Trak 8.0

Electron/Ion Gun Design and Charged-particle Dynamics

PO Box 13595, Albuquerque, NM 87192 U.S.A.
Telephone: +1-505-220-3975
Fax: +1-617-752-9077
E mail: techinfo@fieldp.com
Internet: http://www.fieldp.com

Contents

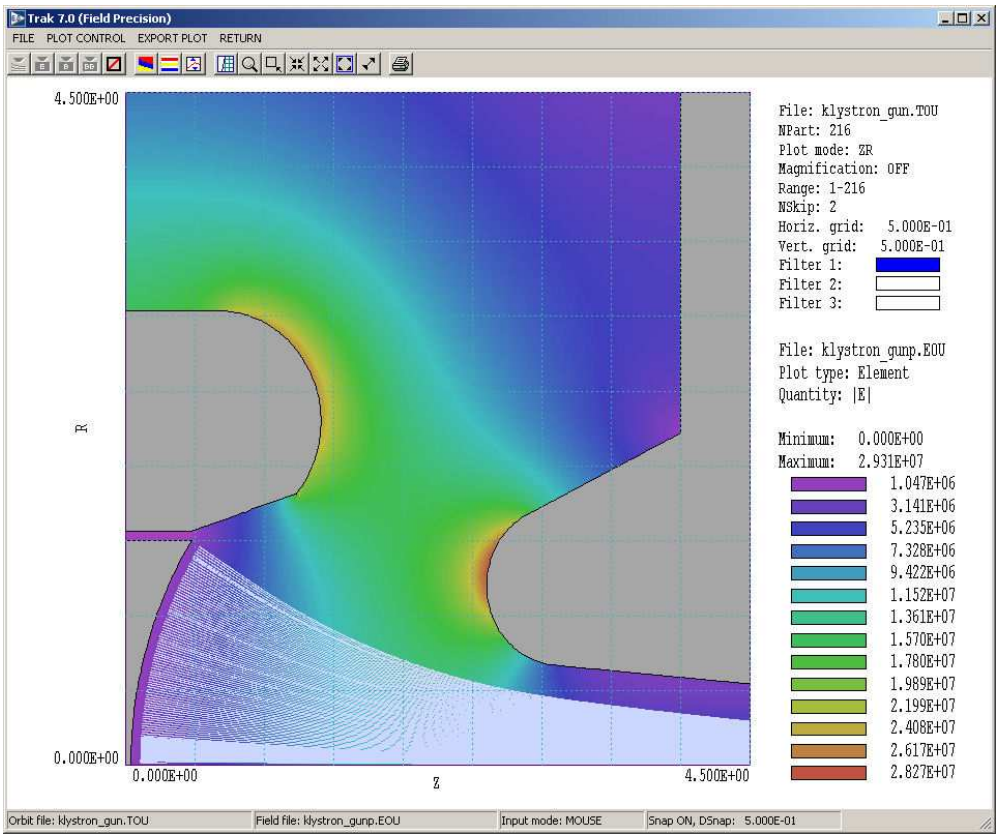

Figure 1: Screenshot – **Trak** in the orbit plotting mode.

1 Introduction

1.1 Program functions

Trak is a versatile program for charged-particle optics. Applications include electron and ion guns, particle accelerators, ion sources, microwave sources, acceleration columns, electrostatic and magnetostatic lenses, vacuum tubes, and electro-optical devices. **Trak** operates in conjunction with field solution programs of the **TriComp** series. The program requires **Mesh** (the **TriComp** universal mesh generator) along with **EStat** and/or **PerMag** to create field input files. You should be familiar with mesh generation and field solution procedures before using **Trak**. The package includes **GenDist**, a utility for creating and analyzing particle distributions.

This manual concentrates on procedures to run **Trak** and does not give detailed explanations of the physics of charged-particle beams. Two texts written by the program author are supplied with the package. They provide a comprehensive overview of charged-particle optics:

- S. Humphries, Jr., **Principles of Charged Particle Acceleration** (Wiley, New York, 1986).

- S. Humphries, Jr., **Charged Particle Beams** (Wiley, New York, 1990).

The second work discusses beam distributions, emittance, beam-generated fields, propagation of high-current beams, the design of electron and ion guns and other topics. Techniques for electric and magnetic field solutions on a conformal triangular mesh are discussed in

S. Humphries, Jr., **Field Solutions on Computers** (CRC Press, Boca Raton, 1997).

1.2 Learning Trak

Trak is one of the most powerful tools available for the designing charged-particle guns and transport devices. Capabilities include mixed source and space-charge-limited emission, self-consistent field emission, advanced techniques for magnetic field created by relativistic beams, and determination of an ideal plasma meniscus shape for ion emission. The extensive features call for a rich command language and a thick instruction manual. To be realistic, it is unlikely that you can use the program effectively without reading some of the manual. On the other hand, we have included many features to make the learning process as painless as possible:

- This manual is available from within the program. There is an index and advanced topics are clearly delineated.
- You can use interactive dialogs to set up program inputs for most calculations. It is easy to add refinements and changes later.
- We included input files for a variety of prepared examples. They cover the full spectrum of charged-particle applications and may serve as a template for your application.

To learn the program quickly, we suggest the follow sequence of activities:

- Read the next section carefully to understand the basic organization of a **Trak** simulation and the associated data files.
- Follow the walkthrough example of Chap. 2. The exercise introduces the fundamental tools you will need for you own simulations.
- Browse Chaps. 3 and 4. Chapter 3 describes how to use interactive dialogs to prepare runs, while Chap. 4 covers commands in the main Trak menu to control a calculation.
- Scan Chapter 5, which describes **Trak** features for plots of orbit and field information.
- Run some of the supplied examples. Try making small small changes to the geometries, material properties and particle distributions.
- As you gain experience, you will want to employ advanced capabilities. For example, you can include multiple emission surfaces in a calculation or set some materials to act as secondary-electron emitters. Advanced features are invoked by adding commands directly to the input script.

- Chapters on advanced features are organized by topic. Chapters 6 and 7 cover script commands to load and to control electric and magnetic field information. For example, you can add specified temporal modulations to fields.

- Chapters 8 through 14 describe advanced techniques for the calculation modes of **Trak**. Chapter 8 covers single-particle tracking in specified fields. Chapter 9 documents commands for precision tracing of electric field lines for applications such as ion mobility spectrometry.

- Chapter 10 summarizes commands for an advanced program mode where the space-charge of high-current beams contributes to the total electric field solution. In this mode, **Trak** accurately models self-consistent Child-law emission from cathodes. Chapter 11 applies to high-current relativistic beams where contributions from particles and current on electrodes are combined to find both self-consistent electric and magnetic fields.

- Chapter 12 covers self-consistent field emission of electrons for applications to vacuum microelectronics.

- Chapter 13 describes the unique **Trak** capability to directly calculate the ideal beam current and emission surface shape for ion extraction from a free plasma boundary.

- Chapter 14 describes commands to control electron secondary emission from metal surfaces. The capability is useful to model electron collectors and multipactor effects in high-power microwave sources.

- Finally, Chapter 15 summarizes advanced diagnostic capabilities of the code, including statistical analysis of beam distributions.

- The **Trak** package includes **GenDist**, a utility for preparing particle input files for **Trak** and other Field Precision programs. **GenDist** also has extensive routines for statistical analysis and plots of distributions created by **Trak**. The features of the program are described in a separate manual.

1.3 Program organization

Trak calculations generally involve the following steps:

1. Define the system geometry for an electric and/or magnetic field solution. The easiest approach is to use the drawing editor of **Mesh**. You can also use a text editor for direct input of boundary vectors. This activity creates **Mesh** input files with names of the form `FPrefix.MIN` for the electric and/or magnetic field solution.

2. Use **Mesh** to process the vector information to generate one or two conformal triangular meshes recorded in files with names of the form `FPrefix.MOU`. **Trak** can handle independent meshes for electric and magnetic fields that overlap a common region of space.

3. Define the properties of materials in the solution using the script-generation dialogs of **EStat** and/or **PerMag**. The resulting files have names of the form `FPrefix.EIN` and/or `FPrefix.PIN` for the solution programs.

Table 1: **Trak** files

Name	Function
`MName.MIN`	**Mesh** input script for the electric and/or magnetic field solution
`MName.MOU`	**Mesh** output (node locations and element identifies)
`EName.EIN`	**EStat** input script (run control and material properties)
`EName.EOU`	**EStat** output file (node locations and electrostatic potential)
`BName.PIN`	**PerMag** input script (run control and material properties)
`BName.POU`	**PerMag** output file (node locations and vector potential)
`TName.TIN`	**Trak** script (program control, field sources, particle input parameters)
`Name.PRT`	Optional **Trak** input file of initial particle parameters
`TName.TLS`	**Trak** listing (assorted quantitative information on the run sequence and particle properties)
`TName.TOU`	**Trak** orbit file (particle positions to create plots)
`Name.PRT`	Optional **Trak** output file listing particle parameters at a stopping or diagnostic plane

4. **Run EStat** and/or **PerMag** to create input field solutions. The solution files `FPrefix.EOU` and `FPrefix.POU` contain complete mesh information as well as values of electrostatic or vector potential. **Trak** uses the information to compute **E** or **B** along particle trajectories. The program can also use the mesh information to determine if a particle strikes a material object (*i.e.*, stopping conditions).

5. Decide on a calculation mode and use the interactive dialogs in **Trak** to prepare an input script `FPrefix.TIN` to control the run.

6. Run **Trak** to determine particle trajectories and (optionally) electric field modifications to include contributions of space-charge.

7. Use **Trak** to create orbit/field plots or **GenDist** to analyze output particle distributions.

The procedure involves many steps and many files – it may appear overly complex for simple benchmark calculations. The step-by-step approach with records of intermediate results rewards your effort in real-world applications. In the end, you have a complete record of all components of the solution, making it easy to reconstruct or to modify simulations. For orientation, Table 1 lists the file types that may appear in the **Trak** calculation. Note that the suffixes indicate the file function.

1.4 Some calculation basics

Trak calculates the orbits of point charged particles moving through electric and magnetic fields. These fields may result from external charges and currents (such as the charge induced on biased metal electrodes), surface charges in nearby dielectrics, or current in magnet coils. Such fields are called *applied fields*indexapplied fields. For low-intensity beams, it is sufficient to determine the applied fields and then to track any number of particle orbits in space. We use the term *single-particle orbits*indexsingle-particle orbits when the fields created by the charge and effective current of the particles are negligible. In this case, particle orbits are independent of each another and it is unnecessary to modify the applied field to reflect the presence of the beam.

Field evaluations and orbit calculations in **Trak** are entirely numerical. Field calculations employ the finite-element method. Here, a bounded spatial region is divided into small segments - the **TriComp** programs use elements with triangular cross sections. The sizes and shapes of the triangles are adjusted to fit the boundaries of physical objects like electrodes. The spatial distribution of the set of triangle nodes (vertices) is called the *computational mesh*. In electrostatic calculations with **EStat**, the potential is determined at the nodes and the material characteristics (such dielectric constant, space charge, or constant potential) are assigned to the triangle volumes. A **Trak** run may also include an independent applied magnetic field solution created with **PerMag**. The solution may have a different computational mesh fitted to the boundaries of magnetic objects (such as coils or ferrites). The magnetic solution consists of values of vector potential at the nodes. A single-particle orbit calculation is conceptually simple. The orbit advances in small time steps. At any time, the program must identify the element that contains the particle. With this knowledge it is possible to collect potential values at neighboring nodes and take a spatial derivative to estimate electric fields. Similarly, a knowledge of the occupied triangle in the magnetic mesh leads to the magnetic fields at the particle position. Given the fields and the initial particle momentum, we can advance the orbit. We can also check whether the particle is outside the mesh or in a non-vacuum element in order to stop the calculation. A **Trak** calculation always follows an **EStat** and/or **PerMag** solution. Even if there is no applied field, we need to create a computational mesh for the calculation of beam-generated fields.

TriComp solution programs handle fields that vary in two dimensions in either cylindrical structures (variation in r and z, uniform in θ) or planar structures (variation in x and y, infinite length along z). **Trak** computes particle orbits in three-dimensional Cartesian coordinates (x, y, z) using the two-dimensional field components available from the solution programs. For rectangular problems, the non-zero field components are E_x and E_y or B_x and B_y. Cylindrical solutions yield the components E_r and E_z or B_r and B_z. The radial components are used to derive the Cartesian field components according to:

$$E_x(x, y, z) = E_r(r, z)(x/r), \tag{1}$$
$$E_y(x, y, z) = E_r(r, z)(y/r), \tag{2}$$
$$B_x(x, y, z) = B_r(r, z)(x/r), \tag{3}$$
$$B_y(x, y, z) = B_r(r, z)(y/r). \tag{4}$$

Orbit calculations are more difficult for high-current beams. In this case, the space-charge of the beam can contribute to electric fields in the propagation region and affect the distribution

of image charge on surrounding electrodes. For this reason, **Trak** has the capability to update the electrostatic field using information on the space-charge density associated with the particle flow. This procedure is called a *self-consistent* orbit calculation. A beam is represented by a collection of model particles, each carrying a fraction of the beam current, δI. At each time step δt, the space-charge of the element occupied by a ray is augmented by an amount $\delta I \delta t$. **Trak** then recalculates the electric field with the extra space-charge. The problem with this approach is that the final orbits of the particles need not be the same as those used to calculate the space-charge. The resolution is to iterate for several cycles using suitable space-charge averaging. This procedure usually converges to the correct solution, even for intense beams.

High-current charged-particle beams have little effect on solenoid-type magnetic fields (B_z and B_r in cylindrical problems, B_x and B_y in planar geometry). Therefore, **Trak** does not modify the applied magnetic field solution. On the other hand, relativistic beams generate magnetic field components B_θ (cylindrical) or B_z (rectangular) that may strongly influence particle dynamics. To address this problem, **Trak** has the capability to find spatial variations of beam current and beam-generated magnetic fields that are recorded on the electric-field mesh (*RelBeam* tracking mode).

1.5 Tracking mode

A **Trak** calculation is performed in one of the following six calculation modes:

- **TRACK**. Single-particle tracking in applied electric and magnetic fields with no beam-generated components. Particles may be generated from a user-specified list or automatically along an emission surface on a region boundary. Application to low-current or neutralized beams.

- **FLINE**. Tracing field lines in an electrostatic solution from **EStat**. A common application is computation of charged-particle flow lines in a resistive medium.

- **SCHARGE** Non-relativistic high-current electron and/or ion beams with self-consistent space-charge effects. Particle generation from a user-specified list and/or Child-law emission surfaces.

- **RELBEAM** Relativistic high-current electron and/or ion beams with self-consistent effects of beam-generated electric and magnetic fields. Particle generation from a user-specified list and/or Child-law emission surfaces.

- **FEMIT** Electron field emission with self-consistent beam-generated fields. Particle generation from a user-specified list and/or an emission surface that follows the Fowler-Nordheim equation. Applications to vacuum microelectronics.

- **PLASMA** High-current ion beam generation from free plasma surfaces, Particle generation from a user-specified list and/or Child-law emission surfaces. The shape of the emission surface is automatically corrected to ensure uniform ion flux.

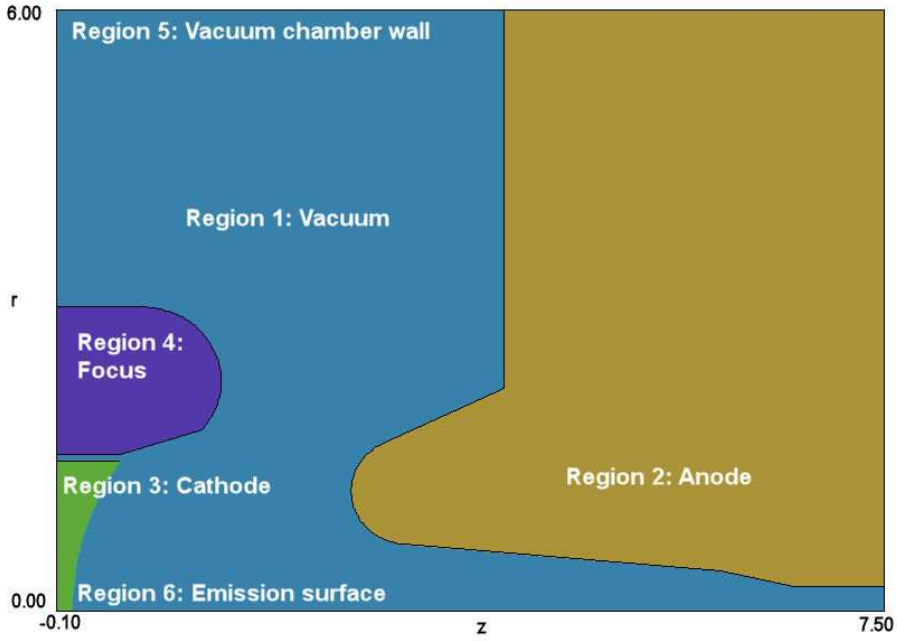

Figure 2: Region definitions for the KLYSTRONGUN example. Dimensions in inches.

2 Application example

2.1 Setup

In this chapter we shall step through a **Trak** application. The calculation determines the performance of a high-intensity, relativistic klystron gun. The gun is designed for strong beam convergence and therefore represents a challenge to the accuracy of a numerical code. Figure 2 shows the geometry, a figure of revolution about the z axis along the bottom. An electron beam of current 460 A is extracted from a spherical-section cathode across a 660 kV acceleration gap. The simulations involve only an applied electric field – we could add a magnetic field to investigate beam matching into a solenoid.

We shall use the following programs: `tc.exe`, `mesh.exe`, `estat.exe` and `trak.exe`. Make sure that you have created a data directory such as `\tricomp\buffer` and the `tc.exe` has the correct settings for the program and data directory. Move the file `KLYSTRONGUN.MIN` to the data directory.

2.2 Defining the geometry

We shall work from an existing **Mesh** input script. Run **Mesh**, click on the command *File/Load.Script (MIN)* and choose the file `KLYSTRONGUN.MIN`. Pick the command *Edit script/Graphics* to enter the drawing editor. Here, you can view the geometry and confirm the region assignments. Click the *Foundation display* command. Note the fine mesh resolution near the axis for a good representation of electric fields within the converged beam. Figure 3 shows a detail of

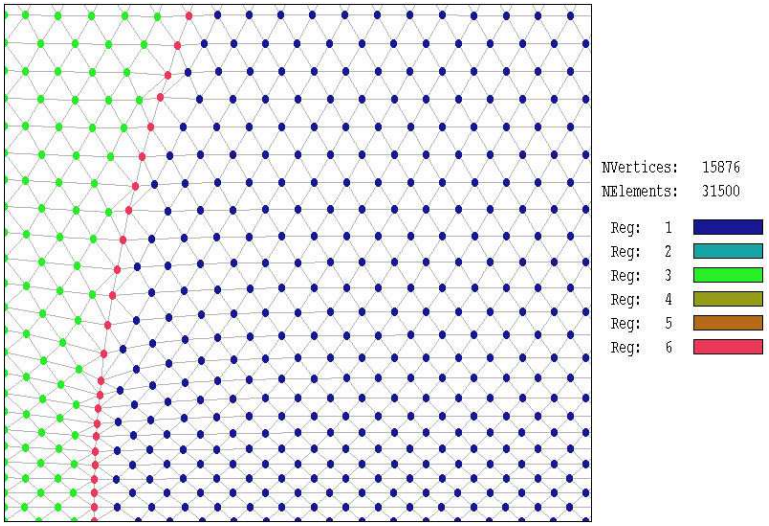

Figure 3: Detail of the mesh on the axis near the cathode for the KLYSTRONGUN example showing node identities.

the mesh at the axis near the cathode. The main difference from a standard input mesh for **EStat** is the presence of Region 6, an open region that covers the surface of the cathode (nodes marked in red in Fig. 3). The associated nodes have the same potential as those of the cathode (Region 3); therefore, they do not affect the electrostatic solution. The marked nodes are used in **Trak** to identify surface facets that will act as emission sites for electrons.

Abandon the drawing to return to the main menu. Click the *Process* command to create the mesh. You can use the plot functions of **Mesh** to inspect the completed conformal mesh. Be sure to click the *Save mesh (MOU)* command to create the file KLYSTRONGUN.MOU.

2.3 Generating the applied field

The first task to create an input electrostatic solution is define control parameters and material properties. Run **EStat** and choose the *Setup* command. Choose the file KLYSTRONGUN.MOU. The program displays the dialog of Fig. 4. Fill in the values as shown. In the *Control parameter* group, we accept most of the defaults. The geometry is set to *Cylindrical* and the unit conversion parameter is set to $DUnit = 39.37$ inches/meter. The material properties in the grid are simple. Region 1 (vacuum) is a dielectric with $\epsilon_r = 1.0$. Regions 2 (anode) and 5 (vacuum chamber) are at ground potential, while the electrode regions of the cathode assembly (Region 3, 4 and 6) are at -660.0 kV. Click *OK* and accept the default name to create the file KLYSTRONGUN.EIN.

Choose the *Solve* command. **EStat** takes a few seconds to find the electrostatic solution and saves the file KLYSTRONGUN.EOU. You can use the plot functions of *EStat* to investigate the solution. Later, **Trak** will create another electrostatic solution file, KLYSTRONGUNP.EOU, that includes the effects of the beam space charge.

2.4 Creating the Trak control script

We can now turn to the main task of generating a beam solution. Run **Trak** and pick the *Setup* command. The dialog gives a choice of run mode. Electrons with 660.0 keV kinetic

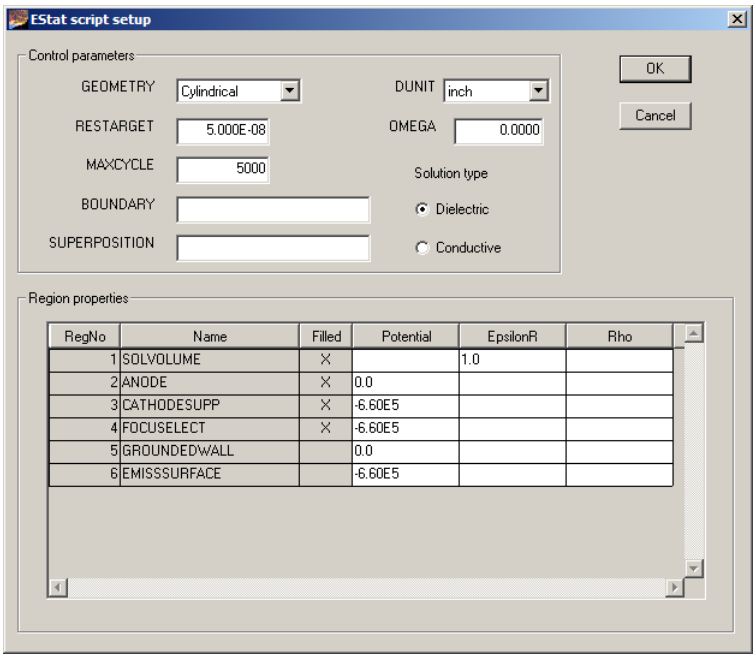

Figure 4: Setting parameters for the **EStat** solution – `KLYSTRONGUN` example.

energy have a relativistic energy factor $\gamma = 2.29$, significantly larger than unity. Therefore, beam-generated magnetic fields play a significant role and we must choose the *RelBeam* option in the initial dialog. After you click OK, the program displays the setup dialog of Fig. 5. Fill in the values as shown. In the *Fields* section we specify that the program should load mesh and initial electric field information from `KLYSTRONGUN.EOU`. The parameters *MaxCycle* and *ResTarget* control recalculation of the electric field with the addition of beam space charge. Spatial quantities appearing the script should be scaled by a factor $(39.37)^{-1}$ to convert them to meters. In the *Particles* section we specify two parameters to control the self-consistent solution. There are 20 cycles of orbit tracking and field recalculation with a charge-averaging factor $Avg = 0.20$. Electrons will be generated along an emission surface of facets connecting the nodes of Region 6. The parameters in the *Emission* section have the following meanings: 1) $Mass = 0.0$ (insert the electron mass in AMU), 2) $Charge = -1.0$ (use the electron charge) and $DEmit = 0.07$ (position the Child-law calculation surface about 2.8 element widths from the physical cathode surface). Section 10.3 describes emission surface parameters in detail. Finally, entries in the *Diagnostics* specify that the code should create a file `KLYSTRONGUNP.PRT` containing particle parameters at the system exit and a file `KLYSTRONGUNP.EOU` that describes the modified electric field. Click *Write script* and save the file in the working directory with the name `KLYSTRONGUN.TIN`.

2.5 Running Trak and plotting orbits

Pick the command *Solve* in the **Trak** main menu and choose the input file `KLYSTRONGUN.TIN`. The blue color of the screen indicates that the program is in the solution mode. The status bar at the bottom reports progress. For each of 20 main cycles, **Trak** computes orbits of 216 model particles, corrects the space charge and then performs an iterative matrix solution to find the modified electrostatic potential. On the final cycle, the program creates the `PRT` and `EOU` files

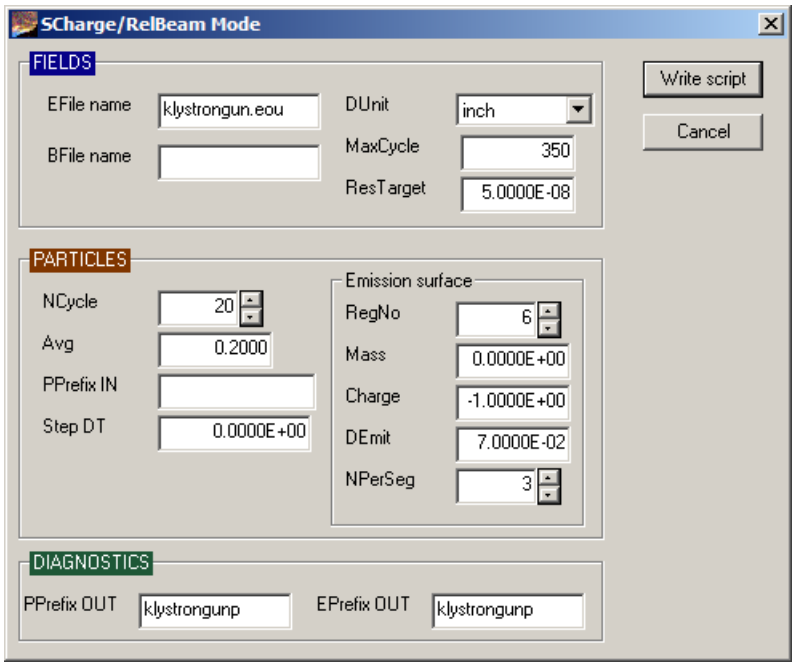

Figure 5: Setting parameters for the **Trak** solution – `KLYSTRONGUN` example.

requested as well as a record of orbit vectors in the plot file `KLYSTRONGUN.TOU`.

