used with the output files from an ICOOL job.

1.2 Basic job structure

An ICOOL job is run by first setting up the command file for001.dat and any other required files for0xx.dat, and then invoking the executable by typing *icool*. The basic parts of the command file are

1. title
2. control variables (&cont)
3. beam definition (&bmt)
4. interactions definition (&ints)
5. internal diagnostics (&nhs, &nsc, &nzh, &nrh, &nem, &ncv)
6. region definition (SECT ...)

A simple example of a complete command file follows.

```
Drift space example with a scatterplot of x vs. y
&cont npart=500 /           ! control variables
&bmt /                       ! beam definition
1 2 1. 1                     ! Gaussian definition of muon beam
0. 0. 0. 0. 0. 0.200       ! means
3e-3 3e-3 0.01 0.005 0.005 0.010 ! sigmas
0                             ! no imposed beam correlations
&ints /                       ! use default interactions
&nhs /                         ! no histograms
&nsc nscat =1 /              ! define 1 scatter plot
-0.10 5e-3 40 1 2 -0.10 10e-3 20 2 2
&nzh /                         ! no z-histories
&nrh /                         ! no r-histories
&nem /                         ! no 2-D emittance calculations
&ncv /                         ! no covariance calculations
SECTION                       ! start problem definition
SREGION                       ! define a region
1.00 1 0.003                 ! length, 1 radial subregion, step
1 0. 0.10                    ! radial extent
NONE                          ! no associated field
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC                            ! vacuum material
CBLOCK                        ! cylindrical block geometry
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
ENDSECTION                    ! end of problem definition
```

Note that the default units used in ICOOL are meters, seconds, GeV/c, tesla and MV/m.

2. Basic concepts

2.1 Comments

Comments may be inserted anywhere in the input file for001. A comment is anything on a line that follows the ! character. Blank lines may also be inserted to improve clarity in the input file.

```
! this whole line is a comment  
SREGION      ! so is the remainder of this line
```

“Comments and blank lines can help format a control file so that it is more understandable to human readers. The computer subroutines that read and parse control files don't want to see the comments and blank lines. A subroutine in ICOOL removes comments and blank lines before the command processor parses them.

Rules for commenting a control file

A *comment* is any string of printable characters whose leftmost character is an exclamation point.

! This is a comment.

!So is this.

! Additional exclamation points !! in a comment don't matter.

A *whitespace* is either a space or a horizontal tab.

A *blank line* either has no characters (except for the end-of-line terminator) or no characters except whitespaces. Blank lines may be placed anywhere in the control file. They are ignored by the command parser.

A *comment line* contains a comment, preceded by zero or more whitespaces. Comment lines may be placed anywhere in the control file. They are ignored by the command parser.

An *end-of-line comment* is a comment placed to the right of a normal input line. End-of-line comments may be placed at the end of any input line, separated from the data ICOOL is to parse by zero or more whitespaces.

1.2342.345 !This is an end-of-line comment.

3.4564.567!So is this (valid but hard to read).

RING!This is valid even though "RING" will be read into a 6-character field.

RING !This is a more readable end-of-line comment.

The Use of Informal Comments

Though it is not forbidden, users are strongly discouraged from using "informal comments" --

those without leading exclamation points that are imagined to be out of view of the parser. There are numerous ways to go wrong. For example, free format reads can extract data from more than one input line; an informal comment on the end of any but the last line will generally produce an error. An informal comment following a text string must come with enough preceding spaces to fill the input field with spaces. All of these potential problems are avoided by using formal comments with leading exclamation points.” {S.B.}

2.2 Beam definition

The beam particles used for a simulation may be defined internally or read in from a file, depending on the value of the control variable BGEN. Internal generation uses the namelist command \$BMT. The input beam can consist of one or more types of particle, each with its own production distribution. The distributions can be either gaussian or part of a uniform circular segment. A number of correlations can be imposed on the distribution, including angular momentum and longitudinal momentum-transverse amplitude. Beams with both signs of charge can be generated using the variable BMALT. ICOOL uses an internal particle numbering system

code	particle
±1	electron
±2	muon
±3	pion
±4	kaon
±5	proton
6	deuteron
7	He ³
8	Li ⁷

where the sign indicates the charge of the particle.

Example: generate a muon beam with a gaussian distribution

```
&bmt /
1 2 1. 1
0. 0. 0. 0. 0. 0.200
0.010 0.010 0.010 0.010 0.010 0.010
0
```

Externally produced beam files are read from the file for003.dat.

2.3 Regions

The basic unit in a problem description is called a region. This command describes the field and material in a given longitudinal area. The basic structure is

```
SREGION
SLEN      NRREG      ZSTEP
IRREG     RLOW       RHIGH
FTAG
FPARM
MTAG      (MTAG2)
MGEOM
GPARM
```

Each region begins with an SREGION command. There is no specific end command for a region.

SLEN is the axial length of the region. In a curved region this length follows the bent path. A region can be broken up into up to 4 radial subdivisions. NRREG is the number of radial subdivisions. ZSTEP gives the stepsize for this region. The following commands are repeated for each radial subdivision. IRREG enumerates the radial regions. RLOW is the minimum radius and RMAX is the maximum radius for this subdivision. FTAG is a character string identifying the field. This is followed by 15 parameters FPARM that give the specific details for the field. MTAG is a character string that identifies the material in the region. For a wedge an optional second material may be given that specifies the material exterior to the wedge. MGEOM is a character string specifying the geometry of the material. This is followed by 10 parameters GPARM that give specific details about the geometry.

Example: Simple solenoid with no material

```
SREGION
0.20 1 0.01
1 0. 0.30
SOL
1. 0. 0. 0. 1.0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

Example: stepped beryllium window

```

SREGION      !
750e-6  1  100e-6
1  0.00  0.25
NONE
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
BE
CBLOCK
0  0  0  0  0  0  0  0  0  0
SREGION      ! rf graded Be window
750e-6  2  100e-6
1  0.00  0.20
NONE
0  0  0  0  0  0  0  0  0  0  0  0  0  0
VAC
CBLOCK
0  0  0  0  0  0  0  0  0  0
2  0.20  0.25
NONE
0  0  0  0  0  0  0  0  0  0  0  0  0  0
BE
CBLOCK
0  0  0  0  0  0  0  0  0  0

```

2.4 Repeats

The simplest form of looping can be done using the REPEAT structure. The basic form of this structure is

```

REPEAT
NREP
  whatever you want to repeat
ENDREPEAT

```

NREP is the number of times you want the commands repeated. REPEAT structures cannot be nested. They are often used to describe a string of identical RF cavity cells.

Example: String of 4 pillbox rf cavities

```

REPEAT
4
SREGION
0.20  1  0.01
1  0.  0.30
ACCEL
2. 201.25 11.0 0. 0.  0. 0. 0. 0. 0.  0. 0. 0. 0. 0.
VAC
CBLOCK
0  0  0  0  0  0  0  0  0  0  0
ENDR

```

2.5 Cells

The CELL structure can also be used for looping, but the main use is to define the lattice field for a channel. The basic structure is

```
CELL
NCELLS
CELLFLIP
CFTAG
CFPARM
    whatever is inside the cell
ENDCELL
```

NCELLS is the number of cells. A cell has an associated field given by the parameters CFTAG and CFPARM. Each repetition of the cell uses the same field, unless CELLFLIP is true, in which case alternate cells have reversed polarity.

Example: 15 cells of an alternating solenoid lattice. The cell field is a model 3 SHEET. The second SHEET parameter specifies the input data is in the file for020.dat.

```
CELL
15
.true.
SHEET
3 20 0.01 0.01 1.51 0.11 15.0 0 0 0 1. 0 0 0 0
SREGION
1.50 1 0.01
1 0. 0.30
NONE
0. 0 0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0 0 0 0 0 0 0 0 0 0 0
ENDC
```

ICOOL can define the magnetic field on a cylindrical grid made up from a sum of individual coils or current sheets. It is possible to define this grid over a short portion of the field lattice and then reuse the same grid over and over. The procedure is to define the grid from $z=0$ to one grid point beyond the end of one cell. The length of the cell is either the period of the magnetic field, or half the period if the field alternates in direction. One then defines the sheets, for example, which create the field starting one cell before the cell of interest and extending to one cell beyond it, i.e. over 3 cells. ICOOL gets the field it needs from the z distance from the start of the cell it is currently in, so this same grid can be used at any z location in the lattice.

2.6 Reference particle

When running with RF cavities it is customary to define a reference particle. The reference particle is used to automatically set the phase of each cavity to 0-crossing of the electric field when the reference particle passes the center of the cavity. The reference particle is initialized using the pseudocommand