Pick the *Plot* command to enter the plot menu and click on *File/Load electric field*. In the dialog choose `KLYSTRONGUN.EOU`. The program reads the file and creates a default plot of equipotential contours. In comparison to the applied-field solution, note that the electric field amplitude over the cathode surface has been reduced to approximately zero by the Child-law emission process. Next, pick *Load/Orbits* and choose `KLYSTRONGUN.TOU`. The program superimposes the orbits on the boundary and field information. At this point, you can experiment with options in the plot menu. Section 5.2 gives a detailed description of available functions. The plot of Fig. 1 was created by choosing the *Element* field plot type and $|\mathbf{E}|$ as the plotted quantity. To differentiate orbits, the parameter *NSkip* in the *Orbit filters* dialog was set to 3.

The **Trak** listing file contains a wealth of information for understanding a run or debugging problems. Return to the main menu, pick the command *File/Edit listing (TLS)* and choose `KLYSTRONGUN.TLS`. Move to the end of the file. Following an extended table of final particle parameters, there is a listing of total emitted current as a function of cycle number (Table 2). This information is useful to check if the choices of the number of cycles and the charge-averaging parameter were correct to ensure solution convergence. With proper averaging, **Trak** converges to a solution where the particle orbits and consequent space-charge density are consistent with the total electric field.

2.6 Modifying the script for advanced dialognostics

Trak has an extensive set of special features. It would be unwieldy, if not impossible, to include all of them in the script generation dialogs (Fig. 5). In this section, we shall add commands to the script we have prepared to invoke two special functions: 1) calculation of beam current density as a function of radius at several axial positions and 2) generation of a file contain

Table 2: Current convergence history – KLYSTRONGUN example.

NCycle	Total emitted current (A)
1	5.1091E+02
2	5.1802E+02
3	7.3695E+02
4	8.6296E+02
5	8.3152E+02
6	5.9833E+02
...	
14	4.7854E+02
15	4.7855E+02
16	4.7860E+02
17	4.7867E+02
18	4.7873E+02
19	4.7876E+02
20	4.7887E+02

information on the spatial distribution of beam-generated magnetic field.

In the main **Trak** menu, pick the command *File/Edit script (TIN)* and choose KLYSTRONGUN.TIN. The file is loaded in a full-featured Windows editor. Modify the Diagnostics section of the script so that it looks like Table 3. The *CDens* commands instruct the program to calculate the beam current density through planes normal to the z axis at several distances from the cathode. The *BBDump* command saves information on the toroidal magnetic field.

In response to the *CDens* command, **Trak** uses information stored on the beam-magnetic-field mesh to add tables to the listing file. Figure 6 shows a plot of the radial current distribution of the converging beam. Note how nonlinear focusing fields on the beam envelope have produced a local enhancement of current density. The file KLYSTRONGUNP.BBD can be loaded into **Trak** with the *Load beam magnetic field* command of the plot menu. Boundaries and values of $|\mathbf{B}_\theta|$ can be included in orbit plots.

Table 3: Modified diagnostic section - KLYSTRONGUN example.

```
DIAGNOSTICS
  PARTFILE: klystrongunp
  EDUMP: klystrongunp
  PARTLIST
* Add these lines
  CDens 1.0 1.2 30
  CDens 2.0 0.8 20
  CDens 3.0 0.6 16
  CDens 4.0 0.4 12
  CDens 5.0 0.3 10
  BBDump KLYSTRONGUNP
END
```

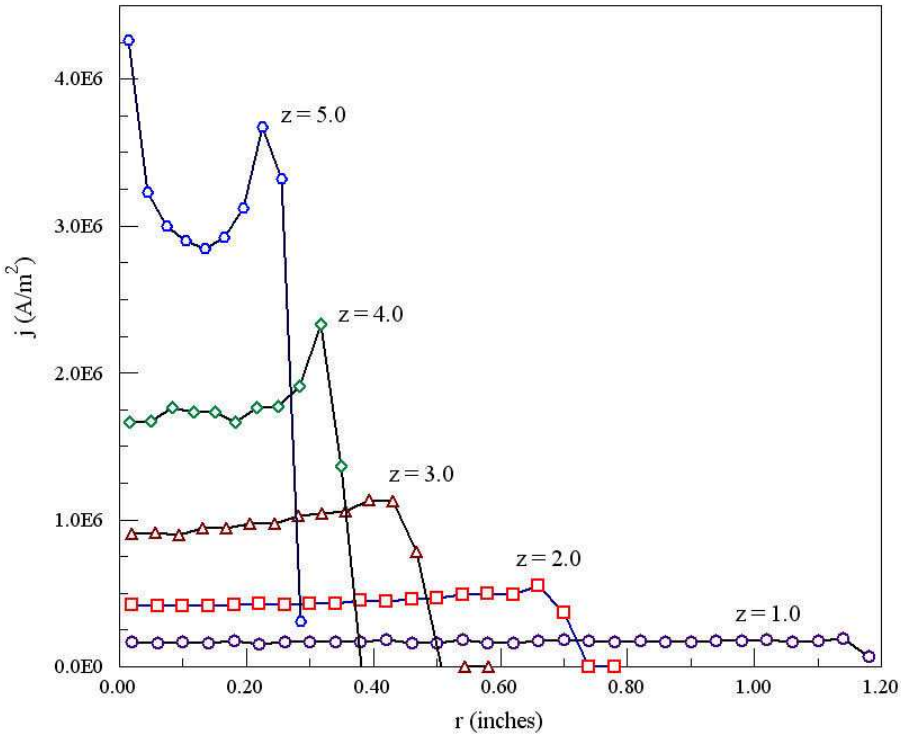

Figure 6: Radial distribution of beam current density at several distances from the cathode – KLYSTRONGUN example.

3 Interactive run setup

3.1 Structure of the Trak script

A control script `FPREFIX.TIN` is required for each **Trak** run. This chapter discusses how to create scripts using the interactive dialogs that appear in response to the *SetUp* command in the main **Trak** menu. Following chapters give detailed information on building scripts and adding features with a text editor.

Whichever method is used, it is useful to understand the script format. A script has three sections: *Fields*, *Particles* and *Diagnostics*. The file has the following structure:

```
Fields
   (Field commands)
End

Particles RunMode
   (Particle commands)
End

Diagnostics
   (Diagnostic commands)
End

EndFile
```

The sections must appear in the order shown and must terminate with an *End* command. Each section has a set of allowed commands. Within a section, valid commands may appear in any order. **Trak** begins processing a section when all commands have been read. The *EndFile* command closes all files and stops the program.

The commands in the *Fields* section control input of field solution files from **EStat** and/or **PerMag**. Advanced commands are available to adjust field values, introduce time variations, add constant magnetic field components and adjust the potential of individual electrodes.

The *Particles* section controls orbit computations. A string parameter in the section heading specifies the tracking mode (Sect. 1.5): *Field*, *Track*, *SCharge*, *RelBeam*, *FEmit* or *Plasma*. The valid commands of the *Particles* section depend on the tracking mode - the program stops with an error message if it finds an invalid command. The commands of the *Particles* section serve three functions:

- Controlling orbit calculations (global boundaries, mesh search options, time step, listing options, ...)
- Initiating particle orbits (particle parameters, marked emission surfaces, ...)
- Defining stopping criteria

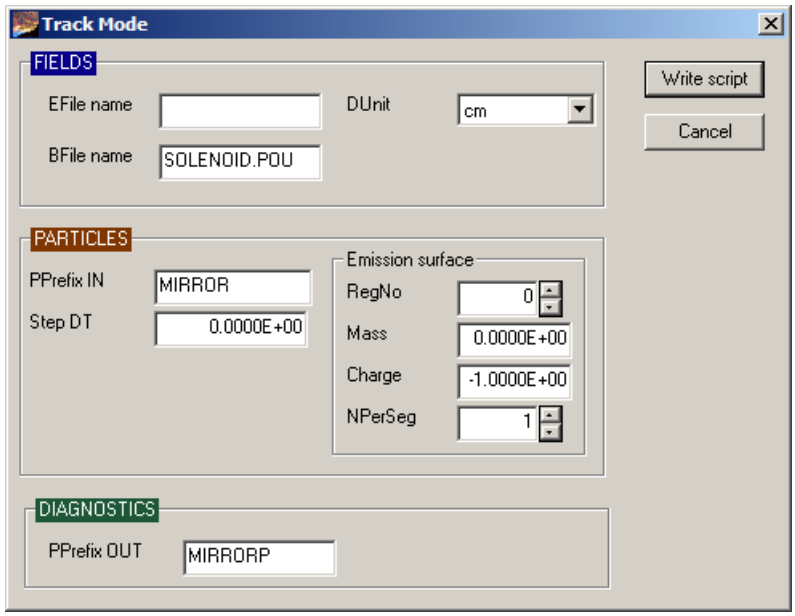

Figure 7: Run setup dialog – *Track* mode.

Trak reads all commands in the *Particles* section, processes the information, and then computes the orbits. Depending on the tracking mode, the program may also update the potential and recalculate orbits over a number of cycles.

The commands of the *Diagnostics* section control special output listings of information on fields and calculated orbits. **Trak** can record scans of applied magnetic fields, self-consistent electric fields and beam-generated magnetic fields. The program makes formatted listings and output files of initial and final particle parameters. The program can perform automatic analyses of final beam distributions. You can also use **GenDist** for distribution calculations if the **Trak** run generates an output particle file.

3.2 Track mode

Trak features interactive dialogs to help you create basic run control scripts. To start, click the *SetUp* menu command or tool. In the initial dialog, pick the appropriate tracking mode. If you choose the default *Track* mode, the program calls up the dialog of Fig. 7. Note that dialog entries are divided into the same three categories as the **Trak** script: *Fields*, *Particles* and *Diagnostics*. Entries in the different dialog groups create commands in the corresponding script section. This chapter gives a brief description of the actions of the dialog entries. Detailed descriptions and commands to control advanced functions are described in Chaps. 6 through 15.

EFILEPREFIX

The prefix of an **EStat** solution to provide electric field information for particle tracking.

BFILEPREFIX

The prefix of a **PerMag** solution to provide magnetic field informations for particle tracking.

DUNIT
A conversion factor for lengths that appear in **Trak** script commands, equal to the number of distance units per meter. For example, to supply dimensions in microns, set $DUnit = 1.0 \times 10^6$.

PPREFIX IN
Supply the prefix of a file `PPREFIX.PRT` that contains a list of start parameters for particles. The format of the file is described in Sect. 8.2.

DT
Supply a time step (in units of seconds) for orbit integrations. If the box is blank or contains 0.0, **Trak** will try to pick an appropriate value based on the properties of the mesh, the electric-field solution and the particle parameters.

The next four commands appear in the *Emission surface* group. An emission surface is an alternate way to initiate particle orbits. **Trak** identifies element facets of a fixed-potential region (electrode) and starts one or more particles per facet close to the surface in an adjacent dielectric element.

REGNO
Supply the number of an emission line region on the surface of a fixed-potential region. Section 8.3 covers techniques to create emission surfaces in **Mesh**.

MASS
Supply the mass of particles created on the emission surface in AMU (atomic mass units). **Trak** inserts the mass of the electron if 0.0 appears in the box.

CHARGE
Supply the charge of particles created on the emission surface in fundamental charge units. Here, protons have $Charge = +1.0$ and electrons have $Charge = -1.0$.

NPERSEG
Enter the number of orbits to start per emission surface facet.

PPREFIX OUT
Supply a name if you want **Trak** to write a file `PPREFIX.PRT` of final orbit parameters. The file may be used as input in a subsequent **Trak** run or ported to the Field Precision programs **GenDist**, **OmniTrak** and **GamBet**.

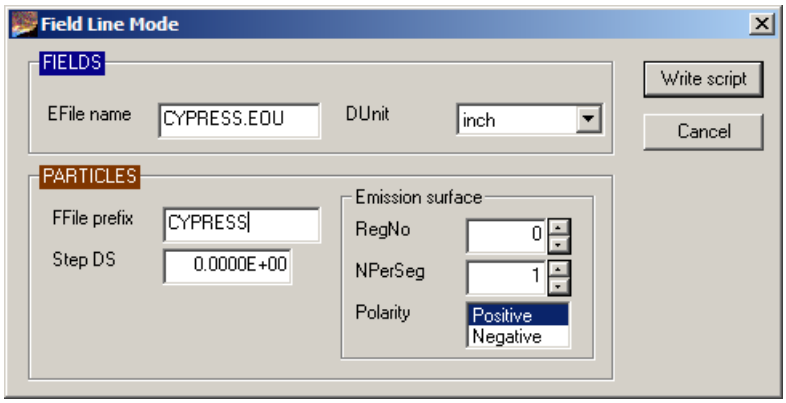

Figure 8: Run setup dialog – *FLine* mode.

3.3 Field line mode

If you choose the *FLine* mode, **Trak** displays the dialog of Fig. 8. Field line tracking applies only to electric fields because magnetic field-line plots can easily be created in **PerMag**. Hence, there is no *BFileName* entry in the *Fields* section.

FFILE PREFIX

Supply the prefix of a file `PREFIX.FLD` that contains a list of start parameters for field lines. The format of the file is described in Sect. 9.1.

DS

Field line integrals proceed in small spatial steps rather than time steps. Enter a value in units specified by *DUnit*. **Trak** will pick a default if the field is blank or equals 0.0.

Entries in the *Emission surface* group are similar to those of the *Track* mode dialog. The main difference is the absence of particle parameters and the presence of the following entry.

POLARITY

For *Positive* polarity, the spatial integral proceeds along the direction of positive electric field.

3.4 Space-charge and relativistic beam modes

The dialog of Fig. 9 is displayed when you pick either the *SCharge* or *RelBeam* modes. In comparison to the dialog of the *Track* mode, there are two additional entries in the *Fields* group.

MAXCYCLE

Particle orbits are computed and the electric field is recalculated over *NCycle* field/particle cycles to include the effect of beam space charge. The parameter *MaxCycle* is the maximum number of iterations in the matrix solution for the electrostatic potential. Higher values give more accuracy at the expense of longer run times. The value of *MaxCycle* should be high

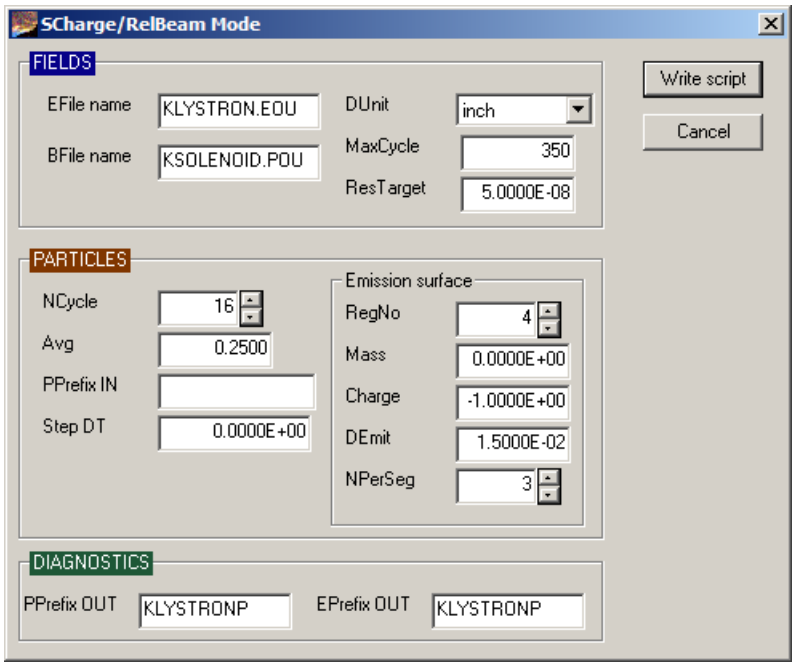

Figure 9: Run setup dialog – *SCharge* and *RelBeam* modes.

enough to ensure convergence at the end of the run. A solution is convergent if the following conditions are satisfied:

- The current from emission surfaces changes little between field/particle cycles.
- The initial relative residual (average error in the electrostatic potential) has a low value ($\ll 1.0 \times 10^{-6}$) entering the final particle/field cycle.

RESTARGET

Set a value for the target *relative residual* in the matrix solution for the electrostatic potential. The quantity is a measure of the accuracy of the solution and should be small compared to unity. **Trak** exits the field recalculation routine it the number of interactions exceeds $MaxCycle$ or if the residual is less than the target value. Choices of $MaxCycle$ and $ResTarget$ affect the solution accuracy versus the run time.

The *Emission* surface group in the *Particles* section has the following additional parameter.

DEMIT

The quantity $DEmit$ is the distance between the physical source surface (*e.g.*, cathode surface) and a virtual emission surface required to model Child-law emission. Enter $DEmit$ in units set by the current value of $DUnit$. The virtual surface be close to the source source and follow its general contours. On the other hand, $DEmit$ must be ≥ 1.5 times the local element width to ensure accurate calculations of the electric field.

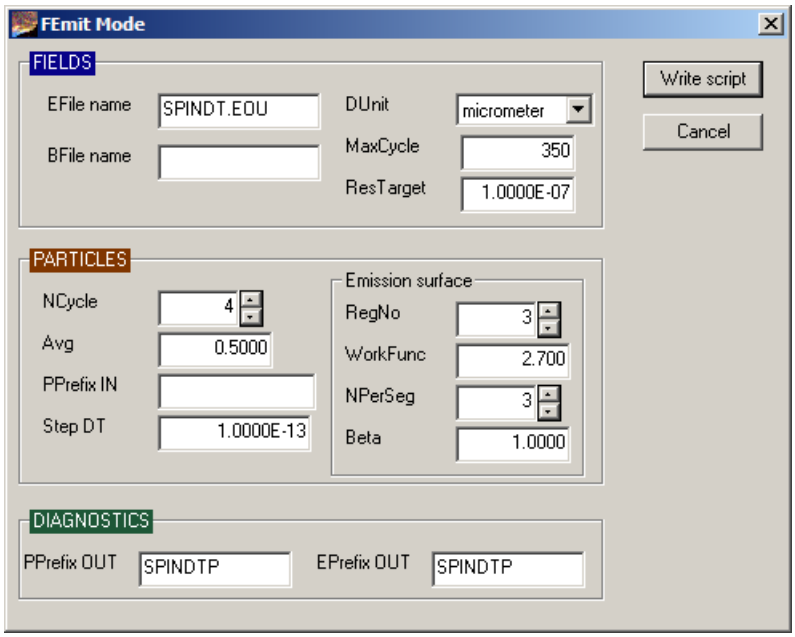

Figure 10: Run setup dialog – *FEmit* mode.

EPREFIX OUT

Record the final electrostatic field solution with the effects of space-charge in a file with the name `EPREFIX.EOU`. The file may be inspected with **EStat** or loaded into **Trak** for a subsequent solution.

3.5 Field-emission mode

Figure 10 show the dialog to generate a script for the *FEmit* mode. The entries are identical to those of the *SCharge/RelBeam* dialog with the exception of those in the *Emission* surface group. Because the mode handles only electron field emission, the options *Mass* and *Charge* are not included.

WORKFUNC

Enter the emission surface work function in units of eV.

BETA

The quantity β is a parameter to represent field enhancement by submicroscopic structure of the emission surface (e.g., a patch covered with carbon nanotubes). **Trak** calculates the local field amplitude $|\mathbf{E}|$ and uses the quantity $\beta|\mathbf{E}|$ in the Fowler-Nordheim equation to find the local electron current density.

3.6 Plasma mode

The optimizing calculation in the plasma mode is complex and may require adjustment of parameters to achieve good results. It is essential to read Chap. 13 to understand the actions

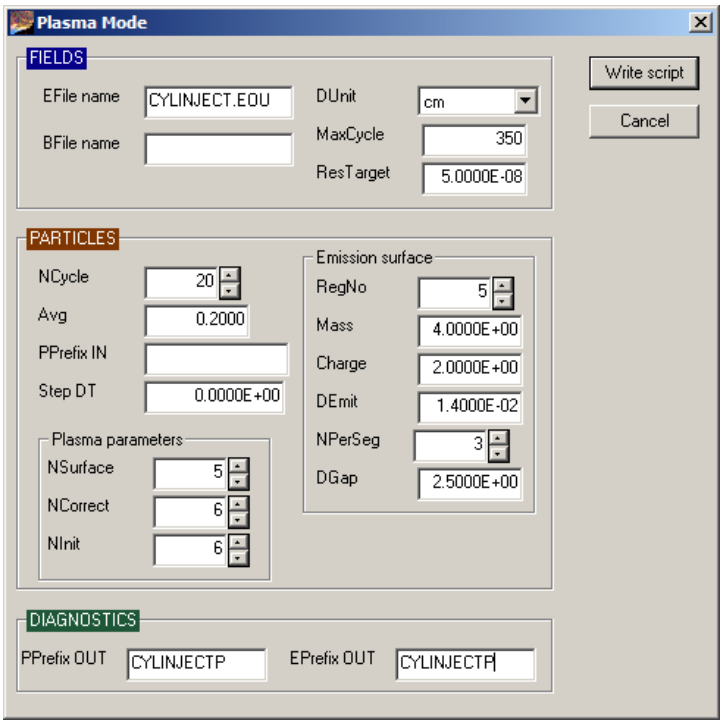

Figure 11: Run setup dialog – *Plasma* mode.

of entries. Many entries have identical function to those in the *SCharge* and *RelBeam* modes. This section gives a brief summary of quantities unique to the *Plasma* mode. The *Plasma* group contains the following entries:

NSURFACE
The number of cycles of surface adjustment. The default is $NSurface = 3$.

NCORRECT
The number of orbit-field recalculations to determine a stable solution for space-charge-limited flow per surface adjustment (default, $NCorrect = 5$).

NINIT
The number of initial orbit-field calculations before the first surface adjustment (default, $NInit = 12$).

The *Emission* surface group contain one additional parameter:

DGAP
The approximate total width of the acceleration gap in units set by the current value of $DUnit$.

4 Running the Trak program

4.1 Interactive mode commands

Trak runs interactively in a window if you launch `trak.exe` from **TC** or run the program without a command-line parameter. In this mode, you can carry out a variety of activities and perform several solutions in a session. In the interactive mode, the program serves three main functions, corresponding to the tools "1", "2" and "3":

1. **Setup**, preparation of control scripts.
2. **Run**, generation of orbit solutions.
3. **Plot**, creation of orbit/field plots.

Note that **Trak** can perform only one function at a time. If you want to carry out extended orbit solutions while you are writing scripts or making plots, then launch a second instance of `trak.exe`. There is no speed penalty if you have a dual-processor machine.

The main menu has five entries: *File*, *Setup*, *Solve*, *Plot*, *Tools* and *Help*. We discussed *Setup* options in Chap.3, while the next chapter covers plotting capabilities. The following commands appear in the *File*, *Solve* and *Help* popup menus:

EDIT SCRIPT (TIN)
EDIT LISTING (TLS)
EDIT FILE

The commands invoke the internal editor to inspect or to modify text input and output files. With the *Edit script* command you can work on files with names of the form `FPREFIX.TIN`, while the *Edit listing* command displays file of type `FPREFIX.TLS`. Choosing a file from an alternate directory does not change the working directory of the program.

RUN

Pick an input file with a name of the form `FPREFIX.TIN` to start a solution. The working directory changes if you pick a file from an alternate directory. The run begins if all required files are available in the working directory. The screen color is blue during extended calculations and the program reports run progress in the status bar.

STOP

This command terminates the run immediately. The program records information currently available and closes all files.

TRAK MANUAL

Displays this document in your default PDF viewer. Note that `trak.pdf` must be in the same directory as `trak.exe`.

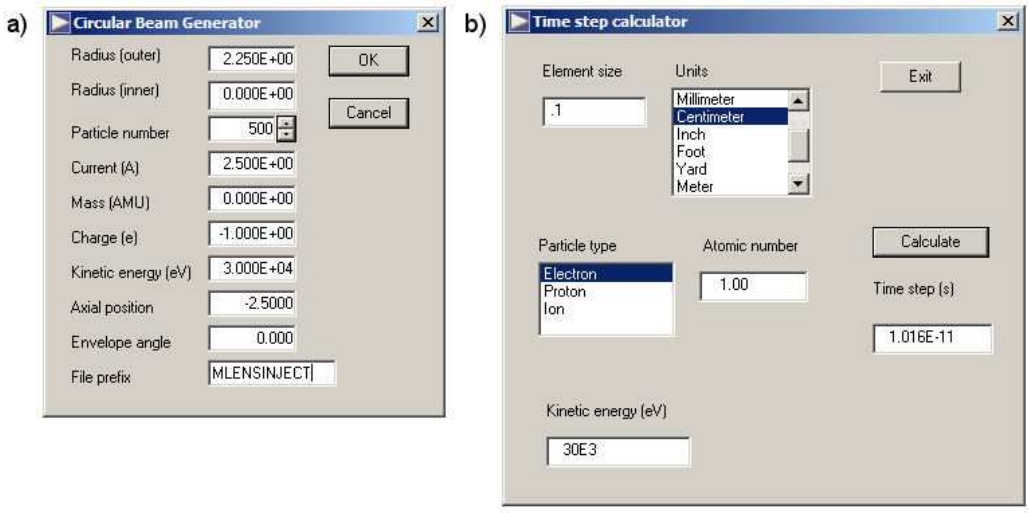

Figure 12: Utilities in the *Tool* menu. *a*) Circular-beam generator. *b*) Time-step calculator.

4.2 Tool menu

This menu contains helpful utilities for run preparation.

CIRCBEAM GENERATOR

This command brings up the dialog of Fig. 12*a* to create a `PRT` file describing a laminar, circular beam with uniform current density. The routine generates a set of $NPart$ orbits uniformly spaced in r. The assigned current is proportional to r to give a uniform current density. The average beam motion is directed along z. A non-zero value for the envelope angle gives a diverging or converging beam with additional momentum components in the x direction. If the $Mass$ equals zero, the program inserts the mass of the electron.

TIME-STEP CALCULATOR

In response to this command, **Trak** displays the dialog of Fig.12*b* for estimating values to use in the *Dt* command. Enter a value for the minimum element size and choose the appropriate length unit. Enter the particle parameters including maximum kinetic energy. The *Atomic number* value is used only for the particle type *Ions*. When you click the *Calculate* button, the program displays an appropriate value for *Dt* in seconds.

4.3 Command-line operation

You can invoke the orbit solution function of **Trak** directly from the command prompt. This feature is useful for batch file operation where a sequence of calculations runs automatically in the background. To make a single **Trak** simulation in the background, go to the Command Prompt in Windows and log to the data directory that contains the required input files. For example, suppose the input files `SWITCH.TOU` and `SWITCH.TIN` are stored in `\TRICOMP\BUFFER` and that the program trak.exe is in the directory `\TRICOMP`. From `\TRICOMP\BUFFER` type

```
..\Trak SWITCH <Enter>
```

The program runs silently, writing detailed information in the listing file SWITCH.TLS and the plot file SWITCH.TOU.

The command mode is useful for autonomous operation under batch file control. As an example, suppose you have prepared the input files KINJECTOR.MIN, KINJECTOR.EIN and KINJECTOR.TIN. The following batch file will initiate a complete calculation:

```
ECHO OFF
ECHO Running KINJECTOR example
START ..\MESH.EXE KINJECTOR
START ..\ESTAT.EXE KINJECTOR
START ..\TRAK.EXE KINJECTOR
ECHO KINJECTOR runs complete
```

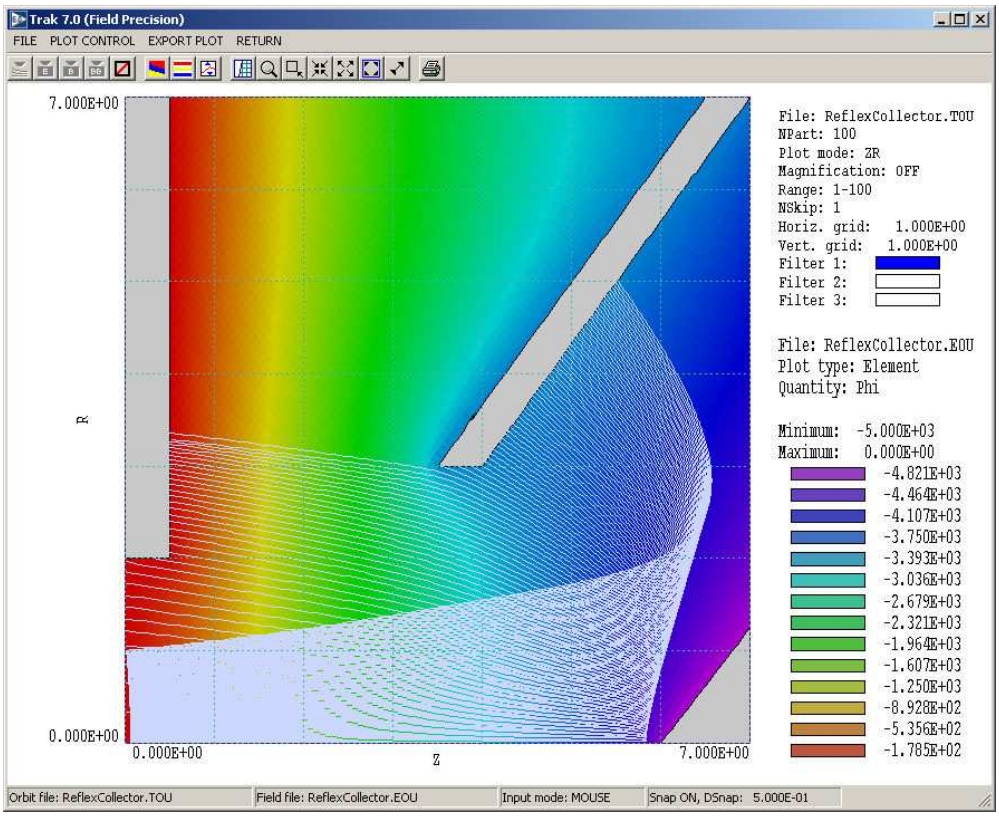

Figure 13: Working environment for creating orbit/field plots.

5 Creating orbit plots

In response to the *Plot* command in the main menu, **Trak** enters a mode to create displays of orbits and field quantities. The program loads the command menu, toolbar and status bar shown in Fig. 13. There are three popup menus: *File*, *Plot control* and *Export plot*. The *Return* command restores the main menu. The display area is divided into three sections: 1) main plot, 2) particle information area (upper right) and 3) field information area (lower right).

5.1 File menu commands

LOAD ORBITS

To begin a plotting session, you must load data from either an orbit or field file. Use this command to load orbits. The dialog shows a list of files with names of the form `FNAME.TOU`. Changing directories in the dialog changes the working directory. In the absence of a field file, **Trak** picks boundaries for the initial plot based on extreme values of the orbit vectors.

LOAD ELECTRIC FIELD

Load a file of boundary and electric field information with a name of the form `FNAME.EOU`. The

file could be output from **EStat** or one created by **Trak** using the *EDump* command. There are three points to note:

- When a field plot is loaded, the boundaries of the solution region are used for the default plot boundaries.
- The program cannot check whether the field corresponds to the orbits – you must ensure that the data files match.
- If you have applied a spatial shift to the electric field in the **Trak** run, the orbits will be inconsistent if load the input applied field. You must create a file of the modified electric field using the *EDump* command and load it for a valid superimposed plot.

LOAD MAGNETIC FIELD

Load information from a file with a name of the form `FNAME.POU` generated by **PerMag**.

LOAD BEAM MAGNETIC FIELD

Load a file with a name of the form `FNAME.BBD` generated by **Trak** in a *RelBeam* calculation in response to the *BBDump* command.

CLOSE ORBIT FILE

Close the current orbit file in preparation for loading another one, keeping the current field plot active.

CLOSE FIELD FILE

Close the current field file in preparation for loading another one, keeping the current orbit plot active.

CLOSE ORBIT AND FIELD FILES

Close all currently-loaded orbit and/or field files in preparation for loading new data.

ORBIT FILE INFORMATION Display information about the current orbit file (Fig. 14). The data on the spatial limits and the range of particle parameters (mass, charge and current) may be useful for setting plot filters. Note that information on the mass and charge of individual particles is present in all `TOU` files created by Version 7.0, but is not included in data files created by earlier versions of **Trak**.

FIELD FILE INFORMATION Show information on the current electric, magnetic or beam-generated magnetic field file.

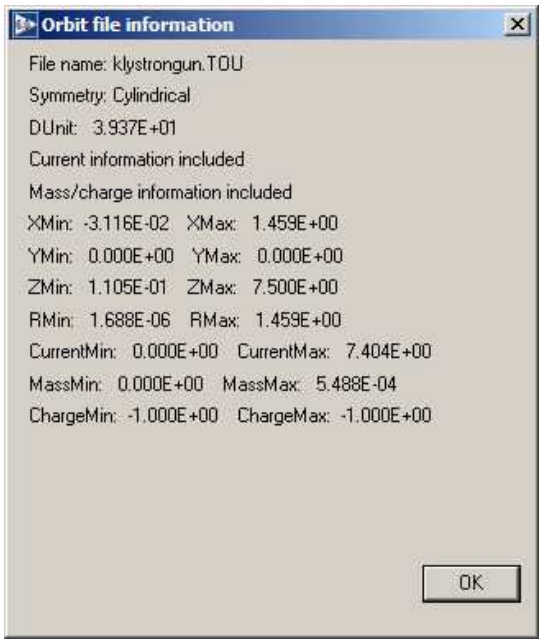

Figure 14: Information on the currently-loaded orbit file displayed in a message box.

5.2 Plot control menu commands

ORBIT PLOT TYPE

Using the (x, y, z) coordinates recorded in the plot file, **Trak** can create the following two-dimensional plots: y versus x, x versus z, y versus z and r versus z. Depending on the field symmetry, different plot types may or may not contain useful information. **Trak** suppresses the field plot if it is inconsistent with the current orbit plot type. The default types are x versus y for planar calculations and r versus z for cylindrical.

ORBIT FILTERS

This command calls up the dialog of Fig. 15 to set filters for plotting particle orbits. You can use filters to set a plot color or to suppress plotting for classes of particles. For example, if the **Trak** calculation involved a counterflow of electrons and ions, you could plot electrons in red and ions in blue by setting filters based on particle mass. As shown in Fig. 15, there are three filters with associated screen colors blue, red and green. When a new orbit file is loaded, the default settings are that Filter 1 is active and encompasses all particles, while Filter 2 and 3 are inactive. The filter status is displayed in the information window. Within a filter, the active status of the selection criteria depend on the information recorded in the `TOU` file. To activate a filter, check the *Active* box and supply information in the active criterion boxes. The input-quantity units are amperes or amperes/m for current, AMU for mass and the fundamental electron charge (e $= 1.60210 \times 10^{-19}$). The boxes on the right-hand side of the dialog define additional selection criteria. The parameter $NSkip$ can be adjusted to reduce the density of orbits in a plot for clarity. For example, when $NSkip = 5$ **Trak** plots orbits $N = 1, 6, 11, ...$ The number N gives the order in which the orbit appears in the `TOU` file and output `PRT` files. Use the parameters $NPlotMin$ and $NPlotMax$ to limit the range of N.

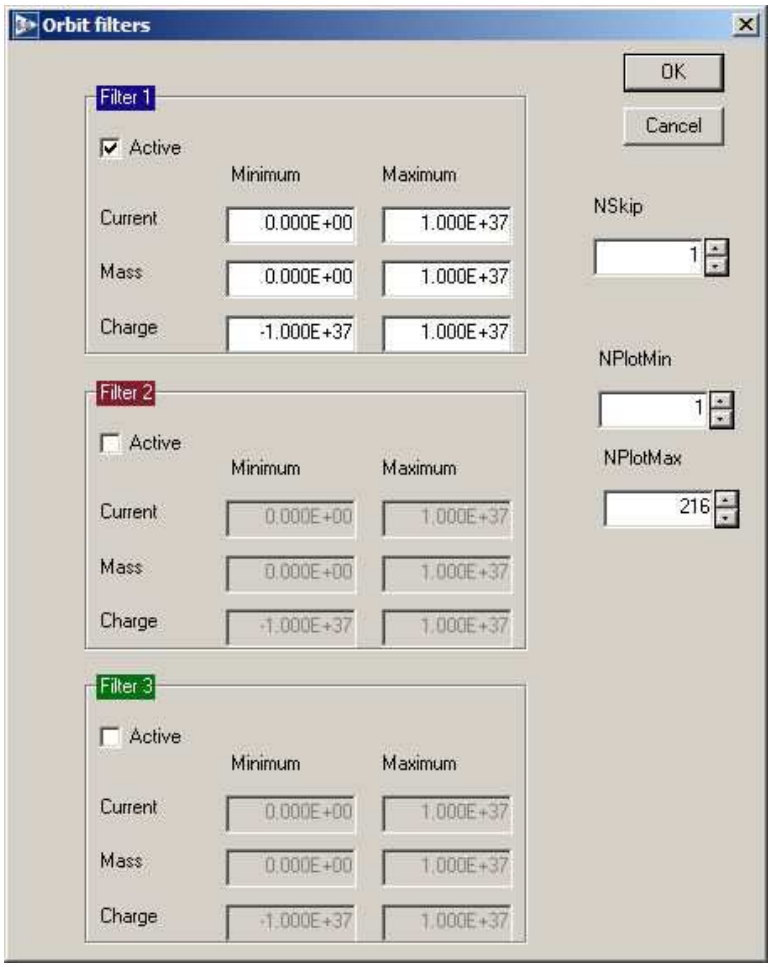

Figure 15: Dialog to set filter criteria for the display of particle orbits.

FIELD PLOT TYPE

The program supports the following plot types:

- **Mesh**. Element facets of the computational mesh.
- **Region**. Computational mesh with elements color-coded by region number.
- **Contour**. Lines that follow constant values of a computed quantity: electrostatic potential, vector potential or stream function or B_θ. When a magnetic field has been loaded, the contours show lines of magnetic flux density **B**.
- **Element**. Elements of the solution space color-coded according to a computed quantity (such as the electric field magnitude)

FIELD QUANTITY

This command affects only element plots. Available quantities are ϕ or $|\mathbf{E}|$ for electric field files, field lines or $|\mathbf{B}|$ for magnetic field files and and B_θ or enclosed current for beam-generated magnetic field files.

PLOT LIMITS

Use this command to set limits for the field quantity or to restore automatically scaling in field plots.

NUMBER OF CONTOURS

Change the number of lines in contour plots.

GRID CONTROL

Open a dialog to control the display of grid lines. Here, you can activate or suppress grid display. Alternatively, you can use the *Toggle grid* tool. You can also set specific horizontal or vertical intervals or restore automatic intervals. In the latter case, **Trak** chooses grid spacing to correspond to easily recognized numbers (i.e., 0.01, 0.02, 0.05, ...). The intervals are displayed in the information area the grid is active.

ELEMENT OUTLINES

Use this command if you want to add the boundaries of elements to mesh, region and element plots. The feature may be useful to check whether the grid resolution is adequate in a **Trak** simulation. If you deactivate element outlines in a mesh-type plot, the program includes only outlines of region boundaries.

XY MAGNIFICATIONS

By default, **Trak** generates plots at approximately true scale. (The exact scaling depends on the properties of your monitor and the proportions of the program window.) The program has an alternate plot mode that is useful for charged-particle beam simulations where the solution size

in the transverse direction may be much smaller than the axial size. In the *XY magnification* mode, you can manually specify the coordinates of the plot boundaries. **Trak** adjusts scales so that the plot fills the full plot area. Note that the zoom and pan commands are inactive in the XY magnification mode.

RESET PLOTS ON LOAD

In the default mode, **Trak** automatically resets plot parameters (current view, plot type, magnification mode,..) when you load new data with the commands of Sect. 5.1. Sometimes, you may want to compare a series of similar runs maintaining a specific view or magnification limits. In this case, use this command to deactivate autoscaling. The current setting (*Autoset plot parameters* or *Fixed plot parameters*) is shown on the right-hand side of the status bar. Note that **Trak** automatically rescales the plot when you switch from *Fixed* to *Autoset* mode.

SAVE CURRENT PLOT

This command (which appears in the *Export plot* menu) is useful if you want to superimpose electric field, magnetic field and orbit information in the same plot or if you want to make a plot that shows small differences between orbit solutions. When you click on the command, **Trak** creates a bitmap image of the current plot. In the dialog, supply a file prefix. The plot will be stored in standard Portable Network Graphics format with the name `FPREFIX.PNG`. You can then superimpose this image on subsequent plots.

COMBINE PLOTS

Combine the current plot with a stored bitmap image (or any `PNG` image) using the bitwise `AND` operation. The dialog displays a list of files with suffix `PNG`.

UNDO COMBINATION

Reverse the image combination and restore the plot.

As an example, you might load a large volume magnetic field solution, set the plot limits to correspond to an electric field solution and then save the bitmap. You could then load orbit trajectories and the electric field solution to create a new plot. Finally, use the *Combine plots* command to include magnetic field lines. Figure 16 shows an example. Here are some rules for using bitmap superposition:

- For an accurate superposition, you must ensure that the current plot and the bitmap plot have the same spatial limits.
- The superposition looks best if one of the plots contains discrete line data (*e.g.*, contour lines or particle orbits). Combining two element plots would give unpredictable results.
- For the best appearance, do not resize the program window between recording and combining operations. If the window has been resized, **Trak** will make its best effort to stretch the bitmap to fit.

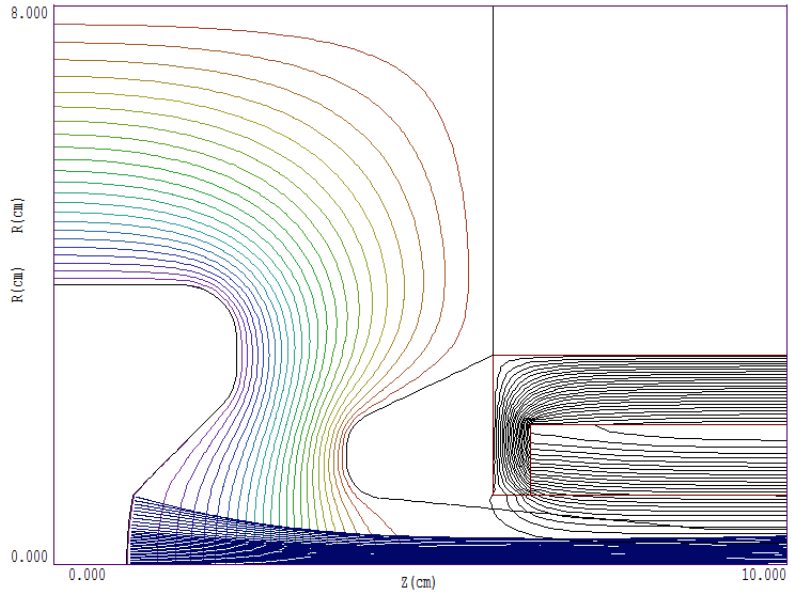

Figure 16: Plot combining particle orbits, electrostatic equipotential lines, magnetic field lines and region boundaries in both field solutions.

- The *Plot file* commands in the *Export* menu use vector operations to reconstruct the current plot and will not include the combined information. To save the combined plot, use the *Save current plot* command.

CONTOUR PLOT STYLE

Choose the method for displaying electrostatic contours or magnetic field lines. There are four choices: monochrome, monochrome with labels, colored and colored with labels. In the colored mode, the lines are color-coded according to the value of the plotted quantity. A legend is included in the information window to the right of the plot. In the labeled modes, contour lines are numbered according to their values. Overlapping labels on closely-spaced lines may look better in a zoomed view.

SETTINGS/MOUSE/KEYBOARD
SETTINGS/SNAP MODE
SNAP DISTANCE
ZOOM WINDOW
ZOOM IN
EXPAND VIEW
GLOBAL VIEW
PAN VIEW
DEFAULT PRINTER
SAVE PLOT FILE
COPY TO CLIPBOARD

These commands adjust the plot view and send plots to printers and graphics files. Their functions are described in the **EStat** and **PerMag** manuals.

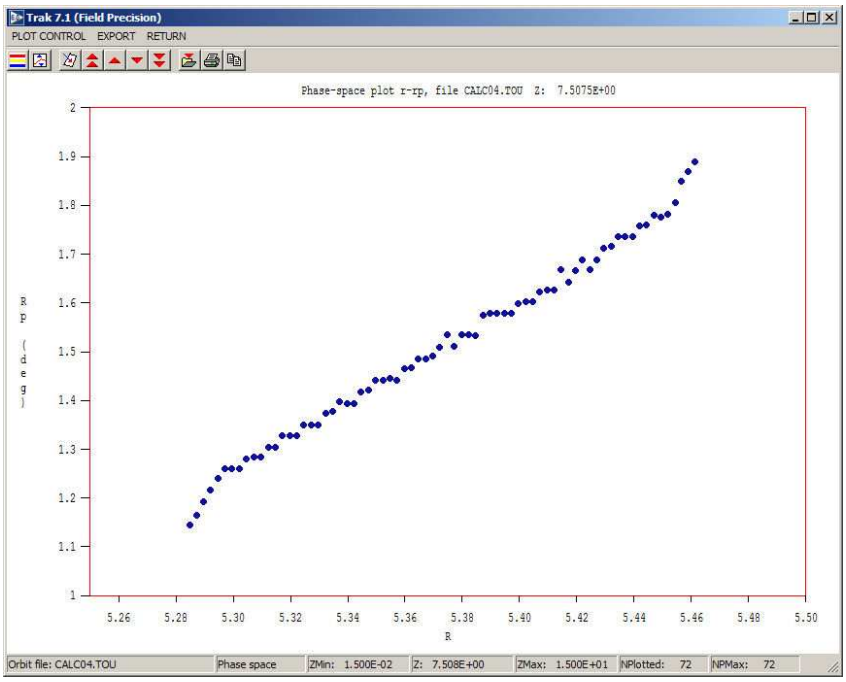

Figure 17: Distribution plot menu showing a phase space plot for an annular beam.

5.3 Distribution plot menu

In the *Distribution* menu, you can investigate the evolution of beam distributions as a function of position along the x (planar) or z (cylindrical) axis. There are two requirements to plot distributions:

- An orbit file (`FNAME.TOU`) must be loaded.
- Particle motion should be *paraxial* about x or z.

The second item implies that the particles constitute a recognizable beam (*i.e.*, orbits make small angles with respect to the axis).

When you enter the *Distribution* menu, **Trak** creates a default phase-space plot at a position along the axis approximately midway between the orbit starting and ending positions (Fig. 17). The plot shows the transverse displacements of particles from the axis (y or r) and the corresponding angle with respect the axis (y' or r'). The displacements are in the units set by the value of *DUnit* in the `TOU` file and the angles are in degrees. You can move in x or z with the following commands:

- *Jump forward* (double up-arrow tool): Move a large step in the $+x$ or $+z$ direction.
- *Step forward* (single up-arrow tool): Move a small step in the $+x$ or $+z$ direction.
- *Step backward* (single down-arrow tool): Move a small step in the $-x$ or $-z$ direction.
- *Jump backward* (double down-arrow tool): Move a large step in the $-x$ or $-z$ direction.

Alternatively, you can use the *Set plane* command to bring up a dialog. Here, you can move a slider to set an approximate position or type an exact value in the box.

In the default mode, **Trak** rescales graphs to fit the full plot area as you move. If you want to compare distributions, use the *Plot limits* command to set fixed values for the minimum and maximum values or displacement and/or angle. In the dialog, uncheck the *Autoscale* box and supply limits. The values computed for the plot may be exported to a text file. To activate the feature, click the command *Toggle plot recording* in the *Export plot* menu. Supply a prefix for a file with a name of the form `FPREFIX.DPL` (**D**istribution **PL**ot). The values for subsequent graphs are recorded in the file. Click the command again to close the file and end recording. The following is a portion of the listing for the graph of Fig. 17:

```
Phase-space distribution, Z:   7.50750E+00
        r         rp(deg)
=========================
  5.28484E+00  1.14543E+00
  5.28728E+00  1.16563E+00
  5.28968E+00  1.19258E+00
```

Click on the *Plot quantity* command or tool to switch to a plot of current density as a function of y or r. Depending on the number of available particles, **Trak** picks an optimum number of bins along y or r and assigns current using the cloud-in-cell method. Each particle has a transverse width $2\Delta r$, where Δr is the average spacing between particles. If a particle overlaps a bin, a portion of the particle current is assigned equal to the fraction of overlap. The procedure gives satisfactory results, even with a relatively small number of particles. The limitation is that variations of current over length scales less than Δr cannot be resolved. Use more particles to improve the resolution. The Plot limits command is not active for current-density plots. To make comparisons, you can transfer information in the recording file to your own plot program. An example is shown below. The recorded quantities are the minimum, maximum and average transverse positions of the bin and current density in A/cm^2.

```
Current density calculation, Z:   8.10690E+00
      RIn          ROut         RAvg      jz(A/cm2)
=================================================
  5.28979E+00  5.30097E+00  5.29538E+00  3.63497E+00
  5.30097E+00  5.31214E+00  5.30655E+00  4.25687E+00
  5.31214E+00  5.32332E+00  5.31773E+00  4.26024E+00
```

6 Electric field input and control

6.1 General commands

The first step processing a **Trak** script is to define electric and/or magnetic fields by reading solution files from **EStat** and/or **PerMag**. Commands that appear in the *Fields* section control this function. This chapter covers commands to load and to modify electric field solutions from **EStat**. Commands are displayed with symbolic parameters and also in the form they would appear in the script file.

DUNIT DUnit
DUNIT = 39.37
This command controls the interpretation of spatial coordinate input from script commands. The quantity *DUnit* (real) is the number of distance units per meter. For example, to enter positions in cm, set $DUnit = 100.0$. The command can appear anywhere in the script and affects all following commands. Note that the values of DUnit defined in the **EStat** or **PerMag** input scripts set coordinates for the input meshes but have no effect on position quantities entered in the Trak script. The quantity *DUnit* is recorded in the `TOU` plot file and used to scale plot dimensions.

BOUNDARY X1 Y1 X2 Y2
BOUNDARY Z1 R1 Z2 R2
BOUNDARY = (0.0, 0.0, 20.0, 5.0)
Orbit calculations are performed inside a two-dimensional solution box with opposite corners defined by (x_1, y_1)-(x_2, y_2) for planar solutions or (z_1, r_1)-(z_2, r_2) for cylindrical geometry. Enter coordinates in units set by *DUnit*. Particle orbits terminate if they move outside the box. **Trak** interpolates the final orbit step so that the stopping point lies exactly on the box surface. If the *Boundary* command does not appear, **Trak** sets the solution volume as the largest box that overlaps the electrical and/or magnetic solutions. In the *Track* (single-particle) mode, the program sets the *ballistic flag* if the *Boundary* command is present. In this case, orbits continue even if they leave the boundaries of the electric and/or magnetic solution volume. The program takes $\mathbf{E} = 0.0$ and $\mathbf{B} = 0.0$ in regions that are inside the orbit calculation box but outside the field solution volumes. As an example, the ballistic mode is useful if you want to trace orbits to a focal point at a distance from a lens with localized field. Note that the ballistic mode may not be used in the *SCharge* and *RelBeam* modes because space-charge must be assigned along the full particle trajectory.

6.2 Loading and modifying EStat solutions

EFILE FileName [EMult]
EFILE = KlyGun.EOU
This command loads an electric field solution from **EStat**. The quantity *FileName* is the full name of the file (*i.e,*, `EFTEST.EOU`). The optional real-number parameter *EMult* is a global scaling factor. All values of electrostatic potential at nodes are multiplied by *EMult* when they are loaded into the program. This factor is useful if you want to investigate scaling with applied voltage without regenerating the **EStat** solution. The **EStat** file must be available in the working directory. The z axis in the field solution corresponds to the z axis for particle tracking in Trak. A solution with cylindrical symmetry has field components E_r and E_z. The program determines E_x and E_y for tracking from E_r from the particle position. A planar solution has field components E_x and E_y with $E_z = 0.0$.

Note: In runs with both electric and magnetic fields, the field solutions must have the same symmetry - planar or cylindrical. In cylindrical simulations, both the electric and magnetic solutions must share the same z axis.

SHIFT E ZEShift
SHIFT(E) = -5.67
This command with the string parameter E moves nodes in the electric field mesh along the x direction (planar) or z direction (cylindrical) according to

$$z_{new} = z_{old} + Z_{eshift}. \tag{5}$$

Enter the real-number parameter *ZEShift* in the current units set by the *DUnit* command. As an example, you could use this command to make small changes in the position of the electric solution relative to a magnetic solution to optimize beam matching to a magnetic focusing system. The command is also useful in simulations of particle motion through a periodic focusing system. Here, you can split the particle simulation into several stages with shifted field solutions.

MODFUNC E TABLE FileName [Toff Tmult Foff Fmult]
MODFUNC(E,TABLE) = StepFunc.WAV
You can use the form of the *ModFunc* command to add an arbitrary temporal modulation of electric fields. This capability may be applied only in the *Track* tracking mode. The keyword *Table* designates that a tabular function will be imported from a file. The quantity *FileName* is the full name of the file (available in the current directory). The file consists of data lines that define the time variation of a function.

Temporal tables have the format:

```
  t1   fe(t1)
  t2   fe(t2)
  t3   fe(t3)
    ...
  tn   fe(tn)
ENDFILE
```

Values of t should be in units of seconds and values of $f_e(t)$ may be in any relative units. The number of data lines should be less than or equal to 256. If the optional real-number parameters *Toff*, *Tmult*, *Foff* and *Fmult* appear, values in the table are modified according to

$$t_{code} = T_{mult}\ t_{tab} + T_{off}, \tag{6}$$
$$f_{code} = F_{mult}\ f_{tab} + F_{off}. \tag{7}$$

The modulation file follows the standard rules for **TriComp** tabular functions. You can use any of the standard delimiters to separate quantities on the line including *Space* and *Tab*. Comment lines (beginning with an asterisk) may be included. Note that all particle orbits start from their initial position at $t = 0.0$. Using the elapsed time t of an orbit, the program applies a cubic spline interpolation to find $f_e(t)$. The table values should define a smooth function with continuous first derivatives. All electric field components are then multiplied by $f_e(t)$.