```
REFP
REFPAR  PZOREF  TOREF  GRADREF  PHMODREF
```

where REFPAR describes the particle type, PZOREF is the initial momentum and TOREF is the initial time. PHMODREF = {3,4} describes how the reference particle moves. If PHMODREF=3 the reference particle moves with constant velocity. If it equals 4, then the particle slows down because of energy loss in matter and speeds in RF cavities at a rate given by GRADREF. The control variable PHASEMODEL must be set to 3 to enable this feature. The REFP command should immediately follow the SECT command in for001.dat. The relative phasing of the real particles with respect to the reference particle can be changed for example by adjusting TOREF or the phase offsets for the RF cavities. A second reference particle defined by the pseudocommand REF2 is used for ACCEL model 10.

Example: use a muon traveling with constant momentum 200 MeV/c as a reference particle

```
SECT
REFP
2 0.200 0. 0. 3
```

The reference particle is designated as particle 0 in the log output. A second reference particle, if any, is denoted as particle -1 in the log file output.

If a problem is restarted using an existing beam file that has a non-zero value for P_z for the reference particle momentum, the reference particle information for the problem is initialized using information from the file. A REFP command must still be used to activate the reference particle logic for the new job. Normally the REFP command immediately follows the SECTION command. The reference particle can be redefined later in a problem, if desired, by using another REFP command. The reference particle data is initialized using the input file data. The momentum is reset by the REFP command if the parameter PZOREF is not equal to 0. The time is reset by the REFP command if the parameter TOREF is not equal to 0.

When a reference particle has been defined and DIAGREF=true, the internal diagnostics in ICOOL compute time and longitudinal momentum as the difference from the reference particle value at a given location. In addition, when the control variable PHASEMODEL is 2-4, the phases of the rf cavities are initialized from the reference particle. The reference particle is started on-axis with no divergence. Stochastic processes, such as multiple scattering and straggling are temporarily turned off. The time when the reference particle crosses the center of the cavity is used to initialize the cavity electric field to 0 (zero crossing). When PHASEMODEL=3 the reference particle is assumed to move with

constant velocity (determined from the parameter PZOREF). When PHASEMODEL=4 the reference particle is tracked through all regions except RF cavities. The reference particle velocity drops due to the mean dE/dx in any material that may be present. It is assumed that the reference particle gains energy linearly in the cavity region. The amount of energy it gains is determined by the parameter GRADREF. You can use (1) the mean value of the initial time position or (2) the parameter PHASE0 for the ACCEL field type to adjust the phase of the non-reference particles.

2.7 Reference orbit

Tracking is done in ICOOL in the Frenet-Serret coordinate system. This is a right-handed system where s is tangent to the reference orbit, y is vertical and x is the third orthogonal coordinate. In a circular orbit x is in the radial direction. We define the reference orbit to be that path where the transverse coordinates x and y and the transverse momenta p_x and p_y always remain zero. The shape of the reference orbit in a global Cartesian coordinate system is determined by the curvature parameter. It is important to understand that the reference orbit is not necessarily the same path taken by the RF reference particle described in the previous section.

2.8 Material

Every region has an associated material type and geometry. ICOOL has a number of predefined material types, such as LH for liquid hydrogen. The standard material geometry is the cylindrical block CBLOCK. However, a number of wedge-shaped geometries are also available for studies of emittance exchange.

```
Example: liquid hydrogen absorber in a drift region
SREGION
0.20 1 0.01
1 0. 0.30
NONE
0 0 0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
LH
CBLOCK
0 0 0 0 0 0 0 0 0 0
```

Example: lithium hydride wedge absorber in a drift region. The wedge has an opening angle of 15° and has the apex oriented vertically.

```
SREGION
0.20 1 0.01
1 0. 0.30
NONE
0 0 0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
LIH
WEDGE
15. 0.05 0.10 90. 0.10 0.10 0 0 0 0
```

2.9 Postprocessor output file

ICOOL can save information about the particles at production and about the particles and their electromagnetic field environment at various z locations in the problem in the file FOR009.DAT for later analysis by the user. Particle information at production is included in the file by setting the control variable OUTPUT1 true. Note that data for the electromagnetic field environment at the production point is NOT included in these records because ICOOL doesn't compute fields until after the particles move away from the production plane. This deficiency can be effectively remedied though by simply defining the first region to be a very thin drift space or by entering the BZ field manually using the control variable BZFLDPRD.

The user can generate particle and field information at the end of a given ICOOL region by preceding the region definition with the OUTPUT pseudocommand. Alternatively, particle and field information at the end of every region can be generated by setting the control variable NTUPLE true.

Particle and field information inside of regions can be generated by setting the control variable RTUPLE true. This variable should be used with caution, since it is capable of generating an enormously large file. The variable RTUPLE works together with the control variable RTUPLEN, which specifies the number of steps inside the region to skip before generating output.

The precision (and length) of the data in the output file can be controlled using the control variable F9DP.

2.10 Restarting a job

It is often convenient to save the beam and reference particle at some location as the starting conditions for a subsequent job. This can be done using the control variables FSAV, IZFILE and FSAVSET. If FSAV is true, all particles that reach the region number IZFILE are written to the file for004.dat. If FSAVSET is true, the data is modified to have z=0 and times relative to the reference particle. This file has the same format as the ICOOL beam input file for003.dat, so the job can be restarted by copying the file for004 to for003 and then running ICOOL with the control variable BGEN false.

2.11 Name substitution

Parameters in for001 may be referred to symbolically using name substitution. The basic structure is

```
&SUB name value
```

Following this command any occurrence of the text string `&name` is replaced with its value. This feature can be used for example to change a parameter in many regions at the same time, or to conveniently change a parameter at the top of the file when doing an optimization study.

```
Example: define region length symbolically
  &SUB slen 0.20
  SREGION
  &slen 1 0.01
```

A variant allows scaling of a class of previously-defined symbolic names.

```
&SCL NameString type value
```

NameString is a set of characters. All previously defined names that begin with this character string will be scaled. The type of scaling is determined by `type = {*, +}`. The amount of scaling is determined by value.

Example: scale all parameters starting with the letter s to 99% of their value

```
&SUB    step1    0.01
&SUB    step2    0.015
&SCL   s      *    0.99
SREGION
0.20  1  &step1
```

2.12 Keyboard input during execution

ICOOOL recognizes two keyboard inputs while it is executing.

- p pauses execution until the Enter key is pressed
- x stops execution with the current particle and region and executes any end of run diagnostics.

3. Control variables

There are a number of variables that can be used to control an ICOOL simulation. Many of these variables have default values that are used if the variable is not defined in a given job.

3.1 Global simulation

BGEN is used to control whether ICOOL generates the beam particles internally or from the external beam file for003.dat.

BUNCHCUT sets the maximum time difference allowed between a particle and the reference particle. It is used to stop tracking particles that fall out of an RF bunch.

DECTRK determines whether the charged daughter particle from a decay is tracked.

NPART sets the number of particles used in the simulation.

NPSKIP sets the number of beam particles in an external file to skip before the simulation starts.

NSECTIONS sets the number of times the particles are tracked through an ICOOL section. A section consists of all region commands between the BEGS and ENDS commands. This is frequently used to simulate multiple turns around a ring.

```
Example: simulate 10 turns around a ring following an injector lattice
&cont npart=1 nsections=10 /
.....
SECT
    (the injector lattice description goes here)
BEGS
    (the ring description goes here)
    (only this part is repeated)
ENDS
```

PHASEMODEL is used to specify how the phases of the RF cavities are determined. In jobs with a reference particle this parameter determines how the reference particle moves. If the value is 3, the reference particle moves with a constant velocity, while if the value is 4, it loses energy in material and gains energy in RF cavities. If the value is 5, the RF cavity information is read in from a file.

RFPHASE specifies an external file of RF cavity information that can be used with phasemodel 5.

TIMELIM sets the maximum time allowed for the simulation.

3.2 Analysis and diagnostics

BETAPERP sets the value of the transverse beta function for calculation of the transverse amplitude parameter A^2 .

BZFLDPRD gives the value of the solenoid field at production. This needed for the postprocessor file and emittance calculations at production.

DIAGREF controls whether time and p_z are specified relative to the RF reference particle for ICOOL internal diagnostics.

FORCERP is used to force the reference particle back to the axis when phasemodel=4.

GOODTRACK determines whether failed tracks in a beam input file are processed.

PHANTOM forces a particle to maintain its initial transverse coordinates after every step. This can be used to examine field components at fixed transverse positions.

RFDIAG is used to write diagnostic information for every RF cavity.

RNSEED is used to set the initial random number seed.

RUN_ENV is used to run ICOOL in a paraxial beam envelope mode, instead of doing particle tracking. This only works for straight solenoid-focused channels. Quantities are only calculated for planes that are written to the postprocessor file, for009.dat. The initial beam parameters are taken from the ICOOL beam input. Envelope parameters are written to the file for011.dat.

Example: examine the paraxial behavior of a straight channel using the beam envelope mode.