By default, **Trak** interprets the table as a periodic function. If the elapsed time of an orbit exceeds the maximum time on the table, **Trak** returns to the beginning. To illustrate, if the orbit time is $t = 5.0$ ns and if the maximum time value in the table is $t_{max} = 4.0$ ns, then f_e is evaluated at 1.0 ns. If you want to apply a single electric-field pulse with zero fields thereafter, be sure that the final entry in the table is of the form

```
  tmax   0.0
```

where t_{max} is larger than the longest particle transit time.

MODFUNC E > Function
MODFUNC(E) > 10.0 + cos(3.1416*$t/25.6)
Define a modulation function for electric field values from a mathematical expression. The symbol > designates that a function string occupies the remainder of the line. The function may be up to 230 characters in length and follows the format described below. The function defines a variation in time, $f(t)$. The parser uses the Perl standard for the time variable: $\$t$ stands for t.

Trak incorporates a flexible and robust algebraic function interpreter. A function is a string (up to 230 characters) that may include the following entities:

- The time variable `$t`.
- Real and/or integer numbers in any valid format (*e.g.*, `3.1415`, `476`, `1.367E23`, `6.25E-02`, `8.92E+04`,...). Integers are converted to real numbers for evaluation.
- Binary operations: `+` (addition), `-` (subtraction), `*` (multiplication), `/` (division) and $\wedge$ (exponentiation).
- Functions: `abs` (absolute value), `sin` (sine), `cos` (cosine), `tan` (tangent), `ln` (normal logarithm), `log` (base 10 logorithm), `exp` (normal exponent) and `sqt` (square root).
- Up to 20 sets of parentheses to any depth.
- Any number of space delimiters.

The parser conforms to the standard algebraic rules and features comprehensive error checking. Errors may include unbalanced parentheses, unrecognized characters and sequential binary operations. To illustrate a valid example, the expression

```
1 - exp(-1.0*(($t^2)/24))
```

corresponds to

$$1 - \exp\left[-\left(\frac{t^2}{24}\right)\right]. \tag{8}$$

CHANGEPOT RegNo PotNew
CHANGEPOT(6) = 25000.0
This command can be used to change the potential of a individual electrode. The quantity *RegNo* (integer) corresponds to the region number defined in **Mesh**. The region must have the fixed-potential property. The quantity *PotNew* (real) is the new value of potential (in volts). You can include multiple *ChangePot* commands in the file. After all electrode values are set, the program performs a relaxation operation to correct values of potential at intervening variable points.

Note: If you modify the electric field solution with the *EMult* parameter, *Shift* command or *ChangePot* command, be careful when creating plots. If you load the original applied field, the field display will not correspond to fields used to compute the particle orbits. Use the *EDump* command in the *Diagnostics* section to make a record of the modified electric field.

6.3 Controlling electric field recalculation

The following commands control the iterative procedure used to update the electric field solution. Updates are required if the *ChangePot* command appears or if effects of beam space charge must be included (*SCharge*, *RelBeam*, *FEmit* and *Plasma* tracking modes).

MAXCYCLE MaxCy
MAXCYCLE = 2500
The integer quantity *MaxCycle* is the maximum number of iteration cycles in the matrix solution for the field. Larger values usually give higher accuracy. The default values are $MaxCycle = 2500$ for initial adjustments in response to the *ChangePot* command and $MaxCycle = 250$ for field relaxations on each tracking cycle in the *SCharge*, *RelBeam*, *FEmit* and *Plasma* tracking modes

OMEGA Omega
OMEGA = 1.95
The quantity *Omega* is the successive over-relaxation factor for the electric field solution. Assign a value in the range 0.0 to 2.0. Values close to 2.0 usually give faster convergence. Lower the parameter if the solution fails to converge. If the *Omega* command does not appear, **Trak** uses default values determined by the Chebyshev prescription.

RESTARGET = ResTarget
RESTARGET = 1.0E-8
The residual is the relative error in the electric field solution and should approach a small value compared to unity. The electric field relaxation terminates if the residual falls below the quantity *ResTarget* (real) or if the number of relaxation cycles exceeds *MaxCy*. The default value is $ResTarget = 1.0 \times 10^{-7}$.

7 Magnetic field input and control

7.1 Loading PerMag solutions

The commands to load and to modify magnetic field solutions created by **PerMag** are similar to those used for electric field files.

BFILE: FileName [BMult]
BFILE = MLENS.POU (1.4)
This command loads a magnetic field solution created with PerMag. The quantity *FileName* is the full name of the file (*i.e.*,`SOLENOID.POU`). The optional real-number parameter *BMult* is a global field scaling factor. Values of vector potential at all nodes are multiplied by *BMult* when they are loaded into the program. This factor is useful if you want to investigate scaling with applied magnetic field without regenerating the **PerMag** solution. The solution file must be available in the working directory. The z axis in the field solution corresponds to the z axis for particle tracking. A solution with cylindrical symmetry has non-zero field components B_z and B_r, while the components from a planar solution are B_x and B_y.

SHIFT B ZBShift
SHIFT(B) = 0.05
The *Shift* command with the key symbol B moves nodes in the magnetic field mesh along the x direction (planar) or z direction. Enter the real-number parameter *ZBShift* in the units set by the *DUnit* command.

MODFUNC B TABLE FileName
MODFUNC(B,TABLE) = SlowRise.PLS
MODFUNC B > Function
MODFUNC(B) > 0.566 + sin(($t + 5.0E-8)/1.2E-7)
The *ModFunc* command with the key symbol *B* adds time variations to magnetic fields. Modulation functions were described in Sect. 6.2. The capability may be applied only in the *Track* tracking mode.

You should note the following facts about time variations of magnetic fields:

- The modulation function is applied to the total magnetic field calculated from all sources.
- **Trak** simply multiplies static field values by the modulation function and makes no checks that the resulting time-dependent fields approximate a solution to Maxwell's equations. You must ensure that the field values are physically valid.

Table 4: Example of a tabular listing of $B_z(0, z)$

```
* Table from SOL_LENS.BOU
*      z           Bz
* ======================
SYMMETRY: Cylin
     -9.000E+00   4.421E-06
     -8.852E+00   5.926E-05
     -8.703E+00   1.173E-04
     -8.555E+00   1.780E-04
        ...
      9.258E+00   1.014E-04
      9.406E+00   7.838E-05
      9.555E+00   5.670E-05
      9.703E+00   3.685E-05
      9.852E+00   1.767E-05
      1.000E+01   7.111E-11
ENDFILE
```

7.2 Alternate magnetic field sources

In contrast to electric fields, **Trak** can combine several sources of magnetic fields. The field components are additive when multiple sources are defined.

BTABLE = FileName Zoff Zmult Boff Bmult
BTABLE = Bucking.DAT (2.0, 1.0, 0.0, 300.0)
Trak can derive fields from expansions based on a table of values of the on-axis magnetic field in both planar and cylindrical geometries. The string parameter *FileName* is the full name of a file with the formal described below. Optionally, you may supply four real numbers as parameters in the command: *Zoff*, *ZMult*, *Boff* and *Bmult*. The parameters modify table values entered in the program according to

$$z_{prog} = Z_{mult}\ z_{tab} + Z_{off}, \tag{9}$$

$$B_{zprog} = B_{mult}\ B_{ztab} + B_{off}. \tag{10}$$

Table 4 shows an example of a magnetic table . The table may contain comment lines, data lines, an *EndFile* command and a *Symmetry* command. The *Symmetry* command must be the first non-comment line in the file and has the parameters *Cylin* or *Rect*. In the *Cylin* option, the data lines contain values of z and $B_z(0, z)$. In the *Rect* option enter x and $B_x(0, x)$. The adjusted values of z should be in the current spatial units defined by *DUnit* and $B_z(0, z)$ should be in tesla. The file may contain up to 256 data and must terminate with the *EndFile* command.

Note: **Trak** uses cubic spline interpolation to analyze the on-axis magnetic field. The method supplies only first and second derivatives. Therefore, the quantities $B_r(r, z)$ and $B_y(y, x)$ vary

linearly in r or y. You must use a finite-element magnetic field solution to investigate effects of non-linear field variations.

BUNI = Bx0 By0 Bz0
BUNI = (0.25E-4 0.0 0.0)
This command defines spatially-uniform magnetic field components. The feature could be used, for example, to simulate the effects of the earth's magnetic field on an electro-optical device. Enter magnetic field values in tesla. Note that uniform field components may be used only in the *Track* tracking mode.

BTHETA AxisCurr RWire
BTHETA = 1200.0 (0.001)
This command adds a toroidal magnetic field generated by an on-axis wire. One application of the feature is modeling wire transport of an intense electron beam. The field is uniform in z. The quantity *AxisCurr* is the wire current in amperes. Use a negative value for a current in the -z direction. The quantity *RWire* is the wire radius in units defined by *DUnit*. The wire radius does not affect the field calculation. It is used to estimate particle losses on the wire. An orbit stops if it reaches a radius less than *RWire*.

8 Single-particle tracking

8.1 Command functions in the *Track* mode

In this chapter we proceed to commands that may appear in the *Particles* section of the input script under the *Track* mode:

```
PARTICLES TRACK
   ...
   (Commands)
   ...
END
```

In this mode **Trak** calculates orbits in the *single-particle limit*. The term implies that the fields created by the particles are negligible compared to the applied fields (*i.e.*, low-current beams). In this case, each particle can be treated independently and the calculation of orbits is a straightforward process. Allowed commands in the *Particles Track* section serve five functions:

- Set starting points for particle orbits.
- Control orbit integrations.
- Define region material properties relevant to orbit tracking.
- Set conditions for orbit stopping.
- Control diagnostic listings that can be generated during orbit integrations.

Many of the commands in the last four categories appear in all particle and field-line tracking modes.

8.2 Starting particles from a list

The simplest way to start particles in the *Track* mode is through list input. Here you specify the species, kinetic energy, start position and direction of from 1 to 20000 particles. Two commands are used for list input:

PLIST

This command signals that a list of particle starting parameters follows in the control script.

The following example illustrates a standard particle list:

```
PLIST
*    Mass Chrg   Eng      x    y      z    px    py    pz
*   ================================================
     0.0  -1.0 0.7399E6  0.1  0.0   -8.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.2  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.3  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.4  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.5  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.6  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.7  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.8  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.9  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.0  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.1  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.2  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.3  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.4  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.5  0.0   -9.0  0.00  0.00  1.00
END
```

The maximum number of lines is 20000. Comment lines (starting with an asterisk) may be included. Each data line contains nine real numbers to represent the following quantities:

- **Mass**. The particle mass in AMU (one atomic mass unit corresponds to $1.660538782 \times 10^{-27}$ kg). An entry of 1.0073 designates a proton. If 0.0 appears in the column, the program inserts the value for an electron. A run in the *Track* mode may contain particles with different values of *Mass* and *Charge*.
- **Charge**. The particle charge in units of e (1.60210×10^{-19} coulomb). An ion has charge +1.0 and an electron has charge -1.0.
- **Energy**. The initial particle energy in eV (1 electron volt = 1.60210×10^{-19} joules).
- **Position**. The particle position (x,y,z) in units set by the current value of *DUnit*.
- **Direction**. Normalized momentum fractions (u_x,u_y,u_z) that give the direction of particle motion. Here, $u_x = p_x/p_{total}$. **Trak** will normalize the numbers; therefore, the sum of the squares of the components need not equal to unity. To represent an initially-stationary particle, set *Energy* = 0.0 and $u_x = u_y = u_z = 1.0$.

The list must terminate with an *End* command.

PFILE = FPrefix
PFILE = Run01

Trak can read particle starting point information from a text file rather than from the control script. The parameter *FPrefix* is the prefix of a file with a name of the form `FPREFIX.PRT` in the current directory. The file contains from 1 to 20000 particle data lines in the format described above terminated by an *End* command. Lines may incorporate any of the standard

TriComp delimiters including spaces and tabs. The file may include comment lines (beginning with an asterisk) and text in any formal after the *End* command.

You can prepare standard particle files (`FPrefix.PRT`) with text editors, spreadsheets or your own programs. Output `PRT` files from **Trak** may be used as input to a subsequent run. Files of high-energy electrons may be transferred to the **GamBet** Monte Carlo code. It is also possible to generate files automatically to represent a variety of distributions using the **GenDist** utility program. **GenDist** can also read `PRT` files created by **Trak** to generate statistical analysis and distribution plots. Finally, you can use **GenDist** to filter distributions to remove unwanted particles.

8.3 Starting particles from an emission surface

Emission surfaces are critical to the operation of **Trak** in the *SCharge*, *RelBeam*, *FEmit* and *Plasma* tracking modes. Emission surfaces are useful in the *Track* mode for a specific application: generation of a distribution of particles with zero kinetic energy from electrodes or surfaces of arbitrary shape. Emission surfaces can be employed only in calculations that include an electric field. In a pure magnetic field, particles with zero kinetic energy would simply remain at the same position.

This section addresses the following questions:

- What is an emission surface?
- How do you create an emission surface in Mesh?
- How does **Trak** identify an emission surface and generate particles?
- How can you control emission surfaces through **Trak** commands?

An emission surface is simply a contiguous line region in the electric field mesh. To review, in the **Mesh** program a line region is one that does not have the *Fill* keyword in the *Region* command. In this case **Mesh** assigns the region number to nodes along the lines and arcs that constitute the boundary, but the program does not change the identity of bordering elements (Fig. 18). A line region need not outline a closed volume. In the *SCharge*, *RelBeam* and *FEmit* modes, the line region must be on the surface of a fixed-potential filled region so that **Trak** can apply the correct emission physics. We can envision that the line region paints a portion of the electrode surface to define the area of emission. Emission surfaces are more flexible in the *Track* mode - the line region may be on the surface of an electrode or within the volumes of dielectric and vacuum regions. If the line region is on the surface of an electrode, set it to the fixed-potential condition in **EStat** and assign the same potential as the electrode. With these settings the presence of the emission surface will not perturb the electric field solution. Similarly, a line region in a dielectric volume should be assigned the same value of relative dielectric constant ϵ_r.

If region number $NReg$ is defined as an emission region, **Trak** searches the mesh and collects all nodes with that number. The program then creates a list of emission facets: element sides that connect two emission nodes (Fig. 18). **Trak** attempts to place the facets in a logical connected order starting from the node nearest the axis and then checks that the facets form a

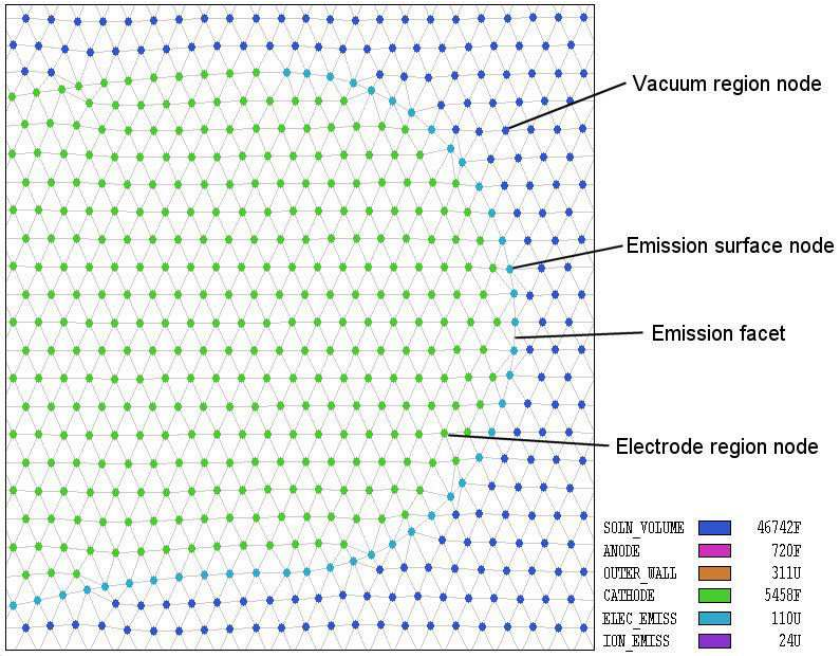

Figure 18: Definition of an emission surface on a conformal triangular mesh.

contiguous set. If we specify that one particle should be created per segment, the code finds the midpoint of each facet. If the facet has dielectric elements on both sides (*i.e.*, the electric field at the midpoint is non-zero), the location is used as the starting point for the particle orbit. If one element has the fixed-potential condition, the code moves the emission point slightly into the dielectric element for a valid field interpolation. **Trak** issues an error message if both elements adjacent to an emission facet are part of a fixed-potential region. To create multiple particles per facet, the code first divides each facet into a number of segments and then starts particles from the midpoints of the segments. Emission surfaces can have arbitrary shape; therefore, emission points may not be uniformly spaced.

The following commands control emission surfaces in the *Track* mode.

EMIT NReg Mass Charge NPerSeg
EMIT(5) = (0.0, -1.0, 3)

This command states that nodes with region number *NReg* constitute an emission surface. The real number parameters *Mass* and *Charge* give the mass and charge of the particles that will be created. Enter the mass in AMU. If 0.0 appears in the column, the program inserts the value for an electron. The quantity *Charge* is the particle charge in units of fundamental charge. An ion has charge +1.0 and an electron has charge -1.0. Note that only one particle species can be created on an emission surface. You can define several emission surfaces in a run, each with a different particle species. The final integer parameter, *NPerSeg*, is the number of segments per facet. For example, the choice $NPerSeg = 3$ instructs the code to create three particle orbits per facet.

START NReg XStart YStart
START NReg ZStart RStart
START(5) = (0.0, 5.0)
In processing an emission surface **Trak** must start with a node at the end of the line region to arrange the facets properly. Sometimes the end point may not be the point closest to the axis, or there may be an ambiguity (such as a horizontal line). The code issues an error message if facet ordering fails. You can correct the problem by actively setting the endpoint of the line region. The integer parameter *NReg* is the region number of surface. Enter the coordinates of a point close to the end in units set by the current value of *DUnit*. To fix problems, note that *Trak* records information on the selection of facets and the ordering procedure in the listing file `FPrefix.TLS`.

8.4 Controlling orbit integrations

The following commands control the numerical solution of particle orbits.

DT Dt
DT = 1.0E-12
This command sets the integration time step (in seconds). Generally, the quantity *Dt* should be less than the minimum time it takes for a particle to cross an element. Lower values give higher accuracy but extend the run time. If the command does not appear, **Trak** makes a guess based on estimates of the minimum element size and maximum particle velocity. The velocity is determined by checking the initial kinetic energies and/or the maximum change in potential energy in the solution space. In solutions with a magnetic field, it is essential that the time step is much smaller than the gyration period:

$$\Delta t \ll \frac{2\pi}{\omega_g} = \frac{2\pi\gamma m_0}{qB}. \tag{11}$$

In Eq. 11, B is the maximum value of magnetic flux density. The particle has relativistic energy factor γ, charge q and rest mass m_0. In solutions with strong magnetic fields, you should always set *Dt* manually, ensuring that it satisfies Eq. 11.

DT DTRef MASS
DT = 2.3E-8 (MASS)
This form of the *Dt* command includes the keyword *Mass*. The form must be used whenever a simulation contains particles with different masses. The quantity *DtRef* is a reference time step for particles with a mass of 1.0 AMU (atomic mass unit). The actual time step used for a particle orbit is given by

$$\Delta t = \frac{\Delta t_{ref}}{\sqrt{m_0}}, \tag{12}$$

where m_0 is the particle rest mass in AMU. Therefore in a mixed simulation with protons and electrons the time step for electron orbits will be about 1/43 of the time step used for the protons. As an example, consider the calculation of electron extraction and ion back-flow in a gun with 250 kV applied voltage. Suppose the minimum element size is 1.0 mm. The maximum

velocity of a proton would be 6.92×10^6 m/s. Therefore, we set $\Delta t_{ref} = 0.001/(6.92 \times 10^6 = 1.44 \times 10^{-10}$ seconds.

TMAX TMax
TMAX = 3.0E-9
This command sets the maximum duration (elapsed time) of orbits. Enter *TMax* in seconds. A common use of the command is to prevent infinite calculations in a system with closed orbits. In the event an orbit exceeds the maximum duration, **Trak** determines its final position and momentum by an accurate interpolation to *TMax*. This feature is useful for analyzing isochronous systems. The default value is $T_{max} = \infty$.

DMAX DMax
DMAX = 15.0
This command sets a maximum total length for orbits. Enter the distance in units set by *DUnit*. When the distance exceeds D_{max}, the code determines the stopping point by interpolation so that all orbits have almost exactly the same length. This command gives an alternate way to prevent infinite orbits. The default value is $D_{max} = \infty$.

NTRACKMAX = NTrackMax
NTRACKMAX = 400
This command sets the maximum number of integration steps. Its main use is to prevent infinite orbits. The default value is $NTrackMax = 20,000$.

The following two commands set specialized controls – it is unlikely that you will need to use them.

INTERP E [LIN,LSQ]
INTERP B [LIN,LSQ]
INTERP BB [LIN, LSQ]
INTERP(B) = LIN
By default **Trak** uses a second-order least-squares fit procedure to calculate electric and applied magnetic field during an integration (*LSQ* option). This setting should be suitable for almost all runs. The *LIN* option activates a simple linear routine that returns a uniform value of electric field, applied magnetic field or beam-generated magnetic field in each element. Although it yields reduced accuracy, the *LIN* option may be useful if the solution volume includes small enclosed regions. In a *cul de sac*, the program may have difficulty locating enough points to make the *LSQ* fit.

NSEARCH E NSearchE
NSEARCH B NSearchB
NSEARCH(E) = 3
Trak must identify elements in the electric and/or magnetic field meshes occupied by particles during orbit integrations. The procedure is challenging on the conformal meshes used in

TriComp because there is no unique relationship between a node's index and its position. Therefore it is necessary to check individual elements. Furthermore, separate searches must be performed for the calculations of electric and magnetic fields because they are defined on independent meshes. To speed the process **Trak** uses a fast search procedure. The elements occupied by the orbit starting point are located by full searches on each mesh. In subsequent steps, local searches are made in the vicinity of the last occupied element. The parameter $NSearchE$ governs the width of the local search region in the electric field mesh. For a good choice of Dt, the code defaults of $NSearchE = NSearchB = 6$ are sufficient. You can speed up integrations by choosing a smaller value. Larger values may be necessary if a mesh contains regions with very small elements.

8.5 Stopping conditions

The implementation of precise stopping conditions is a critical component of a charged-particle simulation. For example, you may want to determine whether a particle strikes a detector and then to find the exact particle parameters at the impact point. Similarly, in characterizing a lens we want a precise prediction of beam properties in a plane normal to the axis. This section describes the wide variety of available methods to stop particles in **Trak**.

We shall first address orbit termination at region surfaces in the finite-element meshes. Regions in the electric and/or magnetic field meshes can be assigned one of three properties that affect particle transport.

- **Vacuum**. Particles move unimpeded through *Vacuum* elements.
- **Material**. Particles stop when they enter a *Material* element.
- **Secondary**. This feature applies only to electron tracking. Electrons are re-emitted when they enter a *Secondary* element. The process of secondary emission is described in detail in Chap. 14.

When a particle enters a *Material* or *Secondary* element, **Trak** employs a sophisticated procedure to project the orbit back to the entry surface to find the final or re-emission position.

Trak makes default assignments of region material properties. In the electric mesh, all elements with $\epsilon_r = 1.0$ are set as *Vacuum*, and all elements with fixed potential or $\epsilon_r \neq 1.0$ are set as *Material*. In the magnetic mesh elements with $\mu_r = 1.0$ are initialized as *Vacuum*, and elements with fixed vector potential or $\mu_r \neq 1.0$ are *Material*. You can use the following commands to change the assignments.

VACUUM E RegNo
VACUUM(E) = 2
VACUUM B RegNo
VACUUM(B) = 5
These commands set all elements with region number *RegNo* in the electric or magnetic mesh to the *Vacuum* condition.

MATERIAL E RegNo
MATERIAL(E) = 4
MATERIAL B RegNo
MATERIAL(B) = 1
This command sets all elements with region number *RegNo* in the electric or magnetic mesh to the *Material* condition.

SECONDARY E RegNo DeltaMax0 EngMax0
SECONDARY(E) = 8 2.35 300.0
SECONDARY B RegNo DeltaMax0 EngMax0
SECONDARY(B) = 3 1.2 250.0
This command sets all elements with region number *RegNo* in the electric or magnetic mesh to the *Secondary* condition. Chapter 14 discusses the properties of secondary emission materials and the interpretations of the additional parameters.

The next set of commands controls the definition of special planes in the solution space to stop, to diagnose or to reflect particles. Normally the location of the plane should be inside the common volume defined by the electric and/or magnetic field meshes. If the script contains the *Boundary* command in the *Fields* section, the planes may be located anywhere inside the specified boundary area.

STOP [UP,DOWN] [X,Y,Z,R] StopPosition
STOP(UP, X) = 5.50
Particle orbits terminate when they cross a stopping plane. **Trak** projects the orbit back to the plane to find precise values of final position, momentum, total distance and elapsed time.

The **Stop** command has three parameters with the following options:

- *UP, DOWN*. The string parameter sets the direction of particle motion along the axis normal to the stopping plane.
- *X, Y, Z, R*. The string parameter specifies the axis normal to the stopping plane. Note that the R option can function in simulations with both cylindrical and planar symmetry. In the latter case, the program records parameters when the particle crosses a circular boundary in the x-y plane.
- *StopPosition*. The real-number parameter is the position of the stopping plane along the normal axis. Enter the value in units set by the current value of *DUnit*.

As an example, suppose a source at $z = 0.0$ creates electrons trapped in an axial magnetic field. We want to stop electrons if they reach an axial displacement of ± 5.0 cm from the source. In this case, we include the commands

```
STOP (UP, Z) = 5.0
STOP (DOWN, Z) = -5.0
```

DIAG [UP,DOWN] [X,Y,Z,R] DiagPosition
DIAG (Up, X) = 4.54

Trak records a set of final particle parameters that are used in the *Diagnostic* commands discussed in Chap. 15. The values may also be recorded in a standard particle file using the *PartFile* command. Sometimes it is useful to record these parameters and then to continue the particle orbits. For example, suppose you want to find the distribution at the waist point of a high-current beam. If the particles stopped at the waist, the electrostatic fields would be incorrect because there was no downstream space-charge assignment. In this case you can define a *Diag* plane. When a particle crosses such a plane, **Trak** records interpolated particle parameters in the plane and continues the orbit until a stop condition occurs. Allowed parameters are the same as those for the *Stop* command. You can define multiple *Diag* planes - note that parameters are recorded at the last plane encountered.

RECORD [UP,DOWN] [X,Y,Z,R] RP1 RP2 ... RPN
RECORD (Up, Z) = 19.0 20.0 21.0

With this command you can record precise particle parameters at a number of positions normal to an axis. There are three differences from the *Diag* command: 1) you can define up to 10 record planes, 2) the information is transferred to multiple `PRT` files during the orbit integrations and is not available for operations with commands of the Diagnostics section and 3) only a single *Record* command may appear in the script. The second and third string parameters specify the direction of particle motion and the axis for normal planes. In cylindrical simulations the most common choice would be *UP Z*. Up to 10 real-number parameters may follow giving the positions along the axis for diagnostics. If there are three entries, **Trak** opens three files with names of the form `RunName01.PRT`, `RunName02.PRT` and `RunName03.PRT` and records values of energy, position and momentum as each particle crosses the corresponding planes. In simulations of the type *SCcharge*, *RelBeam*, *FEmit* and *Plasma*, data are recorded only on the final cycle.

The following rules apply to the *Record* command:

- The file may contain only one *Record* command. Therefore, all planes are normal to a single axis.
- The positions $RP1$, $RP2$, ...RPN must appear in order of increasing value for the *UP* option or decreasing value for the *DOWN* option.
- The command must contain between 1 and 10 position values.
- For the *UP* option, particles with starting positions less than $RP1$ will not be recorded in any of the files. For the *DOWN* option, a particle must start at a position above $RP1$ to be recorded.
- The recording process may not function correctly if the spacing between planes is less than the particle integration step size.

REFLECT [UP,DOWN] [X,Y,Z,R] ReflectPosition
REFLECT (Down, Y) = 0.0
Reflection planes are useful to simulate periodic systems with symmetry in both the fields and particle motions. When a particle crosses a reflection plane, **Trak** projects its position back to the plane and reverses its momentum normal to the plane. Allowed parameters are the same as those for the *Stop* and *Diag* commands. Multiple *Reflect*, *Diag* and *Stop* commands may appear in a control script.

8.6 Orbit listings

The final three commands control information that can be recorded during orbit tracking.

LISTON NStep [P,E,B]
LISTON (2, B)
This command initiates records of orbits in the listing file (`RUNNAME.TLS`). This information is more detailed than the simple coordinate data recorded in the plot file (`RUNNAME.TOU`). The list option is normally deactivated because complex runs could generate huge listings. The optional parameter *NStep* is the number of integration steps between records. The optional key symbol P, E or B determines the type of information recorded. Under the P option (the default) the data lines contain the following quantities: step number, elapsed time, coordinates (in units set by the current value of DUnit), total distance, and normalized momentum components ($\mathbf{p}_{norm} = \mathbf{p}/m_0c$). In the field modes ($E$ and B) **Trak** records coordinates of the integration points and the calculated field components. An extract from a file is shown in Table 5. The *ListOn* command is valuable for debugging runs and checking field interpolations along particle trajectories.

ORBINFO
Use the *OrbInfo* command on those occasions when particles mysteriously stop or refuse to move. When the command is issued **Trak** writes a record in the listing file of the orbit termination status. Table 6 shows an example.

PLOTOFF
This command suppresses generation of the plot file, `RUNNAME.TOU`. This feature saves disk space and time if you are making a run with a large number of particles and you are interested only in the starting and ending parameters.

PLOTSKIP NPlotSkip [NPartSkip]
Trajectories with a large number of steps or distributions with a large number of particles may lead to very large plot files. Use this command to control the size of the `TOU` file. The parameter $NPlotSkip$ is the number of integration steps per plot vector. The default is $NPartPlot = 5$. Use a larger number if the time step is very small (for instance, it a strong magnetic field). Use $NPartPlot = 1$ to make a detailed record of trajectories in the `TOU` file. The optional parameter *NPartSkip* limits the number of trajectories included in the `TOU` file. For example, if $NPartSkip = 5$, only the trajectories for particles 5, 10, 15, ... are included. The default is $NPartSkip = 1$ (all trajectories plotted).

Table 5: Extract of orbit list under the E option

```
Tracking particle number   2
 NTrack      t           x           y           z
 ======================================================
      0  0.0000E+00  5.0000E-02  0.0000E+00 -2.4990E+00
      5  4.0997E-11  5.0012E-02  0.0000E+00 -2.2599E+00
     10  8.1993E-11  5.0105E-02  0.0000E+00 -2.0223E+00
     15  1.2299E-10  5.0400E-02  0.0000E+00 -1.7873E+00
     20  1.6399E-10  5.1073E-02  0.0000E+00 -1.5574E+00
     25  2.0498E-10  5.2292E-02  0.0000E+00 -1.3364E+00
     30  2.4598E-10  5.3994E-02  0.0000E+00 -1.1286E+00
     ...

          Dist         Ex          Ey          Ez
     ================================================
       0.0000E+00 -9.0412E+00  0.0000E+00  3.8246E+04
       2.3908E-01 -2.7417E+03  0.0000E+00  5.0646E+04
       4.7669E-01 -6.7280E+03  0.0000E+00  9.1283E+04
       7.1166E-01 -1.2867E+04  0.0000E+00  1.7572E+05
       9.4158E-01 -1.9363E+04  0.0000E+00  3.1533E+05
       1.1626E+00 -1.7718E+04  0.0000E+00  4.7408E+05
       1.3704E+00 -2.4400E+03  0.0000E+00  5.5682E+05
       ...
```

Table 6: Example of a termination statement

```
Termination status of particle number   2
  Orbit outside the boundary of the electric field solution
Final position: [  5.4806E-03,  0.0000E+00,  3.9995E+00]
Final momentum: [ -2.0874E-03,  0.0000E+00,  1.9884E-01]
Final kinetic energy:   1.0006E+04
```

8.7 Particle starting times

In *Track* mode runs that include modulations, the default condition is that all particles start at $t = 0.0$ s. You can specify individual particle start times relative to the modulation function by adding a column to particle lists or files. In this case, the data lines have the form:

```
PLIST
*    Mass Chrg   Eng       x    y      z    px    py    pz     ts
*   ============================================================
     0.0  -1.0 0.7399E6  0.1  0.0   -8.0  0.00  0.00  1.00  0.0E-6
     0.0  -1.0 0.7399E6  0.2  0.0   -9.0  0.00  0.00  1.00  0.1E-6
     0.0  -1.0 0.7399E6  0.3  0.0   -9.0  0.00  0.00  1.00  0.2E-6
     0.0  -1.0 0.7399E6  0.4  0.0   -9.0  0.00  0.00  1.00  0.3E-6
     0.0  -1.0 0.7399E6  0.5  0.0   -9.0  0.00  0.00  1.00  0.4E-6
     ...
END
```

Specify the start time in units of seconds. If an entry is missing, **Trak** takes the default $t_s = 0.0$ s. When a start time is given, the modulation function is evaluated during an orbit trace as:

$$f_e(t + t_s), \quad f_b(t + t_s), \tag{13}$$

where t is the elapsed time from the initiation of the particle trace and t_s is the start time. The adjusted time is recorded in particle lists in the `TLS` file and in the `TOU` file, Note that any particles that start from an emission surface have $t_s = 0.0$ s.

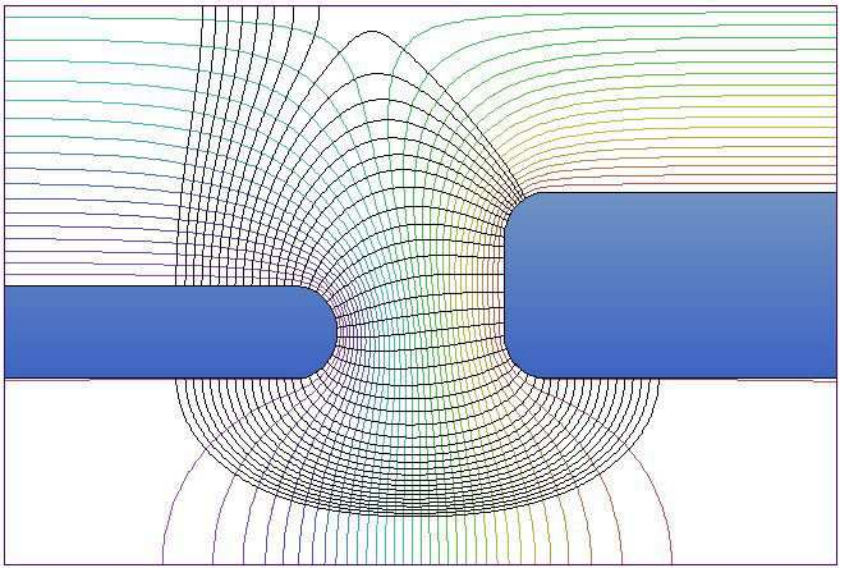

Figure 19: Field lines generated from an emission surface on the left electrode.

9 Tracking electric-field lines

9.1 Script commands

This chapter reviews tracing of electric field lines in **EStat** solutions. **Trak** employs advanced interpolation techniques to find precise intersection points of field lines with the solution boundary or electrode surfaces. This section discusses control commands for the mode, while the second section describes special features for application to ion mobility mass spectrometry.

Allowed commands in the *Particles Fline* section serve three functions:

- Set starting points for traces.
- Control the trace integration.
- Specify listing-file diagnostics.

We begin with commands to initiate field line traces.

FLIST

This command signals that a list of starting positions follows in the control script. Table 7 shows an example of a complete *FList* structure. The maximum number of lines is 20000. Each data line contains four real numbers. The first three ($Xstart$, $Ystart$, $Zstart$) give the initial position in units set by the current value of *DUnit*. To ensure a valid initial field interpolation, the point must not be inside a fixed-potential element. To avoid errors, make sure that the starting positions are displaced a small distance into the dielectric region from the electrode surface. The quantity *Polarity* determines the direction of integration. It may assume the

Table 7: Field line list structure

```
  FLIST
*   Xstart  Ystart  Zstart Polarity
*   ================================
    0.9999  0.0000  -0.50   -1.00
    0.9999  0.0000  -0.75   -1.00
    0.9999  0.0000  -1.00   -1.00
    0.9999  0.0000  -1.25   -1.00
    0.9999  0.0000  -1.50   -1.00
    0.9999  0.0000  -1.75   -1.00
    0.9999  0.0000  -2.00   -1.00
    0.9999  0.0000  -2.25   -1.00
    0.9999  0.0000  -2.50   -1.00
    0.9999  0.0000  -2.75   -1.00
  END
```

values ±1.0. For *Polarity* = +1.0, the integration proceeds along the direction of positive electric field (*i.e.*, from positive to negative potential). The *End* command signals the end of the list.

FFILE FPrefix
FFILE = PReactor
Trak can read field-line starting points from a file rather than the control script. This feature is useful if there are a large number of starting points or if you want to use the same starting points in several different solutions. The parameter *FPrefix* is the prefix of a file with a name of the form `FPREFIX.FLD` in the current directory. Data lines in the file have the same format as those of the *FList* structure. The file list must terminate with a line containing the *End* command.

You can also start field lines from emission surfaces using the *Emit* command. This feature is particularly useful for field lines because we often want to find the destinations of lines that start from different positions on an electrode. The line regions that define emission surfaces can be placed on an electrode or they may follow arbitrary paths in a dielectric region. If an emission segment is on an electrode, **Trak** picks a starting point slightly displaced into the adjoining dielectric element. The mechanics of identifying an emission surface was described in Sect. 8.3.

EMIT NReg NPerSeg [Polarity]
EMIT(5) 3 -1.0
This command states that facets connecting nodes with region number *NReg* constitute an emission surface. The integer parameter *NPerSeg* is the number of segments per facet. The quantity *Polarity* (equal to ±1.0 determines the direction of integration. The default is +1.0.

START NReg XStart YStart
START(5) = (0.0, 5.0) **Trak** orders emission facets in a continuous line starting from the end. By default, the program starts from the node closest to the axis (y_{min} or r_{min}). If there is any ambiguity, use this command to set the endpoint of the emission surface associated with region $NReg$.

The following command controls the line integral:

DS Ds
DS = 0.015
Set the spatial step size for field line integrations. Enter the distance Ds in units set by the current value of $DUnit$. Smaller values of Ds give higher accuracy. If the command does not appear, **Trak** picks a default equal to the diagonal length of the electrostatic solution space divided by 500.0.

Several commands introduced in Chap. 8 for integration control in the Track mode may also be applied:

DMAX = DMax
NTRACKMAX = NTrackMax
NSEARCH E NSearch
INTERP E [LIN,LSQ]

Field lines terminate if they leave the volume of the electric field solution or enter a fixed-potential region. You can define additional stopping conditions with the following commands:

STOP [UP,DOWN] [X,Y,Z,R] StopPosition
DIAG [UP,DOWN] [X,Y,Z,R] DiagPosition
REFLECT [UP,DOWN] [X,Y,Z,R] ReflectPosition

Finally, the following commands can be used to write diagnostic information to the listing file.

LISTON NStep
LISTON (2)
In contrast to the *Track* mode, the listing file data lines in the *FLine* mode always contain the point coordinates and components of the calculated electric field. The key symbols P, E and B are ignored.

ORBINFO
This command gives a record if the termination status of the field line. To gage the accuracy of the integration, **Trak** also records the difference in electrostatic potential between the starting and stopping points compared to the line integral $-\int \mathbf{E} \cdot \mathbf{dl}$. Table 8 shows an example of a termination listing.

Table 8: Termination listing for an electric field line

```
Termination status of field line number   3
  Point inside fixed potential region
Final position: [  1.0000E+00,  0.0000E+00,  1.0001E+00]
Accuracy check
   Phi(Initial):  -5.0000E+03
   Phi(Final):   5.0000E+03
   Phi(Final)-Phi(Initial):   1.0000E+04
   Integral -Edl:   9.9991E+03
```

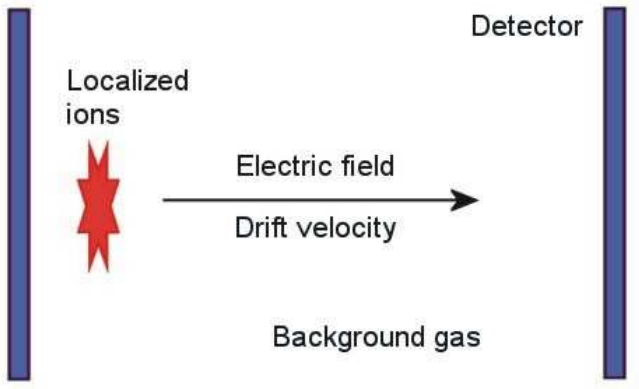

Figure 20: Principle of the ion-mobility time-of-flight spectrometer.

9.2 Ion-mobility spectrometry

Trak includes a useful feature to aid in the design of time-of-flight ion-mobility spectrometers. Figure 20 shows the basic principle. The material to be analyzed is ionized in a gas background (typically air at atmospheric pressure). A electric field is applied and the ions drift with velocity vector $\mathbf{v} = \mu\mathbf{E}$, where $\mathbf{E}$ is the electric field vector (in V/m) and μ is the mobility (in m^2/V-s). Species with different mobilities separate in transit. With a known electric field distribution, the time-of-flight to a detector can be used to infer μ.

For reference, the absolute value of mobility μ is related to the reduced mobility μ_0 (defined at standard temperature and pressure) by:

$$\mu = \mu_0 \left(\frac{760}{P}\right)\left(\frac{T + 273.15}{273.15}\right). \tag{14}$$

In Eq. 14, P is the pressure in mm of mercury and T is the temperature in °C.

Because the drifting ions follow lines of $\mathbf{E}$, the tracking capabilities of the FLine mode are ideal for the application. In addition to calculation of ion trajectories, **Trak** computes the ion time-of-flight for a given mobility. The time increment Δt for an ion to move a distance Δs is:

$$\Delta t = \frac{\Delta s}{|\mathbf{v}|} = \frac{\Delta s}{\mu|\mathbf{E}|}. \tag{15}$$

The time-of-flight from point A to point B is therefore:

$$t_{AB} = \int_A^B \frac{ds}{\mu|\mathbf{E}|}. \tag{16}$$

While tracing a field line **Trak** computes the following integral to yield the *normalized transit time*:

$$\tau_{AB} = \mu\, t_{AB} = \int_A^B \frac{ds}{|\mathbf{E}|}. \tag{17}$$

The quantity τ (in m^2/V) depends only on the nature of the electric field solution. The properties of the background gas and ions are contained in μ. Therefore, a knowledge of the normalized transit time distribution completely characterizes the performance of a detector for any fill gas or ion species.

Figure 21 shows an example of an ion-mobility spectrometer. An entrance grid, a set of biased rings and a detector plane create a field that increases in the z direction. The goal is to capture ions and to compress the distribution to a small-diameter detector. The figure shows equipotential lines and ion drift orbits (corresponding to electric field lines) with entrance radii in the range $0.0 \le r \le 1.2$ cm. An orbit with an initial radius of 1.2 cm has radius 0.393 cm at the detector, a 9.3:1 area compression. If the *PartList* command appears in the *Diagnostics* section for a calculation in the *FLine* mode, **Trak** includes the statistical analysis of Table 9 in the listing file. For the example, the average normalized transit time is $\tau_{avg} = 2.68 \times 10^{-6}$ m^2/V. If the ion mobility is $\mu = 1.36 \times 10^{-4}$ m^2/V-s, the predicted average transit time is 19.7 ms. The dispersion in normalized transit time is $\Delta\tau = 6.03 \times 10^{-8}$ m^2/V. Therefore, we expect that the field-magnitude and trajectory variations limit the detector resolution to about $R \ge \Delta\tau/\tau_{avg} = 2.25\%$.

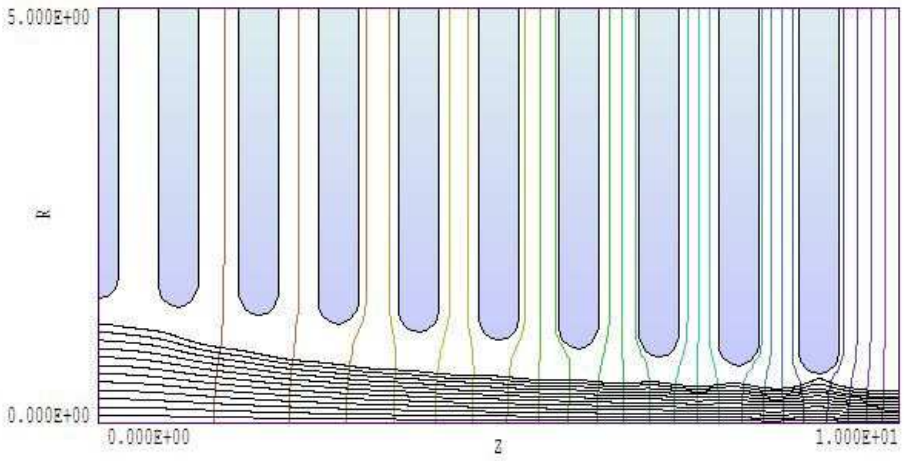

Figure 21: Ion-mobility spectrometer – electrode geometry and ion orbits.

Table 9: Example – statistical analysis for ion-mobility calculations

```
Mobility statistics
Number of values:   12
Normalized transit time (Tau = Int(ds/|E|)
  Tau (average):   2.67762E-06 (m2/V)
  Tau (stddev):   6.02967E-08 (m2/V)
  Tau (minimum):   2.61802E-06 (m2/V)
  Tau (maximum):   2.80662E-06 (m2/V)
Field line length
  D (average):   1.00287E-01 (m)
  D (stddev):   3.18367E-04 (m)
  D (minimum):   9.99953E-02 (m)
  D (maximum):   1.01023E-01 (m)
```

10 Space-charge effects and Child-law emission

10.1 List input for high-current beams

The *SCharge* mode is used to model high-current non-relativistic electron and ion beams. In this mode **Trak** calculates beam space-charge density while tracing particle orbits. Electric fields are then recalculated with the charge contributions of the beam. Several iteration cycles are necessary to find a self-consistent combination of orbits and fields. You can start particles from a list or request that **Trak** create particles on emission surfaces and assign current to satisfy the Child law (space-charge-limited emission) The data lines of the *PList* and *PFile* commands (discussed in Sect. 8.2) have an extra entry to define the current (or current per length in planar simulations) of the model particles.

PLIST

The *PList* command signals that a list of starting points follows in the control script. Each data line contains ten real numbers representing the following quantities: mass, charge, energy, start coordinates (x, y, z), direction vector (u_x, u_y, u_z) and current. Except for the final parameter, the quantities have the same meanings as those discussed for the *Track* option. In cylindrical simulations the current of a model particle is distributed over an annular region (extending from 0 to 2π in θ) centered at the particle position r. Enter the particle current in amperes. For an injected beam of particles uniformly-spaced in radius with uniform current-density, the particle current should be proportional to r_i (the initial particle radius). In planar simulations, enter the current per length (along z) in units of A/m. Note that the current is always a positive number. The direction of the current is given by the sign of the quantity *charge* and the axial velocity of the particle (v_z or v_x). Again, the *End* command terminates the list.

PFILE FPrefix

PFILE = RelInject This command has the same form as the *PFile* command in the *Track* mode. The only difference is the addition of the final quantity *current* in each data line of the file.

Note: In planar simulations in the *SCharge* mode, the x axis must be the direction of average beam motion. Furthermore, the solution must be symmetric about $y = 0.0$. In a typical simulation, you would model only the space $y > 0.0$ and apply symmetry conditions at $y = 0.0$ (*i.e.*, the Neumann condition for the electrostatic potential and a reflection condition for the particles).

10.2 Self-consistent electric field calculations

Self-consistent gun and transport calculations with beam-generated electric fields present a challenge: particle orbits depend on the electric fields while the fields depend upon the orbits through the space charge. **Trak** employs an iterative approach, the *ray-tracing technique*, to

solve the boundary-value problem. The program computes particle orbits in the applied electric fields. The space charge associated with the orbits is then used to find a modified solution of the Poisson equation. Next, the orbits are recalculated in the new fields. The process continues over several cycles until the solution converges. The following commands control the cyclic process.

NCYCLE NCycle
NCYCLE = 15
This command sets the number of orbit-tracking/field-recalculation cycles. The number required for convergence depends on the nature of the problem. A simulation where the beam-generated electric field makes a small contribution to the total field may converge in a few cycles. On the other hand, a simulation where beam fields predominate (*i.e.*, a beam expanding in a field-free transport region) may require 10-30 cycles. In a multi-cycle run, **Trak** writes listing information in response to the *ListOn* command and entries in the plot file only on the final cycle. One sign that a solution has converged is that the electric field recalculation requires fewer than the maximum number of cycles (*MaxCycle* command).

AVG Avg
AVG = 0.20
When beam-generated fields are strong it is necessary to adjust the beam space-charge gradually to achieve a convergent solution. The *Avg* parameter controls the degree of charge deposition. It has a value between 0.0 and 1.0. On any cycle, the space-charge in an element is given by

$$q_{new} = (1.0 - Avg) \times q_{old} + Avg \times q_{beam}, \tag{18}$$

where q_{old} is the previous value and q_{beam} is the value calculated from a sum over orbit traces on the present cycle. For a small value of Avg you should use larger values of $NCycle$. For example, if $Avg = 0.15$, then about 30 cycles would be required to ensure that the space-charge density was within 1% of its equilibrium value. The default is $Avg = 0.15$.

Note: The same averaging factor controls current deposition on the electric field mesh in the *RelBeam* tracking mode.

AVG AvgInit NChange AvgFinal
AVG 0.20 10 0.05
This form for the *Avg* command is useful to aid convergence in solutions with complex electron orbits that are sensitive to emission conditions (such as pinched-beam diodes). In this case, high values of *Avg* give large variations of emitted current between cycles. On the other hand, large values of *NCycle* are necessary to ensure convergence with low values of *Avg*. A good strategy is to use a relatively high initial value to set up an approximate equilibrium, and then to reduce *Avg* for good convergence. In this form of the command, the parameter $AvgInit$ is the initial value while $AvgFinal$ is the final value that is applied for cycles $NCycle \geq NChange$.

For review, the following commands of the *Fields* section (introduced in Sect. 6.3) control electric field recalculations during each cycle:

MAXCYCLE = 2500
RESTARGET = 2.0E-8
OMEGA = 1.95

10.3 Child law emission

In the *SCharge* mode, emission surfaces represent sources of electrons or ions that operate at the space-charge limit. As an example, an emission surface could represent a thermionic cathode when the extracted current density is below the source limit. **Trak** automatically creates model particles on the emission surface and assigns current to satisfy the the Child law. The condition is that the electric field on the surface approaches zero. The numerical method employed is described in S. Humphries, *Numerical modeling of space-charge-limited emission on a conformal triangular mesh*, J. Comp. Phys. **125**, 488 (1996) included with the **Trak** package. Here we shall give a brief description of the procedure so you can understand the functions of parameters in the related commands:

- As in the *Track* and *FLine* modes, **Trak** creates a set of emission facets and determines particle initiation points based on the value of *NPerSeg*.
- A numerical orbit calculation would not be possible if particles were created on the emission surface. The Child condition of zero electric field implies that zero-energy particles would not move, and the calculation would stall.
- To resolve the impasse, **Trak** creates a generation surface by projecting the particle initiation points a distance *DEmit* from the emission surface. Analytic formulas for space-charge limited flow in a planar gap of width *DEmit* are employed to find the appropriate current and kinetic energy to assign to model particles at the generation surface. The problem of stalled orbits does not occur because the kinetic energy and electric field are non-zero at the generation surface.
- **Trak** employs a novel backtracking technique to ensure correct assignment of space charge in the volume between the emission and generation surfaces. The combination of this capability with correction factors for curved electrodes ensures high-accuracy solutions for space-charge-limited flow.
- A program iteration cycle consists of the following operations: 1) create model particles at the generation surface, 2) assign current and kinetic energy based on the present values of local electric field, 3) reverse-track the orbits at fixed energy until they strike the emission surface to set space charge in the gap between the generation and emission surfaces, 4) forward-track the orbits and 5) solve the Poisson equation using the space-charge associated with the orbits. The code makes extensive records in the file `FPrefix.TLS` so you can check the validity of the process.

The following commands control Child-law emission surfaces.

EMIT(NReg) Mass Charge DEmit [NPerSeg JLimit kTs Ns]
EMIT(4) = (0.0, -1.0, 0.05, 3)

This command identifies the line region *NReg* as an emission surface. The region must be on the surface of an electrode (fixed-potential region). The parameters *Mass* and *Charge* have the same meanings as those in the *Emit* command of the *Track* mode. Enter the mass in AMU (atomic mass unit). **Trak** inserts the value of the electron mass if $Mass = 0.0$. Enter the charge of the emitted particles in units of e (i.e., -1.0 for electrons and +1.0 for protons). The quantity *DEmit* is the distance from the emission surface to the generation surface. Enter the value in units set by the current value of *DUnit*. The integer parameter *NPerSeg* is the number of model particles created per facet of the emission surface (default, $NPerSeg = 1$). The real-number parameter *JLimit* (in A/m^2) is a source limit for current emission and kTs (in eV) gives the temperature of the source. The integer Ns is the *splitting parameter* with default value $Ns = 1$. Larger values are useful when $Ts > 0.0$. A particle is split into Ns subparticles. Each subparticle carries a fraction $1/Ns$ of the assigned current and is emitted at a random angle with a distribution width determined by the source temperature. The purpose of the splitting parameter is to improve statistics of downstream particle distributions.