```
Run channel in beam envelope mode
&cont npart=1 ntuple=.true. run_env=.true. /
&bmt /
1 2 1. 1
0. 0. 0. 0. 0. 0.200
0.010 0. 0. 0.010 0. 0.010
.....
```

This will automatically run a single particle along the axis, and evolve the envelope equations. OUTPUT must be enabled to the postprocessor file for009.dat. Envelope parameters are output to the file for011.dat.

“Currently, this only works if the beam parameters are given from the BMT input lines. Only the first beam type is used. This applies even if bgen is set to .false.; the particle data in for003.dat is ignored... There must be nonzero sigma x and sigma Px; in addition, the average Pz must be > 0.

The envelope equations assume that there are no dispersive regions or shaped absorbers,

the bunch is well matched into the rf bucket, and the reference orbit stays on axis. In addition, the envelope equations will not work well for induction linacs or bunching sections.

The only source of longitudinal emittance growth considered is the slope of the dE/ds curve. The size of the rf bucket is ignored, but this can be put in after the fact (see below). Besides paraxial equations of motion, beam scraping is included where the beam is scraped against the widest aperture for each region.

How to interpret the results of the envelope equations

The parameter Psi (listed as "gauss" in the output file) represents how severely the beam is scraped; when Psi is large, only the extreme tails are missing; when Psi is small, the beam distribution is almost flat until the cutoff. The envelope equations give as results z , B_z , $\langle t \rangle$, $\langle pz \rangle$, transverse emittance e_N , beta, alpha, and dimensionless canonical angular momentum, transverse scraping parameter Phi (listed as "gauss" in output file). There are also parameters which do NOT take into account losses due to falling out of rf bucket, in particular the total number N_0 , and the longitudinal emittance e_{L0} . Dimensionless canonical angular momentum is $\langle L_{\text{canon}} \rangle / (2mc e_N)$. Particle number and longitudinal emittance ignores losses due to particles falling out of rf bucket. Thus, if rf bucket separatrix corresponds to amplitude A_{RF} , then after those losses are taken into account the particle number is $N = N_0 G_0(A_{RF}/e_{L0})$.

The longitudinal emittance taking into account losses from falling out of the rf bucket is $e_L = e_{L0} G_1(A_{RF}/e_{L0}) / G_0(A_{RF}/e_{L0})$.

The transverse "scale length" (not given in output file) is given by $A_G = e_N G_1(\text{Psi}) / G_2(\text{Psi})$.

The fraction of particles within the transverse acceptance C_{perp} is $f_{\text{perp}} = G_1(C_{\text{perp}}/A_G) / G_1(\text{Psi})$, or just 1 when $C_{\text{perp}} > \text{Psi} A_G$.

The fraction of particles with the longitudinal acceptance C_L is $f_L = G_0(C_L/e_{L0})$, if $C_L < A_{RF}$, or $G_0(A_{RF}/e_{L0})$ if $C_L > A_{RF}$.

The total number of particles within the 6D acceptance defined by C_{perp} , C_L , is given by $N_{6D} = N_0 f_{\text{perp}} f_L$

Key functions:

$$G_0(x) = 1 - \exp(-x/2)$$

$$G_1(x) = 1 - (1+x/2)\exp(-x/2)$$

$$G_2(x) = 1 - (1+x/2+x^2/8)\exp(-x/2) \quad \{G.P\}$$

SUMMARY is used to write a file containing a list of all the regions (and pseudoregions) defined in the job.

3.3 Print

Print diagnostics are written to the log file for002.dat.

NPRNT sets the number of particles for which print diagnostics is desired.

PRLEVEL specifies the amount of information to print for each particle.

PRNMAX sets the maximum number of steps to print out inside a given region.

3.4 Beam restart

FSAV specifies whether beam information at a given location is saved to the file for004.dat.

IZFILE specifies the location where beam information should be saved.

FSAVSET tells whether the information saved in for004.dat should have the z position and time reset.

3.5 Postprocessor output

These commands control the amount of information written to the postprocessor output file for009.dat.

F9DP sets the number of decimal places to use for floating point variables.

NTUPLE determines whether to write output after every region.

OUTPUT1 determines whether production information is written to the file.

RTUPLE is used to write output after a fixed number of steps inside a region.

RTUPLEN sets the number of steps to skip between outputs when using RTUPLE.

3.6 Accuracy

EPSF sets the tolerance on the fractional change on field variations, energy loss or multiple scattering when using variable stepping.

EPSREQ sets the required tolerance on tracking when using variable stepping.

EPSSTEP sets the desired tolerance for reaching the end of a region.

PZMINTRK sets the minimum value of P_z a track can have, after which tracking is stopped.

SCALESTEP sets a factor that modifies all the step sizes in a problem simultaneously. This only works when not using variable stepping.

STEPMAX specifies the maximum step size to use for variable stepping.

STEPMIN specifies the minimum step size to use for variable stepping.

STEPRK specifies whether to use the Runge-Kutta or the Boris algorithm for stepping.

“The Boris integrator is a space-stepping integration scheme similar to the time-based scheme used in plasma physics (see "Plasma Physics via Computer Simulation", Birdsall and Langdon, p. 356). This scheme is second-order accurate, compared to fourth-order for the default Runge-Kutta, so a smaller stepsize may be required. However, the Boris integrator is less computationally intensive and so may give significantly shorter run times. Adaptive stepsize control is not yet implemented for the Boris integrator.” {P.S.}

The Boris integrator is enabled by setting the control variable STEPRK=false. It only works for straight regions. The program automatically switches to Runge-Kutta integration in curved regions. The underlying algorithm is described in MC note 229.

VARSTEP specifies whether to use fixed or variable (adaptive) step sizes in tracking.

3.7 Moving grid

A solenoidal channel can be defined in ICOOL from a single list of coil properties. A second file is used to define the field grid partitions where the magnetic field must be calculated. The program fills the first grid partition, tracks all the particles to the end of the grid, then recomputes the field values for the second grid partition, tracks all the particles, and so forth.

MAGCONF specifies a file containing the coil information.

MAPDEF specifies a file containing the field grid partition data.

3.8 Spin

The muon spin is followed in pion decay, tracking in fields and interactions in matter.

SPIN determines whether spin calculations are processed. It also generates muon polarization from pion decay.

SPINMATTER determines whether depolarization effects in matter are simulated.

SPINTRK controls whether spin variables are tracked.

Setting the control variable SPIN true enables the handling of spin in ICOOL. At present there are two ways of getting initial spin information: (1) the user may read in a beam input file with spin information already present or (2) muon spin from pion decay may be calculated inside the program. If the control variable SPINTRK=0, the muon helicity is stored in the variable POL(3) when the pion decays and is unchanged as the particle is tracked.

If the control variable SPINTRK=1, the BMT equation is used to track the particle spin in the muon rest frame through electric and magnetic fields. The spin tracking needs a 3-component, normalized Cartesian vector to represent the spin. For muons from pion decay the array POL contains the spin vector in the muon rest frame.

Muon depolarization effects in matter are controlled by the variable SPINMATTER. If SPINMATTER=1 the Rossmanith model of depolarization is used, which relates the amount of depolarization to the energy loss in material regions. If SPINMATTER=2 quantum mechanical formulae are used for the probability of spin flip due to elastic and Coulomb scattering. The elastic scattering effect is more important and requires that delta rays interactions be enabled (LDRAY=true).

3.9 Neutrino production

Neutrino production information can be saved from muon, pion and kaon decays.

NEUTRINO controls whether neutrino production information is written to a file.

NNUDK sets the number of neutrinos to produce at each muon, pion or kaon decay location.

NUTHMAX sets the maximum polar angle for neutrinos written to the file.

NUTHMIN sets the minimum polar angle for neutrinos written to the file.

It is possible to make files of neutrino production data using four control variables. The variable NEUTRINO specifies the output file, if any, that should be used. Neutrinos can

be generated from muon, pion, and kaon decays. A large number (NNUDK) of independent neutrinos can be generated at each decay point. The user can select which of the generated events are written to the output file using the polar angle variables NUTHMIN and NUTHMAX. Neutrinos have a particle type that uniquely identifies their flavor and their parent particle.

Example: save information on neutrino production from pion decay in a file, e.g. for030.dat. Produce 100 neutrinos at each decay point. Only accept neutrinos produced within 100 mrad of the axis.