Notes on the *Emit* command

- The parameters $NPerSeg$, $JLimit$, kTs and Ns are optional. If they do not appear, **Trak** uses default values. If they do appear, they must occupy the correct position in the list. For example, kTs must be the seventh entry after the *Emit* command. To set kTs, you must include values for $NPerSeg$ and $JLimit$. For space-charge limited emission, use a large dummy value for $JLimit$.

- The quantity *DEmit* should be small enough so that the generation surface closely follows the contours of the emission surface. On the other hand, the field interpolations necessary to calculate the Child-law current density will be inaccurate if *DEmit* is less than the local element width. As a rule, pick *DEmit* equal to about 1.5 times the local element width. It may be necessary to create small elements with **Mesh** near the emission surface to achieve a good fit.

- If you set a value for the parameter *JLimit*, **Trak** can model mixed space-charge and source-limited emission. The program calculates the Child value at the generation surface and projects it back to the emission surface. The program then chooses the smaller of this value or *JLimit*. The default value for all emission regions is $JLimit = \infty$.

- If you set a value for the source temperature kTs, **Trak** assigns an angular spread to particles leaving the emission surface. The particles are assumed to have a Maxwell distribution in transverse velocity. The direction of particle emission is determined from a unit vector normal to the emission surface and an angular displacement calculated from the transverse kinetic energy and the longitudinal kinetic energy at the surface. The default value for all emission regions is $kTs = 0.0$ eV.

Use the following equation to convert a value of source temperature in °C to electron volts:

$$kT_s(\text{eV}) = \frac{T_s(^\circ\text{C}) + 273}{11,594}. \tag{19}$$

The probability distribution of transverse velocity $v_\perp$ of particles emitted from a source at temperature T_s approximates the Maxwell distribution:

$$f(v_\perp)dv_\perp = v_\perp \left(\frac{m}{kT_s}\right) \exp\left(-\frac{mv_\perp^2}{2kT_s}\right). \tag{20}$$

We can rewrite Eq. 20 in terms of the normalized variable

$$\chi = \sqrt{\frac{m}{2kT_s}}\, v_\perp, \tag{21}$$

as

$$f(\chi)d\chi = 2\chi \exp\left(-\chi^2\right)\, d\chi. \tag{22}$$

The integral of Eq. 22 gives the cumulative probability, a variable in the range from 0.0 to 1.0:

$$\int_0^\chi f(\chi)d\chi = \zeta = 1 - \exp(-\chi^2). \tag{23}$$

The procedure in **Trak** is to generate a random value of ζ and then to find a weighted value of χ from the inverse of Eq. 23:

$$\chi = -\frac{\ln(1-\zeta)}{2}. \tag{24}$$

The angle of the particle relative to the surface normal is given by

$$\Delta\theta \cong \frac{v_\perp}{v_0} = \chi \sqrt{\frac{kT_s}{T_p}}, \tag{25}$$

where T_p is the kinetic energy (in eV) of the particle at the emission surface. Particles are emitted with a random-uniform distribution of azimuth about the surface normal vector. Note that the derivation is valid in the limit that $v_\perp \ll v_0$. **Trak** generates an error message if kT_s is too high relative to T_p. The criterion is that the ratio $kT_s/T_p \leq 0.40$. If you receive the error message *Source temperature too high for the choice of DEmit*, either increase *DEmit* or reduce T_s. Figure 22 illustrates the effect of source temperature.

RSEED ISeed1 ISeed2 RSEED 8754 662

Set a random seed by supplying two integer values. You can use this command to ensure that a calculation involving emission with a source temperature generates identical results each time it is run. If the command does not appear, the program chooses seed values set by the current clock time to generate statistically independent results for each run.

SUPPRESS SVal1 SVal2 SVal3 ...
SUPPRESS 0.20 0.30 0.40 0.60 0.80 1.00

The current density assigned at the generation surface must be suppressed below the Child-law value on the first few iteration cycles. Otherwise the initial current based on the applied field values would be too high and the code could oscillate between high and low current solutions on subsequent cycles. To aid convergence **Trak** applies suppression factors on the first few cycles. The code uses the following default values for space-charge-limited emission:

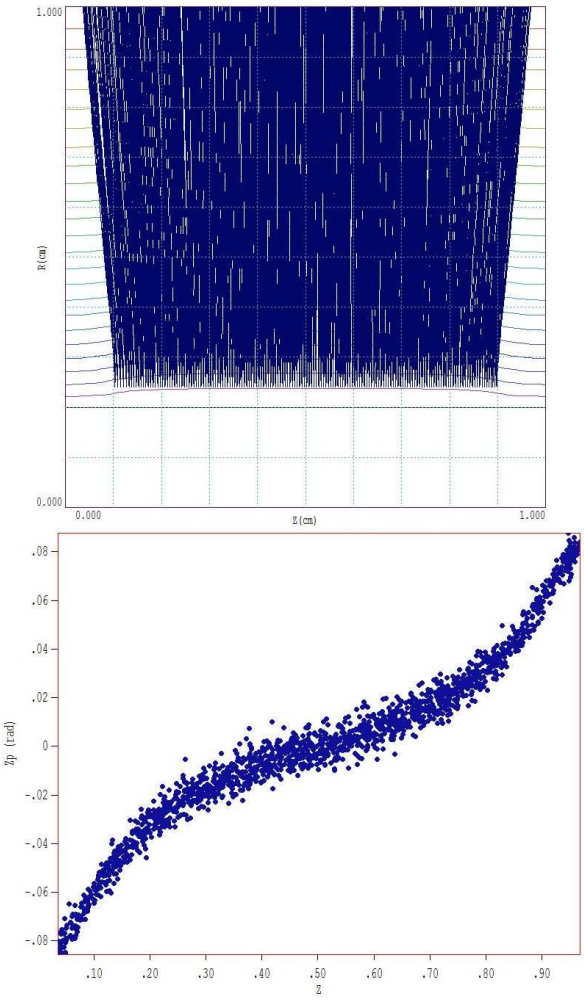

Figure 22: Particles generated at an emission surface with source temperature. Outward radial flow of electrons across a 5000 V gap with $kT_s = 0.25$ eV. Top: equipotential lines and particle orbits. Bottom: axial phase space plot at the outer boundary. The predicted angular divergence is $\Delta\theta \cong \sqrt{kT_s/5000} = 0.007$.

NCycle	Supression value
1	0.250
2	0.300
3	0.500
4	0.750
5	1.000
6	1.000
7	1.000
8	1.000
9	1.000
10	1.000

For unusual solutions you can set you own values. Enter from 1 to 10 real numbers in the range 0.0 to 1.0. Any undefined values default to 1.0.

10.4 Relativistic mode

In the *SCharge* mode, **Trak** does not perform a detailed calculation of beam-generated magnetic fields. The mode is therefore not appropriate for simulations of relativistic electron guns. On the other hand, *Trak* can find accurate results for certain types of relativistic-beam transport problems using the *relativistic approximation*. This approach is faster than the complete calculation under the *RelBeam* tracking option and may give better accuracy for beams with $\gamma \gg 1.0$.

To begin, we shall review some relativistic beam physics. Consider a circular paraxial electron beam traveling along the z axis in free space. The term *paraxial* implies that: 1) electrons have about the same axial velocity ($v_z \cong \beta c$) and 2) orbits make small angles with respect to the axis:

$$r' = dr/dz \ll 1.0. \tag{26}$$

Equation 26 implies that changes in the beam radius r_0 take place over axial distances much greater than r_0. Therefore local beam-generated fields are approximately equal to those of an infinite-length beam.

Suppose the beam carries current I and that the space-charge density is a function of radius, $n(r)$. The electric and magnetic fields created by the beam can be determined from Poisson's equation and Ampere's law:

$$E_z \cong 0.0, \tag{27}$$

$$E_\perp = \ E_r(r) = -\frac{e}{2\pi\epsilon_0 r} \int_0^r 2\pi r' dr' n(r'), \tag{28}$$

$$B_z \cong 0.0, \tag{29}$$

$$B_\perp = \ B_\theta(r) = -\frac{e\beta c\mu_0}{2\pi r} \int_0^r 2\pi r' dr' n(r'). \tag{30}$$

The equations imply that the transverse magnetic force acting on individual electrons is related to the transverse electric force by

$$F_{B\perp} = -\beta^2 \; F_{E\perp}. \tag{31}$$

The quantity β is small compared to unity for non-relativistic beams; therefore, the magnetic force can usually be neglected. In contrast, the repulsive electric force and attractive magnetic force are almost balanced for highly relativistic beams ($\beta \cong 1.0$).

The transverse forces of a planar beam also satisfy Eq. 31. In fact, the relationship holds for paraxial beams of any shape and with any nonuniform distribution of space-charge. The equation also holds if we include field contributions of perfectly-conducting boundaries whose dimensions change gradually in the axial direction. We can prove the result by 1) making a Lorentz transformation by velocity -βc to the average rest frame of the beam, 2) calculating the electrostatic fields resulting from the stationary distribution of charge, and 3) calculating transformed electric and magnetic field values in the laboratory frame. This derivation also shows that because of Lorentz contraction the criteria underlying paraxial beam motion and the definition of gradual changes in boundary conditions is less stringent for relativistic beams. If D is the transverse scale size of the system and L is the axial distance for a significant change in the beam or boundary dimensions, then Eq. 26 holds if

$$\frac{R}{\gamma L} \ll 1.0. \tag{32}$$

where γ is the relativistic energy factor

$$\gamma = \frac{1}{\sqrt{1-\beta^2}}. \tag{33}$$

The quantity γ is related to the rest mass m_0 and kinetic energy T of the particle by:

$$\gamma = 1 + \frac{T}{m_0 c^2}. \tag{34}$$

The theoretical development suggests a strategy to avoid explicit calculations of magnetic fields for relativistic beams. The total transverse force on particles is related to the electric force by:

$$F_{T\perp} = F_{E\perp} - F_{B\perp} = (1-\beta^2) \; F_{E\perp} = \frac{F_{E\perp}}{\gamma^2}. \tag{35}$$

We simply calculate the electrostatic force and then divide by γ^2 to include the effect of the magnetic field. The relativistic mode holds under the following conditions:

- Transverse electric fields arise almost entirely from the presence of the beam space and surrounding conducting boundaries. This condition does not hold in acceleration gaps where there is a strong applied electric field.
- Beam particles move predominantly in one direction and have approximately the same axial velocity $v_z \cong \beta c$.
- The axial scale length for changes in the transverse dimensions of the beam and surrounding boundaries satisfies Eq. 32.

To apply the relativistic approximation in the SCharge mode, simply include the *RelMode* command in the Particles section.

RELMODE
When the command is active, Trak divides the transverse electric force (F_r in cylindrical solutions, F_y in planar) by γ^2 during orbit calculations.

10.5 Restarting a run

Trak has the capability to restore solutions with space-charge and Child-law emission and to proceed over additional iteration cycles. To restart a run, load the electric field file from an initial run. The values of electrostatic potential in this file reflect the presence of space charge. For convergence, the present run should have approximately the same initial conditions as the first run (*i.e.*, electrode potentials, particle lists, emission surfaces,...). In addition to reloading the field solution, you must include the *ReStart* command in the *Particles* section. This command initiates two actions that affect runs with emission surfaces:

- **Trak** automatically sets all suppression factors (*SVal1*, *SVal2*, ...) equal to 1.00 because the initial field values are already almost consistent with Child-law conditions.
- Space-charge averaging is not applied on the first cycle because the initial charge density is close to the final converged distribution.

RESTART
This command signals that an electric field file has been loaded with potential values that are approximately consistent with the presence of beam space charge.

10.6 Other commands

The remaining valid commands for the *SCharge* tracking mode have identical functions to those of the *Track* mode:

DT Dt
DT DtRef MASS
TMAX TMax
START(NReg) XStart YStart
NTRACKMAX NTrackMax
DMAX DMax
NSEARCH [E,B] NSearch
INTERP [LIN,LSQ]
MATERIAL [E,B] RegNo
VACUUM [E,B] RegNo
SECONDARY [E,B] RegNo DeltaMax0 EngMax0
STOP [UP,DOWN] [X,Y,Z,R] = Position

DIAG [UP,DOWN] [X,Y,Z,R] = Position
REFLECT [UP,DOWN] [X,Y,Z,R] = Position
LISTON [NStep] [P,E,B]
PLOTOFF

11 Tracking relativistic beams

11.1 Methods for beam-generated magnetic fields

The simulation of high-current relativistic electron beams is a major challenge for gun-design codes. The drawbacks of approaches to the calculation of beam-generated magnetic fields in previous versions of **Trak** and other codes are described in the reference S. Humphries and J. Petillo, Laser and Particle Beams **18**, (2000), 601 (supplied with the **Trak** package). The article gives a detailed description of numerical methods applied in the present version of **Trak**. This section summarizes critical concepts of the method to help you set up effective solutions. The following discussions are oriented to cylindrical beams - note that the method also applies to planar beams with some restrictions.

The approach has several advantages:

- It is largely automatic and requires little input from the user.
- It gives improved accuracy for field calculations, largely eliminating the problem of filamentation in simulations of highly-relativistic beams.
- The method correctly accounts for boundary currents on conductors. As a result, **Trak** can handle emission from convex or concave cathodes of arbitrary shape as well as beam collection on internal targets.
- It is flexible enough to handle non-laminar beams, counter-flowing particle species and reflex orbits.
- It provides useful diagnostic information on the distribution of beam current striking target surfaces.

The foundation of the method is the calculation of beam-generated magnetic fields on the electric field mesh. There are two motivations for this approach:

- Calculations of electric and magnetic fields are closely coupled, reducing systematic differences.
- Fixed-potential regions in the electric mesh usually correspond to surfaces that carry current and exclude magnetic fields generated by pulsed beams.

The goal of the magnetic field calculation is to find $B_\theta(r,z)$ in the propagation volume consistent with the particle orbits. **Trak** determines values of B_θ at nodes in the electric field mesh and then calculates the field at arbitrary positions by interpolation. The azimuthal magnetic field is related to the enclosed current by:

$$B_\theta(r,z) = \frac{\mu_0 I_{enc}(r,z)}{2\pi r}. \tag{36}$$

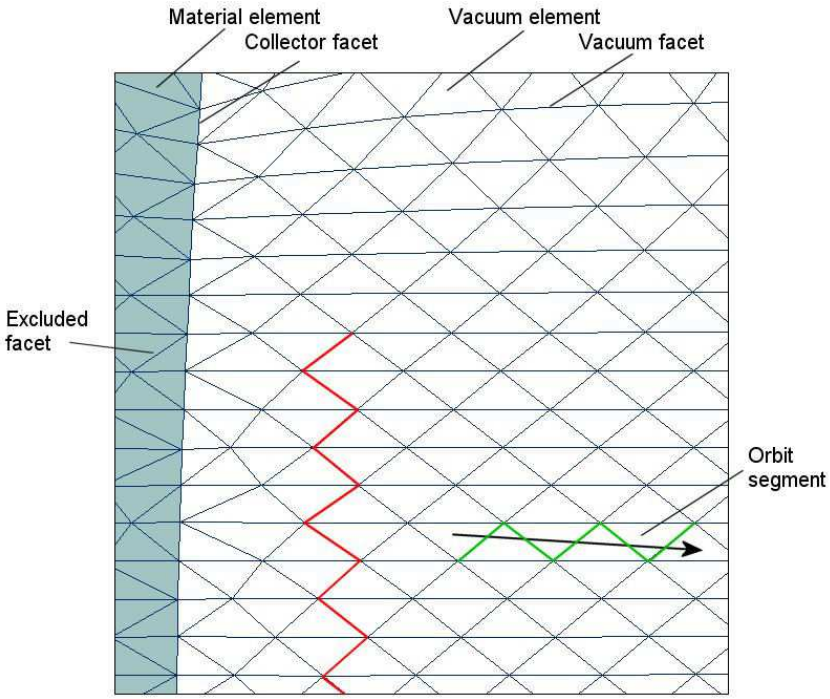

Figure 23: Definition of terms for the calculation of beam-generated magnetic field on a conformal mesh. Red lines show the path from a node to the axis along facets. Green lines show facets intersected by an orbit segment.

The quantity $I_{enc}(z, r)$ is the axial current passing through a plane of radius r at position z. The key to the method is therefore finding the total axial current of model particles that pass through circles defined by the mesh nodes.

Figure 23 shows a segment of the orbit of a particle carrying current I moving through the electric field mesh. The segment corresponds to a time step Δt. The space charge is a volumetric quantity that is conveniently assigned to the elements traversed by the particle. The convention in **Trak** is to augment the space charge of the element at the segment midpoint by an amount $I\Delta t$. In contrast, the current carried by the particle is a flux that is most conveniently accumulated on the surfaces between elements (facets). Here the convention is to assign the current I to all facets traversed by the particle in the $+z$ direction and a current of $-I$ if the particle moves in $-z$. The reference listed above discusses details of current assignment on conformal triangular meshes to ensure that the condition

$$\nabla \cdot \mathbf{B} = 0. \tag{37}$$

is always satisfied. The particle tracking process during an iteration cycle gives a set of facet currents that reflect the fluxes of the model particles. To find the current enclosed within a node, we simply trace a path to the axis (red line in Fig. 23) and add the associated facet currents. If the currents satisfy Eq. 37, the result is independent of the choice of path.

Implementation of the method in **Trak** is complicated by the presence of conductive materials (Fig. 23). There are no particle orbits through these regions; instead, they carry a surface current that depends on the distribution of emitted particles. **Trak** uses the following procedure to represent surface flows consistent with the condition of continuity of current:

- Facets are divided into three categories: *vacuum* facets have vacuum elements on each side, *collector* facets have one vacuum and one material element on each side, and *excluded* facets are surrounded by material elements.

- Currents are assigned to vacuum facets by the procedure shown in Fig. 23. Currents are assigned to collector facets only when an orbit terminates (*i.e.*, passes from a vacuum to a material element).

- After calculating all orbits, **Trak** proceeds to the enclosed current calculation. The first step is to set the quantity along the nodes that constitute collector surfaces. The program assumes that a collector surface is a contiguous set of collector facets that connect to the z (or x) axis. **Trak** searches along the axis to find a collector node and then assigns zero enclosed current. The program then searches for the next node connected through a collector facet and assigns the current of the previous node plus the facet current. The program continues in this manner until it reaches the end of the collector surface.

- After processing all collector surfaces connected to the axis, **Trak** calculates the enclosed current on the remaining nodes connected to vacuum facets. After assigning zero enclosed current to all nodes on the axis, the program works radially outward in layers. When all the nodes in the layer with radial index L are processed, the program moves to layer $L+1$. For each node, the program seeks the shortest connection to a node attached to a vacuum or collector facet in layer L. Enclosed current for the node in layer $L+1$ equals the current of the connected node in layer L plus the contribution of the facet. **Trak** stops if it reaches the boundary of solution volume or if there are no remaining unprocessed nodes.

- The final step is to find values of B_θ at the nodes, inserting the enclosed current and node radius in Eq. 36.

Figure 24 shows sample results of the calculation. The example illustrates a limitation of the method. The calculated fields are valid in the beam propagation region but exhibit dead areas with zero field at large radius. The problem occurs in the region on the right-hand side because the collector surface of the anode does not contact the axis. Therefore, **Trak** can not assign enclosed current values along the surface. Points in the dead space that lie in the shadow of the anode follow facet paths toward the axis that terminate on anode nodes with zero enclosed current. A similar effect occurs on the cathode side because of the break between the cathode and focusing electrode.

The dead spaces do not affect the simulation because they occur outside the region of beam propagation. On the other hand, there may be cases where a representation of the beam-generated field is required over the full solution volume. For example, suppose you want to trace electrons emitted from the outer radius of the focusing electrode. In that case, you must construct the mesh so that all nodes of interest are connected to the axis through a chain of collector facets. The example in the next section illustrates such a setup.

The simulation of space-charge-limited electron emission from a convex cathode (Fig. 25) raises an interesting issue. The collector facets on the outside of the cathode are shadowed from the axis. Furthermore, electrons in the plot start from a generation surface and never pass through the cathode surface. In this case, we might ask how **Trak** assigned facet currents on

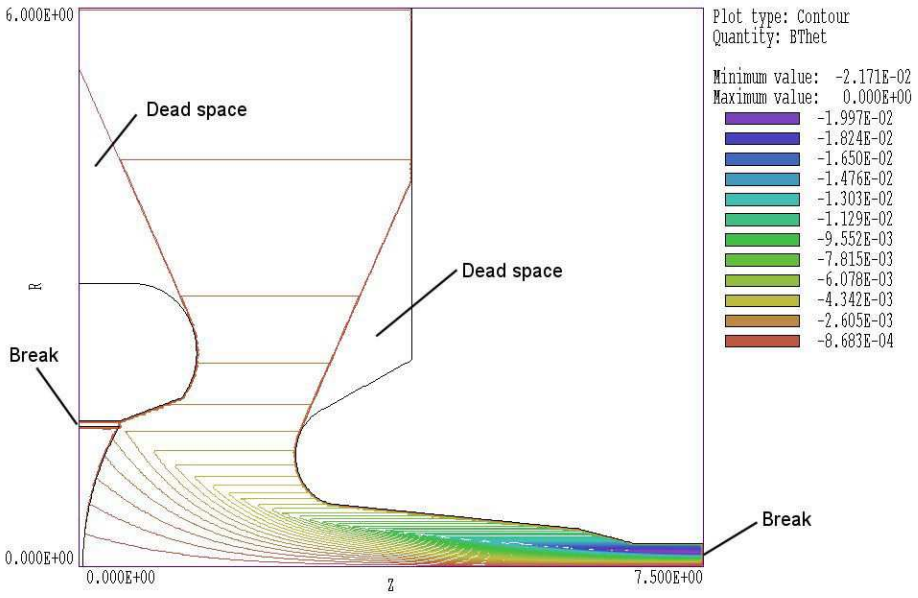

Figure 24: Contour lines of B_θ showing missing values in areas shaded by re-entrant collector surfaces that do not connect to the axis.

the cathode to give the valid field solution shown. The answer is that facet current assignment occurred during back-tracking. Recall from Sect. 10.3 that in a calculation with space-charge-limited emission, **Trak** projects orbits back toward the generation surface at a fixed velocity to ensure correct assignment of space charge in the gap. Facet current assignment is also performed during backtracking. When the particle strikes the cathode, the forward current I is assigned to the intersected facet. Note that the backtracked orbits are not recorded in the `TOU` file.

The method of current assignment can also be applied to the calculations of beam-generated fields in planar simulations. In this case **Trak** determines values of B_z at node points with average beam motion in the x direction. The calculation is performed in the region $y \geq 0.0$ with the assumption that the enclosed current in x equals zero along the line $y = 0.0$. Therefore, the line $y = 0.0$ should represent a beam symmetry axis. The line must have the Neumann condition in the **EStat** calculation and must be defined as a reflection boundary in **Trak**.

11.2 Commands and controls

When **Trak** enters the *RelBeam* mode, the program automatically sets up the mechanisms and allots memory to perform the calculation of B_θ on the electric field mesh. Magnetic field contributions are included in the particle equations of motion. With a good electric field mesh and valid choice of emission surface properties, the calculation proceeds automatically with good reliability. There are two commands unique to the *RelBeam* mode:

ZEROPOINT ZAxis
ZEROPOINT -10.0
In the calculation of beam-generated magnetic fields, the zero point is a position on the symmetry axis where B_θ or B_z equals zero. The default zero point position is on the axis halfway between z_{min} and z_{max} (or x_{min} and x_{max} in planar solutions). In cases where upstream and/or

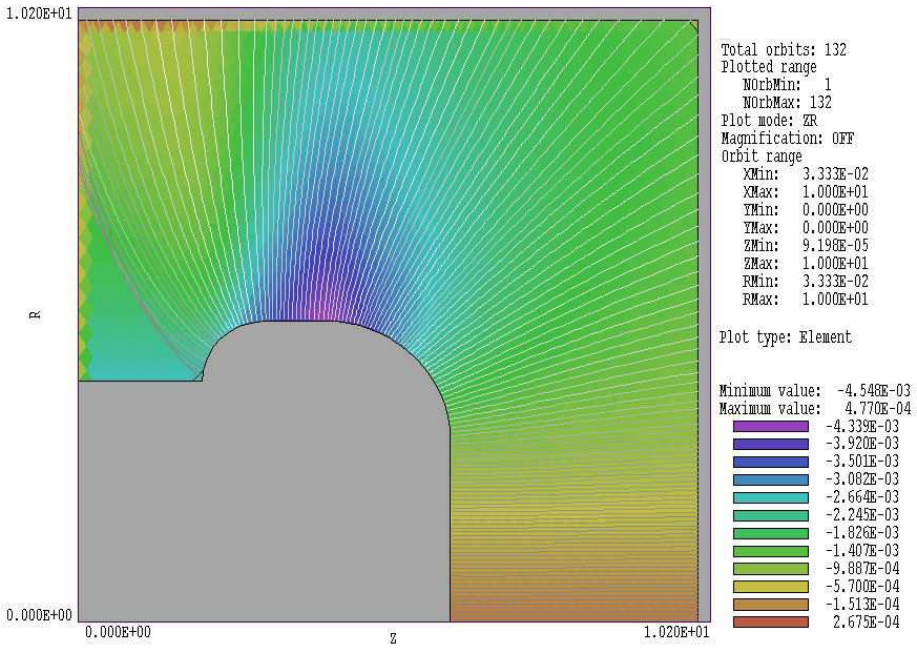

Figure 25: Variations of B_θ for electron emission from a convex cathode.

downstream electrodes intersect the axis, **Trak** must identify facets on their surfaces to calculate the contribution of boundary currents to the beam-generated magnetic field. The procedure is to start from the zero point and to walk along the axis until the intersection of an electrode is detected. The program then identifies the surface facets of the source or target electrode and marks them accordingly. For some geometries, the default zero point may be inside an electrode and **Trak** will issue an error message. In this case, use the *ZeroPoint* command to set a valid position manually. If there is only a source electrode, pick any downstream point on the axis. If there are both source and target electrodes, pick a point between the two. Specify the coordinates in units set by *DUnit*.

BACKTRACK

Particles generated from emission surfaces automatically backtrack and make current contributions to collector facets on the source surface. This may not be the case when particles are created with the *PList* and *PFile* commands. In this case, the particles from an electrode must be generated within a vacuum element near the fixed-potential surface for a valid electric field interpolation. As a result, the nodes on the electrode surface may have zero values for enclosed current and B_θ. The invalid field values would give inaccurate field interpolations near the surface. When the *BackTrack* command appears, **Trak** performs an initial backtrack operation on list-input particles. The program reverses the momentum and sign of the current for each particle and follows the orbits until they terminate. Current is assigned to the collector facet intercepted by the particle orbit. Note that backtracking may also improve the distribution of space-charge and the calculation of electric fields near the surface.