```
Save neutrino production data
$cont npart=1000 neutrino=30 nnudk=100 nuthmax=0.100 $
$bmt $
1 3 1. 1
0. 0. 0. 0. 0. 0.200
0.010 0.010 0.010 0.010 0.010 0.010
.....
```

3.10 Miscellaneous

FFCR inserts form feed and carriage return characters into the log file for002.dat so that there are two plots per printed page.

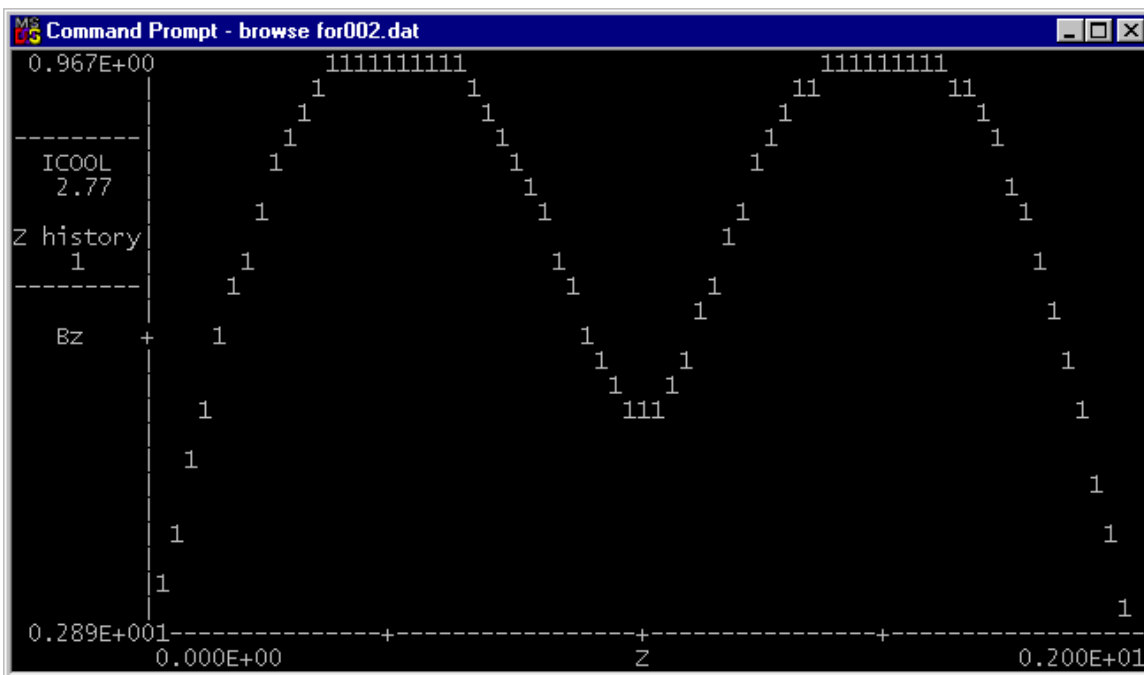
NEIGHBOR specifies whether fields from neighboring regions can overlap when using ICOOL soft-edge field models. The control variable NEIGHBOR can be used to model the contribution to a given region of the tails from the fields from neighboring regions. This is used with the soft-edged, analytic delta hyperbolic tangent model of the field available for the field types {BSOL , DIP , HDIP , QUAD , ROD , SEX , SOL , SQUA }. If NEIGHBOR is true, the program looks to see if the regions immediately preceding and following the current region have a field type in the above set and, if so, add the field to the field at the current location.

Example: consider the field in two adjacent 1 m long solenoid regions.
 Allow the fringe fields to overlap.

Overlapping soft edge solenoid fields.
 &cont npart=1 neighbor=.true. /

```
.....
&zhist  nzhist=1 /
1 0. 0.02857 70 0. 0. 33
.....
```

```
SREGION
1.00 1 0.01
1 0. 0.20
SOL
2. 1.0 0.8 0.1 1 0.2 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
SREGION
1.00 1 0.01
1 0. 0.20
SOL
2. 1.0 0.8 0.1 1 0.2 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```



TERMOUT specifies whether output to the terminal is permitted. This is used by some optimization programs.

4. Interaction variables

4.1 Global controls

Information about interactions is entered with the INTS namelist. The interactions with matter used in a simulation are controlled by setting the logical interaction variables given in the following table.

interaction variable	default	level	description
LDEDX	T	DELEV	mean energy loss
LSTRAG	T	STRAGLEV	straggling
LSCATTER	T	SCATLEV	multiple scattering
LDRAY	T	(none)	delta ray production
LDECAY	T	DECLEV	decay
LELMS	F	(none)	scattering & straggling
LSAMCS	F	(none)	scattering & straggling
LINTERACT	F	INTLEV	nuclear interactions
LSPACE	F	SPACELEV	space charge

The level variable is used to select a particular model for a given interaction.

4.2 Energy loss

PDELEV4 gives the value of momentum to use in the level 4 model.

4.3 Multiple scattering

FACFMS is a factor to correct the characteristic angle squared χ_C^2 in the Moliere theory.

FACMMS is a factor to correct the characteristic angle squared χ_A^2 in the Moliere theory.

4.4 Correlated scattering and straggling

There are three methods for simulating correlated scattering and straggling in particle interactions. The first is to turn on delta rays by setting LDRAY=true. Two other variables control the transition energy between continuous, uncorrelated scattering and straggling and correlated delta ray (knock-on electron) interactions.

DCUTE is the kinetic energy of electrons, above which delta rays are simulated discretely.

DCUTM is the kinetic energy of muons and other heavy particles, above which delta rays are simulated discretely.

The second method is to use the ELMS algorithm [3] by setting LELMS=true. This only applies to liquid hydrogen. Interactions in other materials use the models specified by SCATLEV and STRAGLEV.

ELMSCOR determines whether ELMS runs with correlations or not.

The third method is to use the SAMCS algorithm [4].

4.5 Decays

FASTDECAY is a diagnostic variable that forces muons, pions and kaons to decay immediately.

4.6 Nuclear interactions

The following parameters are used for the Wang model of pion production in proton interactions,

$$d^2\sigma/dp d\Omega = A p_{MAX} x (1-x) \exp\{-Bx^C - Dp_T\}$$

where $x = p_L / p_{MAX}$.

WANGA is the Wang A parameter.

WANGB is the Wang B parameter.

WANGC is the Wang C parameter.

WANGD is the Wang D parameter.

WANGPMX is the Wang p_{MAX} parameter.

WANGFMX is the maximum value for the Wang differential cross section.

4.7 Space charge

FRFBUNSC is the RF frequency for space charge model 4.

PARBUNSC is the number of muons per bunch.

5. Internal diagnostics

ICOOL contains a number of internal diagnostics. These can be used to check if the program is behaving “reasonably”. For simple jobs these diagnostics may all that is needed. The postprocessor file information can be used for more detailed analysis. Apart from the region summary table, all internal diagnostics are written to the log file for002.dat.

5.1 Region summary table

ICOOL identifies physical regions and pseudoregions by an integer region number. The region number can be found from a convenient summary table that lists all the regions that have been defined for the job. It is useful to check this table when first setting up a problem to be sure that it has been set up in the intended way. The region summary table is created by setting the control variable SUMMARY true. The data is written to the file for007.dat. The figure shows the beginning of a region summary table.

```

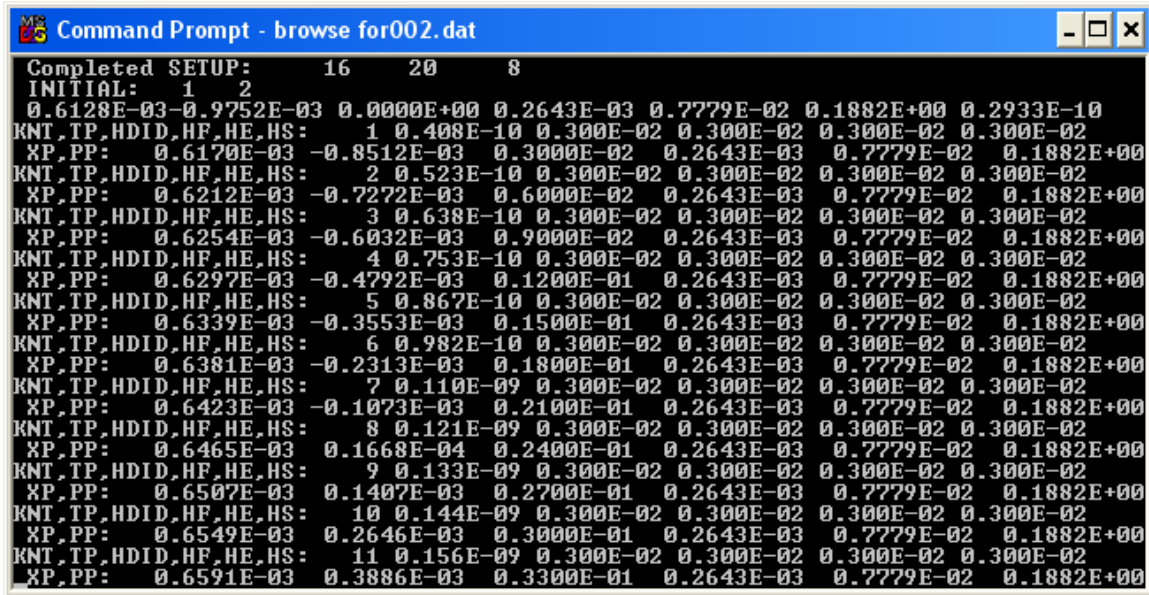
*****
REGION SUMMARY
*****
IZ  IS  IC  CTAG  BTAG  SLEN  ZLO  ZHI  IR  FTAG  MTAG  MGEOM
1   1   1   STUS  NONE  0.30000E+00  0.00000E+00  0.30000E+00  1  NONE  LH   WEDGE
2   1   1   STUS  NONE  0.85000E-01  0.30000E+00  0.38500E+00  1  NONE  UAC  CBLOCK
3   1   1   STUS  NONE  0.33000E+00  0.38500E+00  0.71500E+00  1  ACCE  UAC  NONE
4   1   1   STUS  NONE  0.33000E+00  0.71500E+00  0.10450E+01  1  ACCE  UAC  NONE
5   1   1   STUS  NONE  0.33000E+00  0.10450E+01  0.13750E+01  1  ACCE  UAC  NONE
6   1   1   STUS  NONE  0.33000E+00  0.13750E+01  0.17050E+01  1  ACCE  UAC  NONE
7   1   1   STUS  NONE  0.33000E+00  0.17050E+01  0.20350E+01  1  ACCE  UAC  NONE
8   1   1   STUS  NONE  0.33000E+00  0.20350E+01  0.23650E+01  1  ACCE  UAC  NONE
9   1   1   STUS  NONE  0.85000E-01  0.23650E+01  0.24500E+01  1  NONE  UAC  CBLOCK
10  1   1   STUS  NONE  0.30000E+00  0.24500E+01  0.27500E+01  1  NONE  LH   WEDGE
11  1   2   STUS  NONE  0.30000E+00  0.27500E+01  0.30500E+01  1  NONE  LH   WEDGE
12  1   2   STUS  NONE  0.85000E-01  0.30500E+01  0.31350E+01  1  NONE  UAC  CBLOCK
13  1   2   STUS  NONE  0.33000E+00  0.31350E+01  0.34650E+01  1  ACCE  UAC  NONE
14  1   2   STUS  NONE  0.33000E+00  0.34650E+01  0.37950E+01  1  ACCE  UAC  NONE
15  1   2   STUS  NONE  0.33000E+00  0.37950E+01  0.41250E+01  1  ACCE  UAC  NONE
16  1   2   STUS  NONE  0.33000E+00  0.41250E+01  0.44550E+01  1  ACCE  UAC  NONE
17  1   2   STUS  NONE  0.33000E+00  0.44550E+01  0.47850E+01  1  ACCE  UAC  NONE
18  1   2   STUS  NONE  0.33000E+00  0.47850E+01  0.47850E+01  1  ACCE  UAC  NONE
19  1   2   STUS  NONE  0.33000E+00  0.47850E+01  0.47850E+01  1  ACCE  UAC  NONE
20  1   2   STUS  NONE  0.33000E+00  0.47850E+01  0.47850E+01  1  ACCE  UAC  NONE
*****

```

The region number is given in the first column (labeled IZ).

5.2 Data print out

Data about individual particle tracking can be printed out using the control variables NPRNT, PRLEVEL, and PRNMAX. NPRNT sets the number of tracks to print out. The amount of information is controlled using PRLEVEL. This ranges from only giving values at the end of regions to listing particle and field data after every step inside a region. PRNMAX can be used to limit the number of steps that is printed inside regions.



```
Command Prompt - browse for002.dat
Completed SETUP:      16      20      8
INITIAL:      1      2
0.6128E-03-0.9752E-03  0.0000E+00  0.2643E-03  0.7779E-02  0.1882E+00  0.2933E-10
KNT,TP,HDID,HF,HE,HS:  1  0.408E-10  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6170E-03 -0.8512E-03  0.3000E-02  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  2  0.523E-10  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6212E-03 -0.7272E-03  0.6000E-02  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  3  0.638E-10  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6254E-03 -0.6032E-03  0.9000E-02  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  4  0.753E-10  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6297E-03 -0.4792E-03  0.1200E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  5  0.867E-10  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6339E-03 -0.3553E-03  0.1500E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  6  0.982E-10  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6381E-03 -0.2313E-03  0.1800E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  7  0.110E-09  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6423E-03 -0.1073E-03  0.2100E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  8  0.121E-09  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6465E-03  0.1668E-04  0.2400E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS:  9  0.133E-09  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6507E-03  0.1407E-03  0.2700E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS: 10  0.144E-09  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6549E-03  0.2646E-03  0.3000E-01  0.2643E-03  0.7779E-02  0.1882E+00
KNT,TP,HDID,HF,HE,HS: 11  0.156E-09  0.300E-02  0.300E-02  0.300E-02  0.300E-02  0.300E-02
XP,PP:  0.6591E-03  0.3886E-03  0.3300E-01  0.2643E-03  0.7779E-02  0.1882E+00
```

The figure shows part of the step-by-step print out using NPRNT=1 and PRLEVEL=2.

A message is written in the LOG file FOR002.DAT for each particle that fails to successfully reach the end of the problem. The following information is given in the message. IFLAG is a code indicating the cause of the failure. IEVT is the “event” number. This number increases sequentially up to the number of particles requested for the simulation. IPNUM is the “particle” number. In general this is set to 1. However, when decays are enabled, this number increments by 1 each time the tracked particle decays. IPFLG is a particle-specific code that may be implemented in the future. JSRG is the ICOOL region number where the failure occurred.

5.3 Histograms

Histograms can be produced of particle coordinates, field components, etc. The quantity to be histogrammed is specified using the region number and an ID number from the following table.

ID	variable	ID	variable	ID	variable	ID	variable
1	x	11	x'	21	p _T	31	B _X
2	y	12	y'	22		32	B _Y
3	z	13	θ	23		33	B _Z
4	p _X	14	r	24		34	E _X
5	p _Y	15	φ	25		35	E _Y
6	p _Z	16	p _R	26		36	E _Z
7	ct	17	p _φ	27		37	S _X
8	p	18	L _Z	28	A ²	38	S _Y
9	E	19	L ²	29	r ²	39	S _Z
10	KE	20	s	30	helicity	40	phase

By default histograms are automatically scaled to prevent overflows, although this can be changed using the histogram variable HAUTO. The contents of histograms can be written to the log file using the histogram variable HCPRN.

Example: make a histogram of x at production
 &nhs nhist=1 /
 -0.02 1E-3 40 1 1



The ID number -1 can be used to obtain the production characteristics of tracks that reach the end of the simulation. The ID number may also be set to an ICOOL error flag,

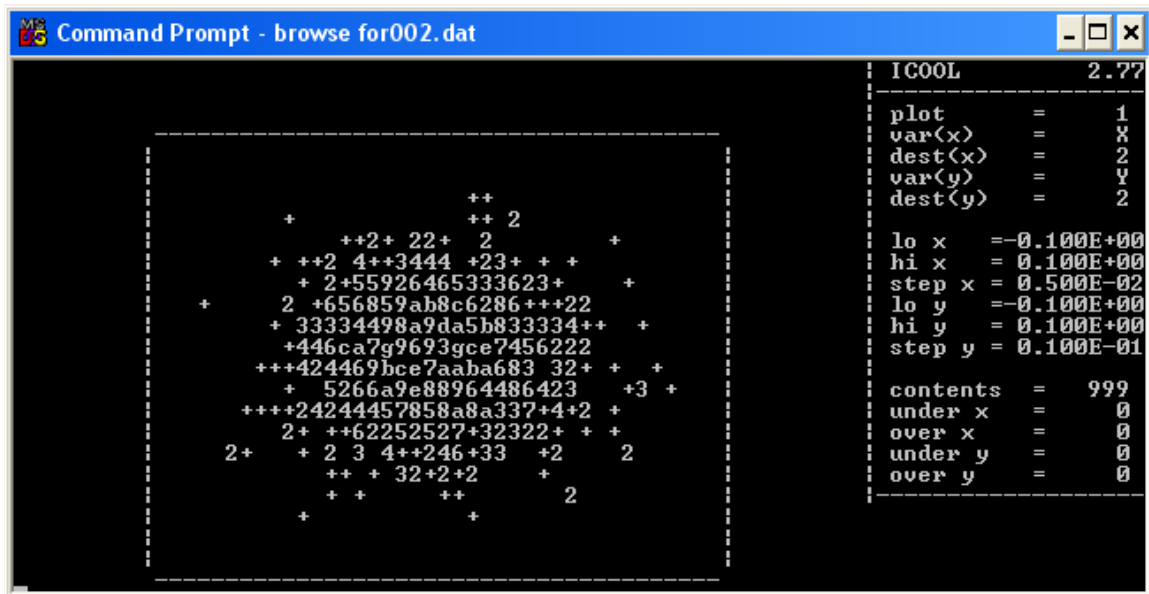
e.g. -23 for particles which are lost due to a radial limit, in order to obtain production characteristics of tracks that failed.