The following commands have special interpretations in the *RelBeam* mode:

PLIST [Pos, Neg]
PFILE [Pos, Neg]
EMIT(NReg) Mass Charge DEmit [NPerSeg JLimit Ts Ns] [Pos,Neg]
Trak determines whether current flows in the $+z$ or $-z$ directions when particles pass through vacuum facets by comparing the directions of the orbit and facet vectors. On the other hand, when a particle strikes a collector facet the program may not be able to determine whether current exits through the right or left-hand boundaries. The axial direction of current along conducting surfaces must be specified for valid orbit solutions in the case of internal collectors or sources. You can add additional string parameters at the end of the *PList*, *PFile* and *Emit* commands to signal whether the associated particle flow on sources and collectors is in the direction from left to right [*Pos*] or from right to left [*Neg*]. The default is *Pos*.

RELMODE [Zrmin Zrmax]
RELMODE [XRelMin XRelMax]
RELMODE = (20.00 40.00)
The relativistic mode calculations discussed in Sect.10.4 may seem unnecessary in the *RelBeam* tracking mode where the program explicitly calculates forces from beam-generated fields. Nonetheless, there is an important reason to include it. For highly relativistic beams ($\gamma \gg 1.0$) passing through transport regions where there is no applied electric field, it is more accurate to determine the transverse force from Eq.35 than to make separate calculations of the electric and magnetic components. Suppose we wanted to investigate a relativistic electron gun where the beam is matched to a long transport solenoid. The best approach would be to calculate magnetic and electric forces explicitly in the injector region and then to switch to the relativistic mode in the transport region. This form of the *RelMode* command enables such a calculation. The two real-number parameters define limits along z (cylindrical) or x (planar). The code applies the relativistic mode only in the region $Z_{rmin} \leq z \leq Z_{rmax}$. Enter values in units set by *DUnit*. The defaults are $Z_{rmin} = -\infty$ and $Z_{rmax} = +\infty$.

11.3 Application example - simulation of a pinched electron beam diode

The example described in this section illustrates several **Trak** techniques. The application, a self-pinched high-intensity electron beam with ion flow effects, is a challenge for a ray-tracing codes. Figure 26 shows the geometry. The anode is a tungsten rod of diameter 1.0 mm located on the axis. The goal is to produce a compressed electron spot on the anode tip with an axial extent less than 1.0 mm. A pulsed voltage of +1.2 MV is applied to the anode. The cathode is a thin coaxial cylinder with an inner radius of 5.0 mm. The figure shows a magnified portion of the simulation volume. The assembly is inside a grounded vacuum chamber with a radius of 35.0 mm. The simulation covers an axial region from -25.0 mm to 25.0 mm.

The file `PINCHBEAM.MIN` (included in the example library) defines the variable resolution mesh. A line region is defined for electron emission on the cathode surface (red line in Fig. 26). The intense electron flow at the anode immediately creates a surface plasma. Therefore, we include the possibility of space-charge-limited ion emission from the anode tip (green line in Fig. 26).

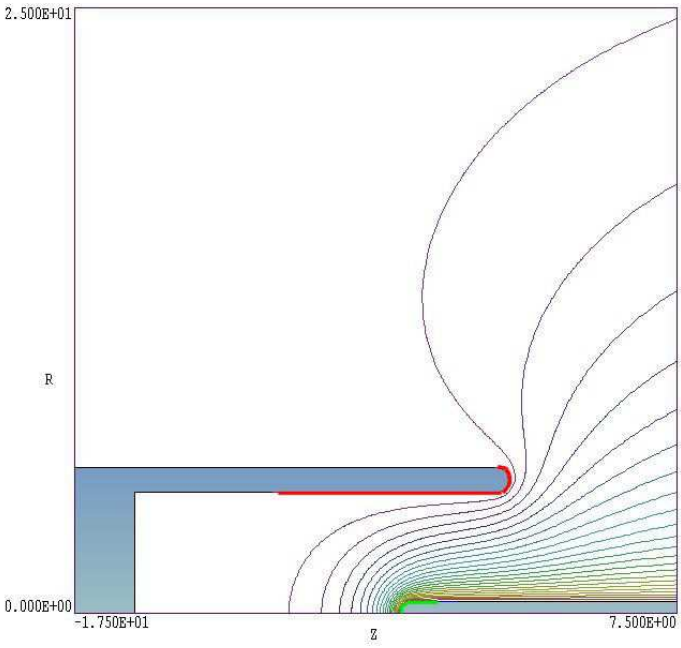

Figure 26: Geometry of the PINCHBEAM example - magnified view of the acceleration gap. Red line: electron emission region on the cathode. Green line: ion emission region on the anode.

Table 10 shows the file PINCHBEAM.TIN with added line numbers. Note the use of the multiplication factor in the *EFile* command of Line 2. The solution PINCHDIODE.EOU was generated with an applied potential of 1.0 V. We can then use the normalized solution with a multiplication factor to investigate the effects of changes in applied voltage. The solution must be performed gradually because of the complex electron orbits in the strong beam-generated magnetic fields. Note the large value of *NCycle* and small value of *Avg* in Lines 8 and 9. Despite the complexity of particle motion, the solution exhibits good convergence. The emitted current of 16.65 kA in Cycle 33 differs by only about 1 percent from the current in Cycle 32. Lines 10 and 11 defined emission surfaces for the electrons and protons. The string *Pos* in Line 10 indicates that the electrons move from left to right while *Neg* in Line 11 signals that ions move from right to left. The zero point (Line 13) for the beam-generated magnetic field is on the axis between the cathode and anode.

The simulation includes a large number of particles (5 per facet) for good statistics. Note that the *Mass* form of the *Dt* command has been employed because there are two particle species. With regard to the reference time step, the minimum element width in the gap region is 0.1 mm. The velocity of a proton (1.0 AMU) is about 1.52×10^{-7} m/s, giving a minimum element transit time of 6.6×10^{-12} s. The value $DtRef = 1.8 \times 10^{-11}$ s was sufficient to give a good resolution of the orbits.

The equilibrium current values were 15.20 kA for electrons and 1.12 kA for protons. The effect of ions in neutralizing space-charge was critical to achieve high-current operation and a well-formed pinch. The current dropped by about a factor of 3 if the ions were omitted. The *BBBoundary* command in Line 16 of the script initiates a listing of accumulated facet currents on the anode rod. The results show that about 78% of the electron current impinged on the first 1 mm length of the anode. An inspection of the electron orbits in Fig. 27 shows that conditions are marginal for a well-contained pinch. Electrons emitted from the inner edge of

Table 10: Contents of the file PINCHBEAM.TIN

```
01: FIELDS
02:   EFile = PINCHDIODE.EOU  1.2E6
03:   MaxCycle = 200
04:   ResTarget = 1.0E-7
05:   DUnit = 1000.0
06: END

07: PARTICLES RELBEAM
08:    NCycle = 33
09:    Avg = 0.20 (20, 0.05)
10:    Emit(5) 0.0 -1.0 0.15 5 POS
11:    Emit(6) 1.0 1.0 0.15 5 NEG
12:    Dt: 1.2E-11 Mass
13:    ZeroPoint: -10.00
14: END

15: DIAGNOSTICS
16:  BBDump: PINCHDIODE
17:  BBBoundary (6)
18:  EDump: PINCHDIODEP
19: END

20: ENDFILE
```

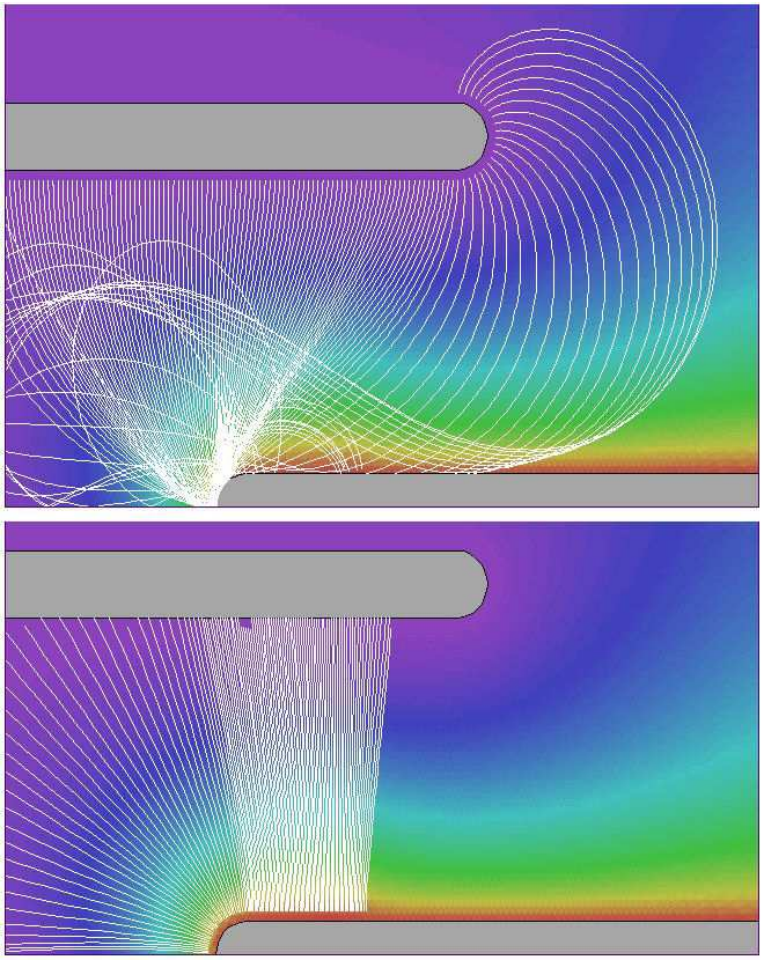

Figure 27: Electron (top) and ion orbits (bottom) for the `PINCHBEAM` example with $NSkip = 5$. Color-coding follows the electrostatic potential.

the cathode tip strike the anode about 2.5 mm from the tip. A small reduction in the cathode inner radius to 4.5 mm gives enhanced current that significantly improves the quality of the focus.

11.4 Restarting a *RelBeam* calculation

The following activities are necessary to restart a calculation in the *RelBeam* mode: 1) load the electric field file created by the *EDump* command in a previous run using the *EFile* command, 2) load values to calculate the beam-generated magnetic field from the same run using the *BBFile* command, and 3) include the *ReStart* command in the particles section.

BBFILE FileName
BBFILE PINCHBEAM.BBD
Load quantities to calculate beam-generated magnetic fields generated in a previous solution with the *BBDump* command. **Trak** issues an error message if an electric field has not been loaded or if the mesh of the BBD file does not match the electric field mesh.

You can also use the *BBFile* command in a run to track single particle orbits in the self-consistent fields of a relativistic beam. In a *Track* mode calculation, load field information using the *EFile*, *BFile* and *BBFile* commands and start particles of any species using lists or emission surfaces. **Trak** includes contributions to the magnetic field from all sources. Note that field values will not be changed by the presence of particles in the *Track* mode.

11.5 Space-charge and current neutralization of relativistic electron beams

Although **Trak** does not include detailed models of plasmas, you can investigate effects of global space-charge and current neutralization. Partial space-charge neutralization of a relativistic electron beam may occur if low-energy plasma electrons are expelled, leaving behind an excess ion density. The space-charge neutralization factor f_e is often defined as the ratio of the excess ion density to the beam density. The presence of the ion density reduces the electric field by a factor $(1 - f_e)$. For example, if $f_e = 0.9$, then the loss of plasma electrons reduces the average fields in the beam volume by a factor of 10.0.

To apply local space-charge neutralization, define a dielectric region in the electric field mesh and assign a relative dielectric constant $\epsilon_r = 1/(1 - f_e)$ in the **EStat** solution. The dielectric region must be well separated from acceleration gaps so that it does not affect the applied fields. In the subsequent **Trak** solution, contributions to electric fields from the beam space charge will be reduced by $(1 - f_e)$. Be sure to include a command of the form

```
VACUUM(E) = NReg
```

in the *Particles* section of the **Trak** file. Here $NReg$ is the region number of the dielectric. This command over-rides the default assignment of the *Material* attribute to all regions in the electric field mesh that have $\epsilon_r \neq 1.0$.

Partial current neutralization may occur if a flow of plasma electrons is driven by the inductive fields of a pulsed beam. The current neutralization factor f_m is the ratio of the

average plasma current in the propagation volume to the beam current. The effect is to reduce the beam-generated magnetic fields by a factor $(1 - f_m)$.

CNEUT Fm
CNEUT = 0.90
In response to this command, **Trak** multiplies all values of B_θ or B_z calculated from the beam current by the factor $(1 - f_m)$. For relativistic beam propagation in a plasma, the factor lies in the range $0 \leq f_m \leq 1.0$.

Although space-charge neutralization may be applied to selected regions through the choice of region dielectric constant, the current neutralization factor applies globally to all active nodes in the electric field mesh. Suppose your goal is to model effects of neutralization on a relativistic electron beam injected through a conducting foil into a gas cell. A possible approach would be to divide the solution into two parts. The first part addresses the vacuum injector with particle orbits terminated at the foil. The `PRT` file created by the first solution is used as input to the gas cell solution which could include both space-charge neutralization and a global current neutralization factor.

12 Field emission of electrons with space-charge effects

In the *FEmit* particle-tracking mode, **Trak** can model field emission of non-relativistic electrons. As in the *SCharge* mode, the program can automatically generate particles over marked source surfaces. The calculations include self-consistent effects of space charge. You can add additional electrons or ions using the *PList* or *PFile* commands. There are three differences from the *SCharge* mode:

- Only electrons may be created on emission surfaces.
- Because the electric field always has a non-zero value on the source surface, it is not necessary to create a generation surface. Electrons are created directly at the source facets.
- Electron current is assigned according to the Fowler-Nordheim equation rather than the Child-law algorithm.

Trak uses Fowler-Nordheim functions tabulated in A. Modinos, **Emission Spectroscopy** (Plenum Press, New York, 1984), p. 12. If the quantity E is the local electric field amplitude (including space-charge contributions) at a source facet with work function ϕ, then the current density in (A/m^2) is given by

$$j_{FE} \;=\; 1.537 \times 10^{-6}\; \frac{e^{\Gamma} E^2}{\phi\; t(\chi)^2}, \tag{38}$$

where

$$\Gamma = -6.83 \times 10^{9}\; \frac{\phi^{3/2}\; v(\chi)}{E}, \tag{39}$$

and

$$\chi = 3.79 \times 10^{-5}\; \frac{\sqrt{E}}{\phi}. \tag{40}$$

In the equations E is expressed in V/m and ϕ in eV. Table 12 lists values for the functions $v(\chi)$ and $t(\chi)$. **Trak** uses a cubic spline interpolation to find intermediate values.

The set of allowed commands is similar to that for the *SCharge* mode. The one difference is that the *Emit* command has different parameters:

Table 11: Fowler-Nordheim functions

χ	$v(\chi)$	$t(\chi)$
0.0000	1.0000	1.0000
0.0500	0.9948	1.0011
0.1000	0.9817	1.0036
0.1500	0.9622	1.0070
0.2000	0.9370	1.0111
0.2500	0.9068	1.0157
0.3000	0.8718	1.0207
0.3500	0.8323	1.0262
0.4000	0.7888	1.0319
0.4500	0.7413	1.0378
0.5000	0.6900	1.0439
0.5500	0.6351	1.0502
0.6000	0.5768	1.0565
0.6500	0.5132	1.0631
0.7000	0.4504	1.0697
0.7500	0.3825	1.0765
0.8000	0.3117	1.0832
0.8500	0.2379	1.0900
0.9000	0.1613	1.0969
0.9500	0.0820	1.1037
1.0000	0.0000	1.1107

EMIT WorkFunc [NPerSeg Beta]
EMIT(4) (3.56, 2, 1000.0)
This command identifies region number *RegNo* (integer) as a source surface and sets associated emission properties. **Trak** issues an error message if the region does not define a valid source surface (*i.e.*, sets of nodes on the edges of facets between *Material* and *Vacuum* elements). At least one *Emit* command is required under the *FEmit* tracking mode. Up to twenty *Emit* commands may appear in the *Particles* section. Because only electrons are allowed, it is not necessary to specify mass and charge of emitted particles. The real-number parameter *WorkFunc* is the work function ϕ in eV. The integer parameter *NPerSeg* governs how many model electrons are created per surface facet. The optional parameter β is a field enhancement factor that may be useful to simulate emission from carbon nanotube assemblies. If E_{loc} is the local electric field at the facet, then the program uses the quantity $E = \beta E_{loc}$ in Eqs. 38 through 40.

SUPPRESS SVal1 SVal2 SVal3 ...
SUPPRESS 0.20 0.30 0.40 0.60 0.80 1.00
Space-charge effects are usually small in field emission problems so the role of the suppression factors discussed in Sect. 10.3 is not as critical. The default values in the *FEmit* mode are

```
NCycle  Supression value
=======================
   1            0.500
   2            0.750
   3            1.000
   4            1.000
   ...
```

Note that the number of iteration cycles should be in the range $NCycle \geq 3$ for the default values of suppression factors.

The remaining commands allowed in the *Particles FEmit* section are identical to those in the *Particles Track* mode:

DT Dt
DT DtRef MASS
TMAX TMax
START(NReg) XStart YStart
NTRACKMAX NTrackMax
DMAX DMax
NSEARCH [E,B] NSearch
INTERP [LIN,LSQ]
MATERIAL [E,B] RegNo
VACUUM [E,B] RegNo
SECONDARY [E,B] RegNo DeltaMax0 EngMax0
STOP [UP,DOWN] [X,Y,Z,R] = Position
DIAG [UP,DOWN] [X,Y,Z,R] = Position
REFLECT [UP,DOWN] [X,Y,Z,R] = Position
LISTON [NStep] [P,E,B]
PLOTOFF

13 Extraction of high-current ion beams from a plasma

When modeling electron extraction from a thermionic or field emission cathode, we can assume that the profile of the source surface is specified a priori. Extraction of ions from a plasma is a more difficult challenge because the surface shape that satisfies all physical constraints is not known in advance. In the *Plasma* mode, **Trak** performs the Child-law emission calculations of the *SCharge* mode described in Sect.10.3. In addition, the program automatically flexes the plasma surface to ensure a uniform distribution of current-density. The calculation is complex, so it is important that you understand the physical basis of ion extraction from a plasma and the goals of the calculation. Section 13.1 discusses this topic and summarizes numerical methods applied in **Trak**. Section 13.2 reviews new commands that appear in the *Plasma* mode, while Section 13.3 discusses benchmark examples.

13.1 Ion extraction from a free plasma surface

This section briefly reviews the physical basis of the *Plasma* mode in **Trak**. For a complete discussion of Child-law emission, the Bohm current density and the formation of a plasma meniscus, see S. Humphries, **Charged-particle Beams** (Wiley, New York, 1990), Chap. 7. This book is included in the Trak package.

To begin, we must define the term *free plasma extraction surface*. Assume that plasma ions generated by a source expand in a field-free region through an aperture into an acceleration region with an applied electric field (Fig. 29). The position of the extraction surface (the transition between the plasma environment and the vacuum flow region) is determined by a balance between the plasma ion flux (which we shall denote by an effective ion current density j_p) and the extraction current density governed by the Child law, j_C. In the one-dimensional geometry of Fig. 28, the current density limit for ions with charge-state Z_i and mass m_i in a gap of width d and applied voltage V_0 is given by the expression:

$$j_C = \frac{4\epsilon_0}{9}\sqrt{\frac{2Z_i e}{m_i}}\,\frac{V_0^{3/2}}{d^2}. \tag{41}$$

If the ion flux exceeds the vacuum current limit ($j_p > j_C$), then the plasma expands to decrease the gap width d until flux balance is achieved: $j_p = j_C$. In this case, the surface in Fig. 28 moves to the right. Conversely, if $j_p < j_C$, the surface recedes to the left. In most plasma sources for ion generation, the ion temperature (T_i) is much smaller than the electron temperature (T_e). In this case, the available effective ion current density is given by the Bohm expression:

$$j_p \cong 0.6\ eZ_i n_i\sqrt{kT_e/m_i}. \tag{42}$$

The **Trak** model treats a homogeneous plasma (*i.e.*, j_p has a uniform value over the extraction surface). Therefore, a *Plasma* mode simulation has the following primary goal: for given applied voltages and electrode geometry, find a shape of the extraction surface that guarantees a uniform value of j_C. A critical issue is the nature of the extraction surface at the *triple*

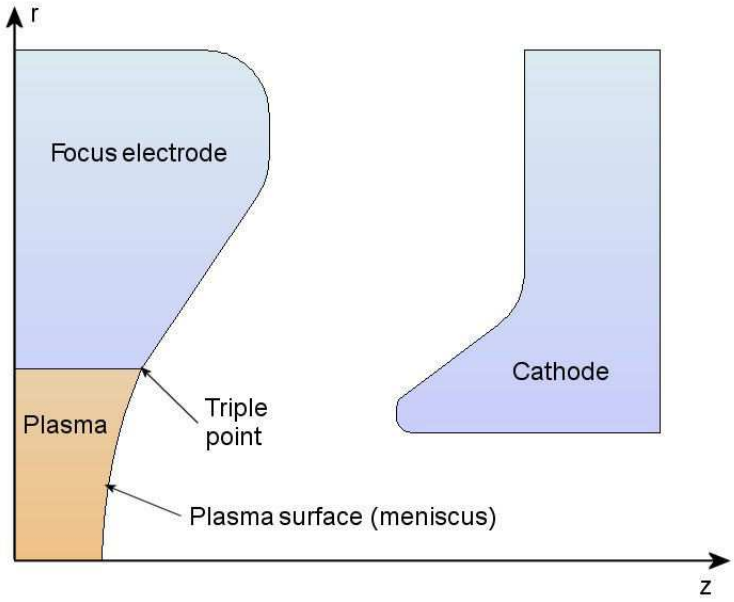

Figure 28: Terminology for ion extraction from a free plasma surface.

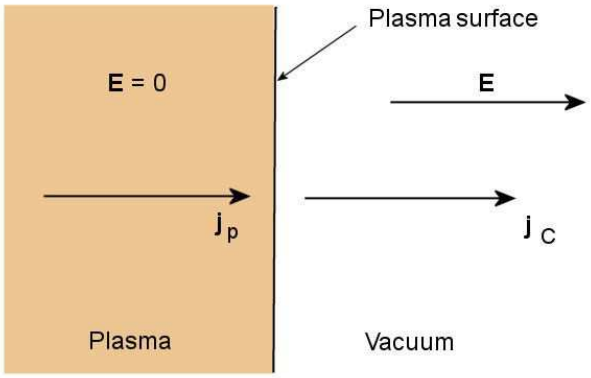

Figure 29: One-dimensional extraction from a free plasma surface.

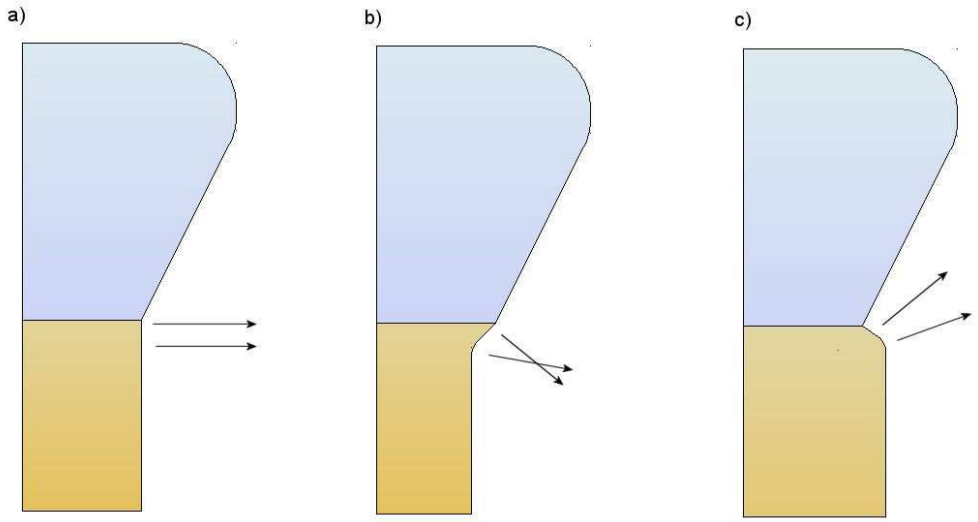

Figure 30: Balancing plasma flux j_p with space-charge-limited flux j_C to achieve a smooth connection at the triple point. *a*) Ideal balance. *b*) Insufficient plasma flux. *c*) Excess plasma flux.

point: the intersection of the plasma, the aperture electrode and the vacuum acceleration region (Fig. 28). Figure 30*a* shows the ideal situation where the values of j_p and j_C at the triple point allow a smooth connection between the extraction surface and the focusing electrodes. A reduction in j_p from the optimum value causes the plasma to recede into the aperture. In this case the focusing electrode acts as an electrostatic shield, giving a large reduction in the field amplitude at the plasma edge. The implication is that the plasma edge is effectively tied to the aperture at the triple point. A low value of j_p gives a distorted extraction surface (Fig. 30*b*) with attendant poor beam optics. Conversely, a value of j_p above the optimal value causes the plasma to bulge into the extraction gap resulting in a divergent beam (Fig. 30*c*). In summary, there are two constraints on an acceptable plasma extraction surface:

- The value of j_C must be uniform.
- The plasma surface must make a smooth connection at the triple point.

The combination of the two conditions with specified electrodes and applied voltages defines a unique plasma surface shape and Bohm current density.

Trak follows a multi-step procedure to determine self-consistent plasma surfaces. Although the operations are complex, most of them are performed automatically so that the user need not be concerned with details. The benchmark examples of Sect. 13.3 show that it is relatively easy to set up a Plasma mode calculation.

1. The user creates an electric-field model of the gun geometry in **Mesh**. The mesh must include a filled (volumetric) region to represent the plasma (Fig. 28). The plasma region must have enough depth (*i.e.*, number of elements parallel to the displacement direction) to accommodate flexing of the surface. The user makes an initial guess of the extraction surface shape (such as a flat plane). The initial surface should connect smoothly to the aperture at the triple point. The plasma region must appear in the **Mesh** script

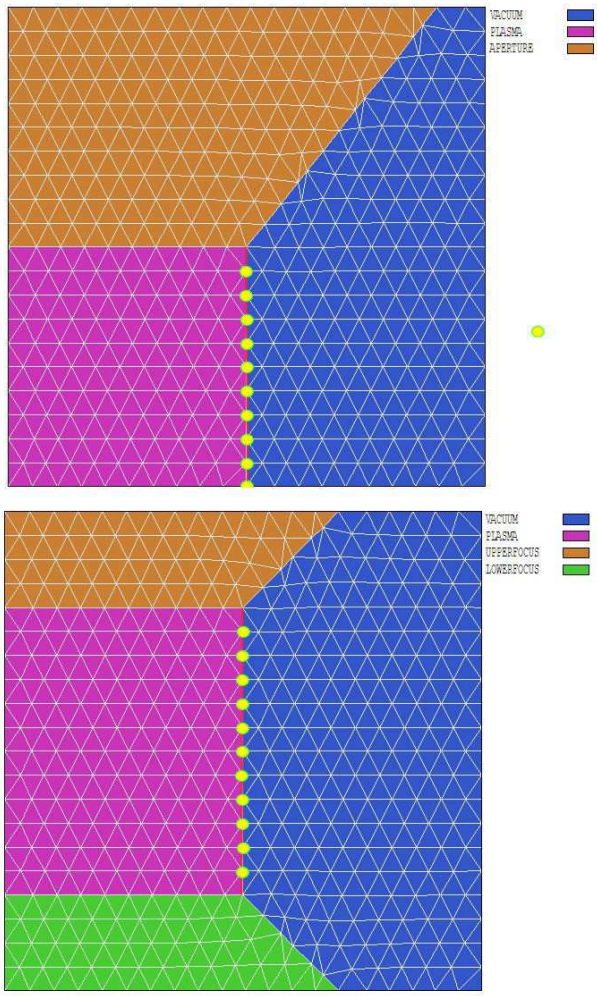

Figure 31: Definition of plasma facets (red line) and plasma nodes (yellow dots). Plasma elements are colored violet and vacuum elements are colored blue. Top: Disk plasma with symmetry axis on the lower edge. Bottom: Annular plasma.

before the surrounding aperture region. This ordering ensures that nodes on the shared plasma-aperture surface are associated with the aperture.