5.4 Scatterplots

Scatterplots can be produced of pairs of particle coordinates, field components, etc. The quantities to be plotted are specified using pairs of variable ID numbers and region numbers. By default scatterplots are automatically scaled to prevent overflows, although this can be changed using the scatter plot variable SAUTO.

Example: make a scatterplot of x versus y at plane 2

```
&nsc nscat=1 /
-0.1 5E-3 40 1 2 -0.1 1E-2 20 2 2
```



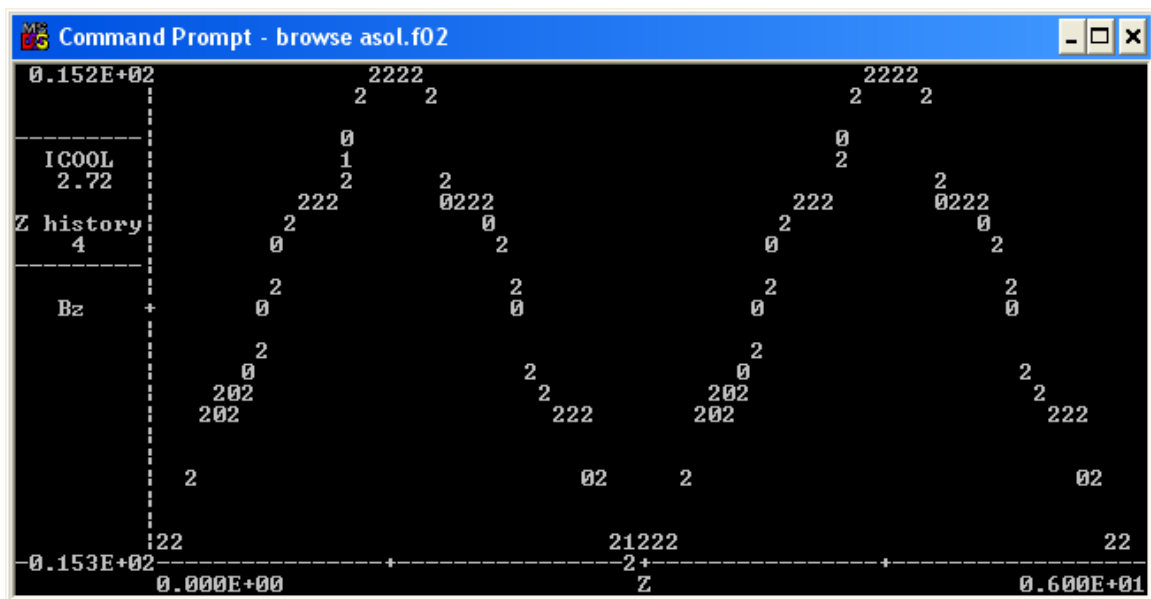
Density is represented by the plot symbol in the sequence {+, 2-9, a-z, A-Z, *}.

5.5 Z-history

A z-history is a plot of a quantity as a function of the longitudinal coordinate z (or s in a curved region). Z-histories can be produced of particle coordinates, field components, etc. The quantity to be plotted is specified using the variable ID number. By default z-histories are automatically scaled to prevent overflows, although this can be changed using the z-history variable ZAUTO. The contents of z-histories can be written to the log file using the z-history variable ZHPRN.

Example: make a z-history of B_z over the first 6 m of a lattice

```
&nzh nzhist=4 /
(first 3 definitions)
2 0. 0.08571 70 0. 0. 33
```



5.6 R-history

An r-history is a longitudinal plot of the minimum, maximum, mean and standard deviation of all particles in a distribution at the end of regions. The quantity to be plotted is specified using the variable ID number. By default r-histories are automatically scaled to prevent overflows, although this can be changed using the r-history variable RHAUTO. The contents of r-histories can be written to the log file using the r-history variable RHPRN.

Example: plot an r-history of the kinetic energy over regions 3 to 210.
&nrh nrhist=1 /
3 3 0. 0. 10



5.7 Emittances

2D emittances can be printed out at selected region numbers. A number of emittance variables {DISCORR, PXYCORR, IPZCOR, SIGMACUT, SIG_CUT, EDETAIL} can be used to control what corrections and correlations are used in the calculations.

Example: compute the 2D emittances for 16 regions. Correct for solenoidal field.

```
&nem          nemit=16          pxycorr=.true.  /
 38  74 110 146 182 218 254 290 326 362 398 434 470 506 542
578
```

Total number of initial good particles =		1000						
Total number of final good particles =		994						
***** Weighted NORMALIZED 2-D EMITTANCES *****								
des	No	N	emitX [m]	emitY [m]	emitZ [m]	Tr	fET	fEL
38	1000.	1000.	0.1582E-02	0.1564E-02	0.1624E-02	1.000	1.000	1.000
74	1000.	996.	0.1498E-02	0.1458E-02	0.1716E-02	0.996	0.940	1.056
110	999.	996.	0.1420E-02	0.1384E-02	0.1626E-02	0.996	0.891	1.001
146	998.	997.	0.1336E-02	0.1334E-02	0.1705E-02	0.997	0.849	1.050
182	998.	993.	0.1256E-02	0.1240E-02	0.1798E-02	0.993	0.793	1.107
218	998.	995.	0.1225E-02	0.1183E-02	0.1811E-02	0.995	0.766	1.115
254	998.	994.	0.1150E-02	0.1134E-02	0.2001E-02	0.994	0.726	1.232
290	997.	990.	0.1093E-02	0.1041E-02	0.2071E-02	0.990	0.678	1.275
326	997.	984.	0.9883E-03	0.9793E-03	0.2011E-02	0.984	0.625	1.238
362	997.	988.	0.9734E-03	0.9505E-03	0.2226E-02	0.988	0.612	1.371
398	997.	988.	0.8993E-03	0.9431E-03	0.2334E-02	0.988	0.586	1.437
434	996.	980.	0.8372E-03	0.8833E-03	0.2282E-02	0.980	0.547	1.405
470	996.	985.	0.8654E-03	0.8177E-03	0.2531E-02	0.985	0.535	1.558
506	995.	981.	0.7761E-03	0.7789E-03	0.2561E-02	0.981	0.494	1.577
542	994.	978.	0.7403E-03	0.7303E-03	0.2521E-02	0.978	0.468	1.552
578	994.	979.	0.7086E-03	0.7060E-03	0.2757E-02	0.979	0.450	1.697

The ICOOL log file contains two sets of emittance calculations. The first set of emittances use 2-dimensional calculations for the x - P_x , y - P_y , and z - P_z phase spaces. The emittance definition removes the slope in each of these 2-dimensional spaces. In addition a control variable (PXYCORR) is provided to compute the emittance using canonical momentum variables inside solenoidal fields (Larmor frame). Another control variable (IPZCOR) is provided to remove the P_z -amplitude correlation in the longitudinal phase space calculation. The 6-dimensional emittance calculated by this method is the product of the three 2-dimensional emittances.

The second set of emittance calculations (“LBNL”) is derived from the computed covariance matrix. The transverse emittances comes from the determinant of a 4-dimensional covariance matrix and the 6-dimensional emittance is the determinant of a 6-dimensional covariance matrix. This method essentially removes all non-diagonal correlations from the final emittance values. In addition, other quantities (e.g. beta functions) following from the covariance matrix are computed

The program computes dispersions statistically as

$$D = r(x, Pz) \sigma(x) \langle Pz \rangle / \sigma(Pz)$$

$$D' = r(Px, Pz) \sigma(Px) / \sigma(Pz)$$

where

$$r(u, v) = (\langle uv \rangle - \langle u \rangle \langle v \rangle) / (\sigma(u) \sigma(v))$$

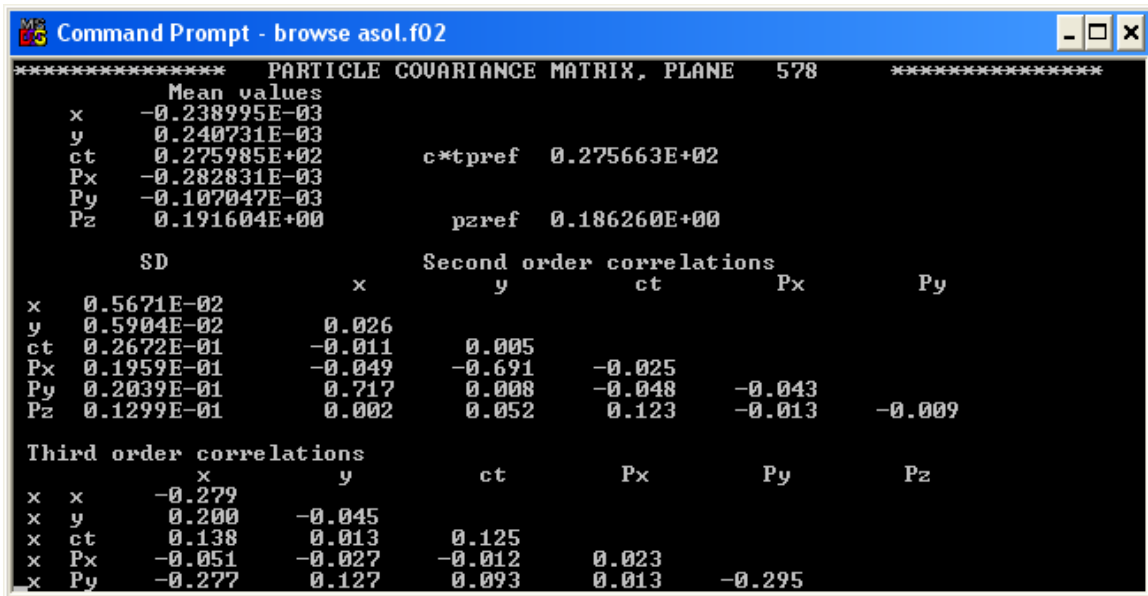
5.8 Covariances

Second- and third-order covariances among the particle variables can be printed out for selected region numbers. The figure shows part of the covariance print out at a given plane.

Example: compute emittances and other correlations from the covariance matrix

```
&ncv ncovar=2 /  
1 578
```

(the figure shows some of the output)



```
Command Prompt - browse asol.f02  
***** PARTICLE COVARIANCE MATRIX, PLANE 578 *****  
Mean values  
x -0.238995E-03  
y 0.240731E-03  
ct 0.275985E+02 c*tpref 0.275663E+02  
Px -0.282831E-03  
Py -0.107047E-03  
Pz 0.191604E+00 pzref 0.186260E+00  
  
SD Second order correlations  
x y ct Px Py  
x 0.5671E-02 x 0.026 y 0.005 ct -0.025 Px  
y 0.5904E-02 -0.011 0.008 -0.048 -0.043 Py  
ct 0.2672E-01 -0.049 -0.691 -0.048 -0.013 -0.009  
Px 0.1959E-01 0.717 0.008 -0.048 -0.013 -0.009  
Py 0.2039E-01 0.002 0.052 0.123 -0.013 -0.009  
Pz 0.1299E-01 0.002 0.052 0.123 -0.013 -0.009  
  
Third order correlations  
x y ct Px Py Pz  
x x -0.279 y ct Px Py Pz  
x y 0.200 -0.045 ct Px Py Pz  
x ct 0.138 0.013 0.125 ct Px Py Pz  
x Px -0.051 -0.027 -0.012 0.023 ct Px Py Pz  
x Py -0.277 0.127 0.093 0.013 -0.295 ct Px Py Pz
```

5.9 Magnetic field maps

ICOOL can copy an internally created magnetic field map to an external file. This can be done for 2D maps using the GRID command or by using SHEET model 3. An internally created 3D map generated from the BACKGROUND command can also be written to an external file.

5.10 RF diagnostics

Diagnostic information about particle and field variables in RF cavities can be generated by using the control variable RFDIAG.

6. Pseudoregion commands

6.1 Simulation

APERTURE imposes a rectangular or elliptical constraint on all particles transverse coordinates at this location.

Example: insert a symmetric rectangular collimator with a half-opening of 15 cm in x.

```
APERTURE
2   -0.15  0.15  1
```

CUTV applies a cut on an ICOOL variable.

Example: only accept particles with p_z greater than 200 MeV/c.

```
CUTV
6   2   0.200
```

DENP applies a transverse density profile.

Example: associate a quadratic density profile along the x direction with beryllium material.

```
DENP
BE   1   0.5  0.3  0.1  0.
```

DENS adjusts the density of a material.

Example: use gaseous hydrogen at a pressure of 20 atm as a material.

```
DENS
GH   20.
```

DVAR makes an additive change to one of the particles variables.

Example: offset all particles x coordinates by -1 cm. Variables are specified using ICOOL's internal variable numbering, e.g. 1 means x. The third parameter controls whether the change also affects the reference particle.

```
DVAR
1  -0.01  0
```

EDGE takes into account fringe field focusing for hard-edge magnet models.

Example: model a 1.2 T hard-edge solenoid including effects of the end fields.

```
EDGE
SOL
1 -1.2  0.  0.  0.
SREGION
1.00  1  0.01
1  0.00  0.30
SOL
1  0  0  0  1.2  0  0  0  0  0  0  0  0  0  0
VAC
CBLOCK
0 0 0 0 0 0 0 0 0 0
EDGE
SOL
1 1.2  0.  0.  0.
```

REFP specifies reference particle parameters.

Example: define a 200 MeV/c muon that moves with constant velocity as the reference particle.

```
REFP
2 0.200  0.  0.  3
```

REF2 specifies the parameters for a second reference particle. This is only used for the ACCEL field model 10 for the adiabatic buncher.

RESET changes the times of all particles to the current value of the reference particle time.

ROTATE changes the angles of the coordinate system.

Example: bend particles vertically by rotating coordinates 90° around z, followed by a normal 0.2 T dipole field.

```
ROTATE
90.  3  0
SREGION
1.00  1  0.01
1  0.00  0.30
DIP
1  0.2  0  0.200  0  0  0  0  0  0  0  0  0  0
VAC
CBLOCK
0 0 0 0 0 0 0 0 0 0
```

TAPER is used to set up a tapered lattice. It is used in conjunction with the BLOCK(4) and BSOL(5) field models.

TRANSPORT performs a matrix transformation on all particles.

6.2 Postprocessor file

OUTPUT enables writing of particle information to for009.dat in the following region.

Example: generate postprocessor output for the next region

```
OUTPUT
SREGION
0.20 1 0.01
1 0. 0.30
QUAD
1. 1.0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

6.3 Field grid

GRID defines a new magnetic field grid. This can be done either by reading in a file of current elements and computing the field or by reading in an existing field grid.

Example: define a new field grid #2 using the data in an external file of current sheets, e.g. for032.dat. Let the grid have 1 cm spacing in z and r and extend over 3 m in z and 30 cm in r. Save the field values on the grid in file for043.dat.

```
GRID
2
SHEET
0 32 0.01 0.01 3.01 0.31 10. 0 43 0 1. 0 0 0 0
```

Example: define a new field grid #3 using an existing grid in the external file for050.dat.

```
GRID
3
SOL
0 50 0. 0. 0 0. 0. 0 0 0 1. 0 0 0 0
```

6.4 Error analysis

DISP randomly displaces particle transverse coordinates.

TILT randomly rotates all particles' coordinates in 3D.

RKICK makes random magnetic kicks on all particles.

Example: apply random dipole kicks to all particles at this location. Let the standard deviation of the error field be 0.1 T and let the kick be applied with random azimuthal orientations.

```
RKICK
DIP 0. 0.1 0 -1
```

6.5 Background fields

The background field construct sets up a static magnetic field on a 3-D cartesian grid. A new definition is begun using the BACKGROUND command. The definition is terminated with the ENDB command. One of the parameters ZTOTALBKG specifies the total incremental distance from the present location that this background can be used. Following the BACKGROUND command the user can enter a series of BFIELD commands, which are used to construct the field on the background grid. Two of the

parameters of this command, ZMINBKG and ZMAXBKG, specify the region of the grid where this field will be used. In general ZMAXBKG = ZTOTALBKG. The field type specified by the BFIELD command can either be an internally defined ICOOL field or STUS, which allows a user-specified static field to be read in from a file. In order for higher order interpolation to work properly, it is necessary to make the background grid at least 1 grid point larger than the region where you actually want to track particles. Otherwise the particles will see a zero field at the last grid point.

The background field can be used in two ways. First the field can be defined as a sum of predefined ICOOL fields. Second the user can read in an external file containing a 3D field grid. The background field, if any, is superposed on the ICOOL cell and region fields.

BACKGROUND begins a background field definition.

BFIELD defines a field contribution to the total background field.

ENDB ends the definition of a background field.

Example: define a new field for the following 3 m consisting of a 0.5 T solenoid and a 0.2 T/m quadrupole field.

```
BACKGROUND
.false. 0.200 3.00
-0.20 0.02 21 -0.20 0.02 21 0. 0.02 151 2
BFIELD
0. 0.20 0.00 3.00
SOL
1 0 0 0 0.5 0 0 0 0 0 0 0 0 0 0
BFIELD
0. 0.20 0.00 3.00
QUAD
1 0.2 0 0 0 0 0 0 0 0 0 0 0 0 0
ENDB
```

Example: use an external field grid, e.g. for044.dat, as a background field for the next 3 m.

```
BACKGROUND
.false.  0.200  3.00
-0.20  0.02  21  -0.20  0.02  21  0.  0.02  151  2
BFIELD
0.  0.20  0.00  3.00
STUS
1  44  0  0.200  1.  2  0  0  0  0  0  0  0  0  0
ENDB
```

6.6 Miscellaneous

DUMMY inserts a dummy placeholder in a job. This can be used to reserve a place for an OUTPUT command without changing the region numbers used for ICOOL's internal diagnostics.

7. Field description

ICool uses three methods for specifying the field environment of the tracked particles. (1) Each region has a field associated with it. For simple problems this may provide a sufficient description of the field. (2) Groups of regions can be associated together into cells. Each cell has a field associated with it that applies to all the regions in the cell. This can be used for example to describe RF cavity regions inside a solenoid field over the cell. (3) A background field grid can be defined over a specified region of s from any combination of region fields or from a user-supplied field grid. This can be used, for example, to build a complicated field pattern or to apply error fields to some part of a lattice. The actual field seen by a particle at a given location is the superposition of all of these component fields.

7.1 Hard-edge fields

Hard-edge fields drop abruptly to 0 at the ends. The simplest hard-edge fields are uniform longitudinally over the entire length of the region. These are listed in the following table.

Longitudinally uniform hard-edge fields		
type	description	model
ACCEL	RF cavity	1 (mode=0)
BROD	bent axial rod	1
BSOL	bent solenoid	1
DIP	dipole	1, 3
EFLD	electric field	1
HDIP	horizontal dipole	1
QUAD	quadrupole	1
ROD	axial current rod	1
SEX	sextupole	1
SOL	solenoid	1
SQUA	skew quadrupole	1

Example: 20 cm long, 1 T/m quadrupole field

```

SREGION
0.20 1 0.01
1 0. 0.30
QUAD
1. 1.0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

```

There is in addition a hard edge dipole model with rotated pole faces.

Longitudinally non-uniform hard-edge fields		
type	description	model
DIP	dipole	4

Example: 20 cm long, 1 T dipole with 15° rotated pole faces centered in a 40 cm long region

```

SREGION
0.40 1 0.01
1 0. 0.30
DIP
4. 1.0 0. 0.200 0. 0. 0.20 0.10 15. 15. 0.10 0.10 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

```

7.2 Linear longitudinal variation

There are several simple models where the longitudinal field variation is piecewise linear.

Longitudinally non-uniform hard-edge fields		
type	description	model
FOFO	alternating solenoid	1
ROD	axial current rod	2, 3
SOL	solenoid	1

Example: 1 m long, 2 T solenoid with 10 cm end regions

```

SREGION
1.00 1 0.01
1 0. 0.30
SOL
1. 2.0 0.80 0.10 0. 0.10 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

```

7.3 Soft edge fields

Soft edge fields in ICOOL use a $\Delta \tanh(s)$ longitudinal field dependence.

Soft-edge fields		
type	description	model
BSOL	bent solenoid	2
DIP	dipole	2
HDIP	horizontal dipole	2
QUAD	quadrupole	2
ROD	axial current rod	4
SEX	sextupole	2
SOL	solenoid	2
SQUA	skew quadrupole	2

Example: 60 cm long, 1 T solenoid with 10 cm symmetric fall-off at the ends

```

SREGION
1.00 1 0.01
1 0. 0.10
SOL
2. 1.0 0.60 0.20 3. 0.10 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

```

7.4 Current elements

ICOOL can produce a field from a set of current elements.

Fields from current elements		
type	description	model
BLOCK	current block	1, 2
BLOCK	tapered current	4
COIL	current loop	1, 2
SHEET	current sheet	1 - 4
SOL	solenoid	3 - 5
HELIX	helical winding	2, 3

Example: field from 20 cm long, 30 cm radius cylindrical current block

```
SREGION
1.00 1 0.01
1 0. 0.30
BLOCK
1. 0.40 0.30 0.40 0.20. 100. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

Example: field from an external file, e.g. for022.dat, containing sheet parameters. Prepare a grid and interpolate field values from the grid.

```
SREGION
1.00 1 0.01
1 0. 0.30
SHEET
3. 22 0.01 0.01 1.01 0.31 10. 2. 0. 0. 1. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

7.5 B on-axis

The field in a region can be determined if either the field or multipole coefficients are given on-axis.

Fields from B on-axis		
type	description	model
SOL	solenoid	9
BSOL	bent solenoid	3, 4
BSOL	tapered lattice	5
HELIX	helix	4

Example: combined dipole plus solenoid field from an external file, e.g. for023.dat, of on-axis multipole coefficients. Use as a cell field.

```
CELL
1
.false.
BSOL
4 23 0.200 4 0.80 1 1 1 1 1 1 1 1 1 1 0
.....
ENDC
```

7.6 Field grids

The field may be taken from a field grid in an external file. The first four types give essentially equivalent ways to input a 2D r-z field grid.

Field grids		
type	description	model
BLOCK	r - z	3
COIL	r - z	3
SHEET	r - z	5
SOL	r - z	6
BROD	x - y	1
STUS	x - y - s	2

Example: get the field from an r-z grid that has been previously stored in grid #3 using a GRID command.

```
SREGION
1.00 1 0.01
1 0. 0.30
SHEET
5. 3 2 0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

Example: get the field from a curved x-y-s grid that has been previously stored in grid #1 using a GRID command.

```
SREGION
1.00 1 0.01
1 0. 0.30
STUS
2. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

7.7 Miscellaneous static fields

Other static fields		
type	description	model
FOFO	alternating solenoid	2
HELIX	helix	1
HORN	toroidal horn	1, 2
WIG	wiggler	1, 2
ROD	axial current rod	5
SOL	solenoid	7

Example: 10 cells of an alternating solenoid lattice with peak field of 1 T and magnetic period of 1 m. Use as a cell field.

```

CELL
 10
.true.
FOFO
2 1.0 0. 1.00 0. 0 0 0 0 0 0 0 0 0 0
.....
ENDC

```

7.8 Time-varying fields

Time-varying fields		
type	description	model
ACCEL	E_z only	1 (mode 1)
	RF cavity	2 - 4, 9 - 13
	induction linac	6 - 8
	external RF file	5
KICK	kicker	1, 2, 3

Example: 201 MHz pillbox RF cavity

```

SREGION
0.30 1 0.01
1 0. 0.55
ACCE
2. 201. 10 0 0 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
VAC
CBLOCK
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

```

8. ICOOL utilities

We list here some of the useful utility programs that operate on ICOOL files.

8.1 ERRSUM2

This program creates a summary of error messages by reading the ICOOL log file for002.dat. It displays a short summary on the terminal and writes a more detailed description to the file errsum2.dat.

8.2 ECALC9F

This is the standard emittance calculation program for ICOOL. It was written by Gregg Penn. It reads the ICOOL file for009.dat. The calculation is controlled with the input file ecalc9f.inp. The output of the program is written to the file ecalc9f.dat. ECALC9F assumes azimuthal symmetry in the fields and is most suited to solenoidal channels. Two variations of the code, ECALCXY and EMITCALC, are more suited for channels containing dipoles or quadrupoles.

8.3 ENDOF9

This program reads the ICOOL file for009.dat and produces a new file endof9.dat in the for009 format that only contains information from tracks that make it to the end of the simulation.

8.4 NOTEND9

This program is the inverse of ENDOF9. It reads the ICOOL file for009.dat and produces a new file notend9.dat in the for009 format that only contains information from tracks that do not make it to the end of the simulation. This could be useful, for example, in understanding the acceptance of a channel.

8.5 EXTPAR9

This program reads the ICOOL file for009.dat and produces a new file extpar9.dat in the for009 format that only contains information on specified tracks. The calculation is controlled with the input file extpar9.inp.

8.6 EXTREG9

This program reads the ICOOL file for009.dat and produces a new file extreg9.dat in the for009 format that only contains information at specified regions. The calculation is controlled with the input file extreg9.inp.

8.7 BUNCH9

This program reads the ICOOL file for009.dat. It produces a short summary of the amount of bunching on the terminal and a more detailed analysis in the file bunch9.dat. Several auxiliary files are also produced. The calculation is controlled with the input file bunch9.inp.

9. Other useful utilities

9.1 HistoRoot

The HistoRoot program, which is distributed in conjunction with G4Beamline [5], can be used to examine for009 output.

10. References

- [1] R.C. Fernow, ICOOL: A simulation code for ionization cooling of muon beams, Proc. 1999 Particle Accelerator Conference, New York, 1999, p. 3020.
- [2] R.C. Fernow, Recent developments on the muon-facility design code ICOOL, Proc. 2005 Particle Accelerator Conference, Knoxville, 2005, p. 2651.
- [3] W. Allison et al, *Ab initio* liquid hydrogen muon cooling simulations with ELMS, J. Phys. G: Nucl. Part. Phys. 34 (2007) 679-685.
- [4] Sergei Striganov, On the theory and simulation of multiple Coulomb scattering of heavy charged particles, FERMILAB-Conf-04/056-AD, April 2004.
- [5] T. Roberts, G4Beamline User's Guide, January 2010.