2. The user generates an initial electrostatic solution with **EStat**. The plasma and aperture regions should be assigned the fixed-potential condition with the same value of applied voltage.

3. In **Trak**, the plasma region is identified by an *Emit* statement. The program initially sweeps through the mesh to identify facets and nodes of the source surface. Source facets lie between an element with the region number of the plasma and a *Vacuum* element. Plasma surface nodes lie along the source surface. Figure 31 shows the two possible options for assignment of surface nodes: *a*) a cylindrical or planar gun where the surface has an axis of symmetry and *b*) an annular gun.

4. **Trak** orders the facets to form a connected set with respect to distance from the axis or a specified start point. The program determines a set of unit vectors normal to the facets

that point out of the plasma region. The vectors are used to construct an emission surface at a distance *DEmit* from the source surface to carry out the Child-law calculation.

5. **Trak** performs a calculation of space-charge-limited flow from the initial surface by creating one or more model ions on each facet of the emission surface. The program runs *NInit* cycles to achieve a stable, converged solution using the same procedure as in the *SCharge* mode.

6. **Trak** calculates the current density on source facets by averaging and back-projecting the model particle currents at the emission surface. The variation of current density is smoothed to prevent the unstable growth of ripples in the surface during the adjustment procedure.

7. The value of emitted current density at each surface node (j_n) is computed by averaging the values on adjacent facets. In addition, a node unit vector (pointing out of the plasma) is computed by averaging the vectors of adjacent facets. The position of each node is moved along this vector a distance $\Delta_n = (\alpha d/2)(1 - j_n/j_{out})$ The relationship follows from the scaling of Eq. 41. The quantity d is the width of the acceleration gap, α is a safety factor to ensure stability of the calculation, and j_{out} is the current density on the facet adjacent to the triple point. Normalizing the displacement to j_{out} ensures that the plasma surface intersects the triple point at a small angle. The direction of the shift is away from the acceleration gap (into the plasma) when $j_n > j_{out}$. In this case the shift gives a local reduction in the space-charge-limited current density.

8. The positions of plasma and vacuum nodes near the plasma surface are relaxed to preserve element integrity.

9. New normal vectors are computed for the displaced facets and the emission surface is reconstructed.

10. **Trak** performs *NCorrect* cycles of orbit and electric field recalculation to determine current densities for space-charge-limited flow from the new plasma surface. The program then returns to Step 6 to recalculate the plasma surface based on updated values of current density. The cycle of surface regeneration and solution correction is carried out *NSurface* times to achieve a convergent solution.

It is important to recognize cases where the **Trak** model is invalid. The program cannot be applied to systems where there is a strong transverse magnetic field at the extraction surface (*i.e.*, magnetically-insulated ion diodes, transverse extraction from a Penning source,...). In this case, the condition $j_p = j_C$ may not hold. Nonetheless, **Trak** can give useful information on plasma extraction from sources with applied magnetic fields (*e.g.*, duoplasmatron,...) as long as the source has sufficient shielding to isolate the magnetic field from the high-voltage extraction gap. The model also does not apply to sources that produce a mixture of ion species or charge states. In this case the behavior of the extraction sheath is more complex than the simple Child-law model. For example, ions with high-charge state may be preferentially extracted. Finally, we should note that the *Plasma* mode may also be useful in the design of electron guns with thermionic cathodes. Through the iterative procedure, you can determine cathode shapes that will guarantee uniform current density of the extracted beam.

13.2 Plasma mode commands

The following commands perform the same function as those described in Chap.10 for the *SCharge* mode:

AVG = 0.20
MAXCYCLE = 2500
RESTARGET = 2.0E-8
OMEGA = 1.95
START(5) = 0.25, 0.00
RESTART

Note the *NCycle* command is not valid in the *Plasma* mode. **Trak** automatically determines *NCycle* from the parameters supplied to control the plasma surface calculation. Note that you can add list particles with the commands:

PLIST
PFILE

The *Emit* command has a different format in the *Plasma* mode.

EMIT(NReg) Mass Charge DEmit NPerSeg DGap [Ts]
EMIT(4) = (4.0, 1.0, 0.05, 3, 2.50)
This command identifies the filled region *NReg* in the electric-field mesh as a plasma. The plasma surface is defined as the boundary of the region adjacent to a *Vacuum* region. The plasma must be a fixed-potential region. Specifications for the following five parameters are required. The parameters *Mass* and *Charge* have the same meaning as in the *Emit* command of the *Track* and *SCharge* modes. Enter the mass in AMU (atomic mass unit). **Trak** inserts the value for an electron if $Mass = 0.0$. Enter the charge of the emitted particles in units of e (*i.e.*, -1.0 for electrons and +1.0 for protons). The quantity $DEmit$ is the distance from the source surface to the emission surface. Enter the value in units set by the present value of $DUnit$. As a rule of thumb, set $DEmit$ equal to about 1.5-2.0 times the width of elements adjacent to the plasma surface. Large values of $DEmit$ may prevent convergence of the plasma surface search. The integer parameter $NPerSeg$ is the number of model particles created per facet of the plasma surface. The real-number parameter $DGap$ is the approximate width of the acceleration gap. Enter the value in units set by $DUnit$. Larger values of $DGap$ may speed the calculation. Reduce $DGap$ if the calculation does not converge. The signs of convergence failure are: 1) a wavy of irregular plasma surface, 2) code termination because of distorted elements. The optional parameter Ts (in eV) equals the tempreature of the plasma to calculate the angular divergence of particles at the emission surface. **Trak** can handle only a single plasma region in the *Plasma* mode. Therefore, the *Particles* section may contain only one *Emit* command.

The following command controls the plasma surface calculation.

PLASMAPARAM NSurface NCorrect NInit [NRadius]
PLASMAPARAM 4 5 12

The *PlasmaParam* command has four integer parameters, one of them optional. The quantity $NSurface$ is the number of cycles of plasma surface adjustment. The quantity $NCorrect$ is the number of orbit-field recalculations to determine a stable solution for space-charge-limited flow per surface adjustment. $NInit$ is the number of initial orbit-field calculations before the first surface adjustment. The optional parameter $NRadius$ controls the width of the region near the surface over which mesh smoothing is applied to plasma and vacuum nodes. (Default values: $NSurface = 3$, $NCorrect = 5$, $NInit = 12$, $NRadius = 4$).

In the remainder of this section, we shall discuss how to choose values for the quantities in the *PlasmaParam* command and how to ensure solution convergence. To begin, the number of required surface adjustments (controlled by $NSurface$) depends on how much the initial surface must be flexed to achieve uniform current density. The solution will converge faster if you make an improved initial guess of the surface shape based on a previous run. **Trak** records extensive information on the surface adjustment procedure in the listing file `RUNNAME.TLS`. The table labeled *Adjustment of plasma surface nodes* lists the smoothed values of j_n used to calculate the displacements Δ_n as well as the initial and final values of the plasma surface node coordinates. **Trak** also records the root-mean-squared node displacement. This quantity will be small for a convergent solution. Other indications of convergence are: 1) values of j_n are almost equal and 2) the total emitted current approaches a fixed value.

Valid position adjustments Δ_n require accurate values for the smoothed, space-charge-limited current density. Therefore, **Trak** must perform several orbit-field recalculations to update the emitted current density for the new source/emission surface positions. The parameter $NCorrect$ controls the number of orbit-field recalculations per surface adjustment. The value should be large enough to ensure that total emitted current approaches a stable value between surface adjustments. The parameter $NInit$ equals the number of orbit-field recalculations before the first surface adjustment. The number must be sufficient to achieve a converged solution for the initial surface geometry. You can reduce $NInit$ by loading a previously-computed solution for space-charge-limited flow from the initial surface and by using the *Restart* command.

Finally, the parameter $NRadius$ controls the half-width of the region near the plasma surface over which mesh smoothing occurs. If a plasma node has indices (K_0, L_0), then graded smoothing is applied to nodes in the range $K_0 - NRadius \le K \le K_0 + NRadius$, $L_0 - NRadius \le L \le L_0 + NRadius$. The value of $NRadius$ must be larger than the number of elements over which the surface is displaced from the initial guess. Increase $NRadius$ if **Trak** reports an element distortion in response to a surface shift. You can reduce the chances for element distortion by making a better guess of the initial surface.

13.3 Application examples

This section describes some simple examples that illustrate the physics of ion extraction from a plasma as well as **Trak** modeling techniques. Figure 32 shows the geometry of a benchmark calculation for a cylindrical proton gun defined by the files `PLASMA01.MIN`, `PLASMA01.EIN` and `PLASMA01.TIN`. The aperture has a radius of 2.0 cm, the extraction gap has width $d = 4.0$ cm and the applied voltage is $V_0 = 50.0$ kV. The focusing electrode is inclined at an angle

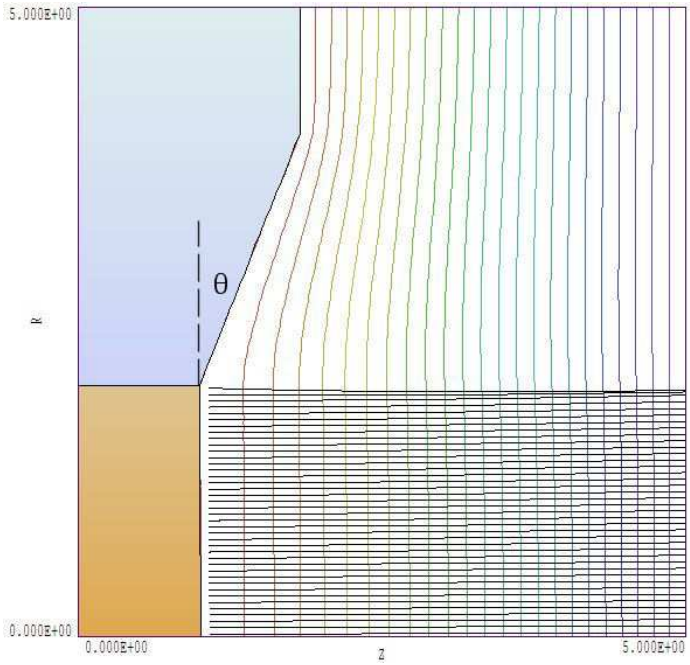

Figure 32: Geometry of the example PLASMA01 with focusing electrode angle $\theta = 22.5^{\circ}$. Figure shows the self-consistent plasma surface, equipotential contours and model particle orbits. Dimensions in cm.

$\theta = 22.5^{\circ}$ and the cathode is a plane on the right-hand side of the solution volume. With these parameters we expect that the solution will approximate an ideal Pierce gun. Therefore the ion current density j_C should be almost uniform over a flat plasma surface that intersects the triple point. The current-density magnitude is given approximately by Eq. 41: $j_C = 380.5$ A/m^2. For comparison, the **Trak** calculation in the *Plasma* mode gives a surface with a convexity of 0.009 cm on axis and average current density $\overline{j_C} = 402.6$ A/m^2. In the final state, the current density is uniform over the surface to within $\pm 0.06\%$. The calculation illustrates the stability and convergence of the *Plasma* mode procedure.

Next, suppose we increase the focus-electrode angle to 30.0° (example `PLASMA02`). The extra metal reduces the electric field on the outer edge of the aperture, suppressing the space-charge-limited current density. Therefore, emission is non-uniform over a flat surface that intersects the triple point (Fig. 33). In this case, **Trak** must flex the plasma to create a concave surface, thereby reducing the current density near the axis. Table 12 shows the **Trak** input script. The *Emit* command specifies proton generation using an emission surface a distance $DEmit = 0.075$ cm from the source surface. For comparison, the local element width along z is 0.050 cm. Two model particles are created per facet and the gap width is $DGap = 4.0$ cm (Fig. 32). The *PlasmaParam* command specifies 4 cycles of surface adjustment with 5 orbit-field recalculations for each cycle. There are 12 initial orbit-field recalculations to achieve an accurate starting solution.

The left-hand-side of Fig. 34 shows the final concave shape of the plasma surface along with computed equipotential lines and model particle orbits in the converging beam. The right-hand side shows the solution for a shallow focusing-electrode angle ($\theta = 15.0^{\circ}$). In this case, the electric field magnitude is higher at the outer radius so that the plasma must assume a

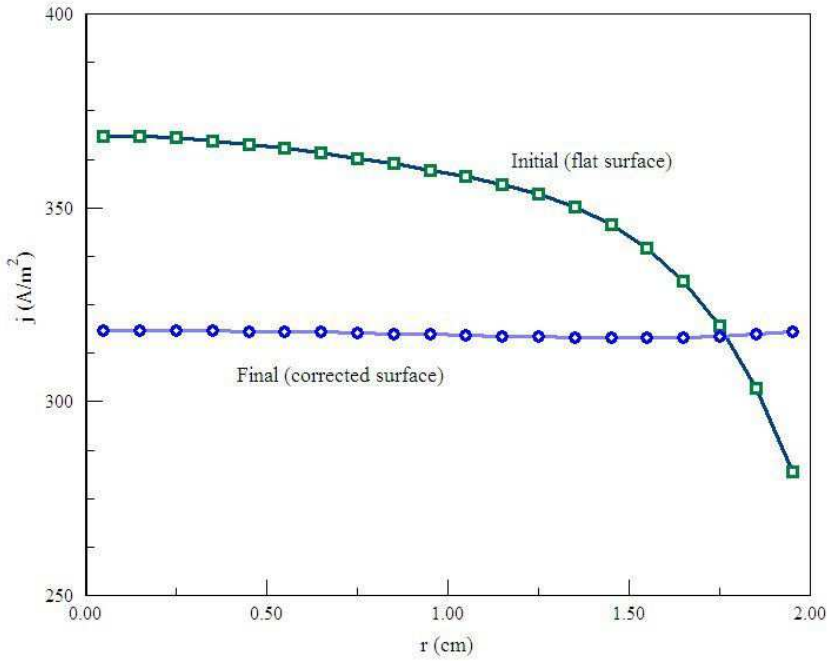

Figure 33: Radial variation of current density for example PLASMA02 with $\theta = 30.0^{\circ}$. Green curve shows the initial state while the blue curve shows the current density after the iterative plasma surface adjustment.

Table 12: Trak input file PLASMA02.TIN

```
* FILE: Plasma02.TIN
FIELDS
    EFILE: Plasma02.EOU
    DUNIT: 100.0
    END
PARTICLES Plasma
   EMIT(2) 1.0 1.0 0.075 2 4.0
   PLASMAPARAM 4 5 12
   AVG 0.30
END
DIAGNOSTICS
   JSOURCE
   PARTLIST
   EDUMP Plasma02P
END
ENDFILE
```

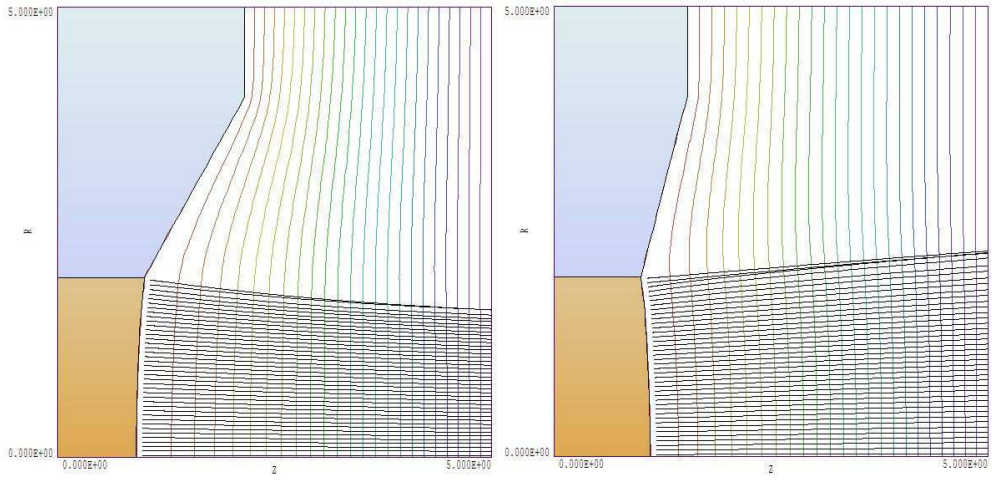

Figure 34: Self-consistent plasma surface shape, equipotential lines and model particle orbits. Left: example `PLASMA02` with $\theta = 30.0^{\circ}$. Right: example `PLASMA03` with $\theta = 15.0^{\circ}$.

convex shape for uniform current density.

The examples suggest the following logical sequence for the design of a practical cylindrical ion gun:

1. Starting from the Pierce solution ($\theta = 22.5^{\circ}$) with flat emission surface, we add an exit aperture in the cathode. The effect of the aperture is to reduce the electric-field magnitude on the plasma surface near the axis.

2. To compensate for the reduction in on-axis current density, we increase the angle of the focusing electrode to suppress emission near the outer radius of the plasma surface.

3. A further increase in the focusing electrode angle would give a converging beam. In this case, it is possible to reduce the radius of the exit aperture (Fig. 28).

14 Modeling secondary emission of electrons

14.1 Models for secondary electron emission

Trak can represent electron secondary-emission processes to model electro-optical devices and collectors for high-power vacuum tubes. These capabilities apply only to electrons and can be used in the *Track*, *SCharge*, *RelBeam* and *FEmit* modes. In the latter modes the space-charge of emitted electrons is added to the electric field recalculation. A single model electron orbit can represent a multi-generational set of electrons. Regions with elements corresponding to secondary emitters are defined with the *Secondary* command. When an electron enters a secondary element, the secondary-emission coefficient δ is calculated from the incident energy and particle angle relative to the surface at the entrance point. The current of the electron is multiplied by δ and it is restarted at a point in the vacuum space near the surface entrance point. The kinetic energy of the emitted electron follows a Maxwell distribution with $T_e = 2.0$ eV. The direction of emission is normal to the electrode surface.

Trak determines the secondary emission coefficient from a parametric model based on work by Jonker [J.L.H. Jonker, Phillips Research Reports **6**, 372 (1951), Phillips Research Reports **7**, 1 (1952), Phillips Research Reports **12**, 249 (1957)] and Vaughn [R.M. Vaughn, IEEE Trans. Electron Devices **ED-36**, 1963 (1989) and IEEE. Trans. Electron Devices **ED-40**, 830 (1993)Ṫhe model involves the angle α between the direction of the incident electron and a vector normal to the surface. The defining function for the angular dependence of δ is

$$F(\alpha) = \frac{1}{\sqrt{cos(\alpha)}}. \tag{43}$$

The maximum value of secondary coefficient and the corresponding incident electron energy are given by

$$\delta_m = \delta_{mo}\ F(\alpha), \tag{44}$$
$$E_m = E_{mo}\ F(\alpha), \tag{45}$$

where δ_{mo} and E_{mo} are the values at normal incidence. These quantities are tabulated for a variety of materials in Tables 13 and 14. The secondary emission coefficient as a function of the angle α and kinetic energy E of the incident electron is given approximately as

$$\delta(\alpha, E) \cong \delta_m(\alpha)\ \left[f \exp^{(1-f)}\right]^a, \tag{46}$$

where $f = E/E_m(\alpha)$. The parameter a has the value 0.62 for $f < 1$ and 0.25 for $f \geq 1$. To prevent infinite values of δ, the code takes $\alpha = 80.0^o$for orbits with incident angles that exceed 80.0^o. The reference S. Humphries, N. Dione and J. Petillo, *Secondary-electron emission modeling on a conformal mesh*, Proc. WorkShop on RF Superconductivity, Santa Fe, 1999 (included with the **Trak** package) gives a detailed description of numerical methods used to treat secondary emission in the **Trak** code.

Table 13: Secondary emission coefficients for elements, adapted from D.R. Lide, ed., **Handbook of Chemistry and Physics**, 74th Edition (CRC Press, Boca Raton, 1993), 12-107

Element	σ_{mo}	E_{mo} (eV)	Element	σ_{mo}	E_{mo} (eV)
Ag	1.5	800	Li	0.5	85
Al	1.0	300	Mg	0.95	300
Au	1.4	800	Mo	1.25	375
B	1.2	150	Na	0.82	300
Ba	0.8	400	Nb	1.2	375
Bi	1.2	550	Ni	1.3	550
Be	0.5	200	Pb	1.1	500
C (diamond)	2.8	750	Pd	> 1.3	> 250
C (graphite)	1.0	300	Pt	1.8	700
C (soot)	0.45	500	Rb	0.9	350
Cd	1.1	450	Sb	1.3	600
Co	1.2	600	Si	1.1	250
Cs	0.7	400	Sn	1.35	500
Cu	1.3	600	Ta	1.3	600
Fe	1.3	400	Th	1.1	800
Ga	1.55	500	Ti	0.9	280
Ge	1.15	500	Tl	1.7	650
Hg	13	600	W	1.4	650
K	0.7	200	Zr	1.1	350

Table 14: Secondary emission coefficients for compounds, adapted from D.R. Lide, ed., **Handbook of Chemistry and Physics**, 74th Edition (CRC Press, Boca Raton, 1993), 12-107

Material	σ_{mo}	E_{mo} (eV)	Material	σ_{mo}	E_{mo} (eV)
Alkali halides			**Oxides**		
CsCl	6.5		Ag2O	1.0	
KBr (crystal)	14	1800	A12O3 (layer)	2-9	
KCl (crystal)	12	1600	Ba0 (layer)	2.3-4.8	400
KCl (layer)	7.5	1200	Be0	3.4	2000
KI (crystal)	10	1600	Ca0	2.2	500
KI (layer)	5.6		Cu20	1.2	400
LiF (crystal)	8.5		Mg0 (crystal)	20-25	1500
LiF (layer)	5.6	700	Mg0 (layer)	3-15	400-1500
NaBr (crystal)	24	1800	MoO2	1.2	
NaBr (layer)	6.3		SiO2 (quartz)	2.1-4	400
NaCl (crystal)	14	1200	SnO2	3.2	640
NaCl (layer)	6.8	600	**Others**		
NaF (crystal)	14	1200	BaF2 (layer)	4.5	
NaF (layer)	5.7		CaF2 (layer)	3.2	
NaI (crystal)	19	1300	BiCs3	6	1000
NaI (layer)	5.5		BiCs	1.9	1000
RbCl (layer)	5.8		GeCs	7	700
Sulfides			Rb3Sb	7.1	450
MoS2	1.1		SbCs3	6	700
PbS	1.2	500	Mica	2.4	350
WS2	1.0		Glasses	2-3	300-450
ZnS	1.8	350			

14.2 Control commands

The following commands may appear in the *Track*, *SCharge*, *RelBeam* and *FEmit* sections.

SECONDARY [E,B] NReg DeltaMax0 EngMax0
SECONDARY (E,5): 2.45 320.0
This command designates that region $NReg$ in the electric or magnetic field mesh is a *Secondary* type and assigns emission parameters. In contrast to the simple *Vacuum* and *Material* specifications, the command requires two parameters. The real-number quantity $DeltaMax0$ is the maximum value of the secondary emission coefficient for normal incidence. The real quantity $EngMax0$ is the kinetic energy of the incident electron (in eV) at which the maximum occurs.

SECONDPARAM KECutOff MinFact
SECONDPARAM 1.0 2.0E-3
This command sets global parameters that control the termination of multi-generation electron orbits to prevent infinite calculations. The quantity $KECutOff$ is a cutoff value (in eV) for kinetic energy. An orbit terminates if the energy of the incident electron falls below this value. The default is $KECutOff = 2.5$ eV. During a multi-generation orbit calculation **Trak** maintains a quantity $MultFact$ equal to the effective number of electrons in a generation per incident electron. This quantity equals the product of secondary emission coefficients for all collisions with secondary materials, $MultFact(N) = \delta_1 \times \delta_2 \times ...\delta_N$. An orbit is terminated if the multiplication factor drops below the value $MinFact$. The default is $MinFact = 1.0 \times 10^{-4}$.

SECONDLIST In response to this command **Trak** writes a detailed record of secondary emission events during an orbit integral to the listing file. In runs with $NCycle > 1$, a record is made only on the final cycle.

Finally, additional information on particle multiplication is contained in data written in response to the *RegList* command in the *Diagnostics* section.

15 Particle and field diagnostics

Trak records information on the calculation in the listing file `FPrefix.TLS`. After computing particle orbits, the program can perform several diagnostic operations in response to commands in the input file. These are contained in the *Diagnostics* section that follows the *Particles* section. After the *Diagnostics* section (or if there is no section), **Trak** looks for the *EndFile* command, closes all files, and terminates operation.

15.1 General control commands

Diagnostic commands divide into five classes: 1) general control, 2) electric fields, 3) applied magnetic fields, 4) beam-generated magnetic fields and 5) field-line or particle orbit quantities. There are three general control commands:

NSCAN NScan
NSCAN = 100
Set the number of intervals for line scans initiated by the *EScan*, *BScan* or **BBScan** commands. The default value is $NScan = 50$.

INTERP [E,B,BB] [LIN,LSQ]
INTERP(BB) = LIN
The electric and magnetic field calculations in the *EPoint*, *EScan*, *BPoint*, *BScan*, *BBPoint* or *BBScan* commands may employ either the linear or least-squares-fit method. The string parameter may assume the values LIN or LSQ. The command affects subsequent field calculations. Multiple *Interp* commands may appear in the *Diagnostics* section.

DUNIT DUnit
DUNIT = 39.37
Change the unit conversion factor for the input of spatial quantities in the diagnostic commands.

15.2 Electric field diagnostics

These commands may be used to record the final state of the electric field. These calculations function only if an electric field mesh has been loaded. Note that calculations are performed on the final field solution which may include the effects of scaling, spatial shifts and beam space charge.

EDUMP FPrefix
EDUMP = KlyMod
Record a file of electric field values with the name `FPrefix.EOU` in the current directory. The file may be loaded into **EStat** or **Trak** for plotting and analysis. It may also be used as input

to a subsequent **Trak** run. For instance, you may want to generate detailed edge orbits in the self-consistent fields of an ion gun.

EPOINT = (X, Y, Z)
EPOINT = (0.0, 0.0, 12.0)
Record a computation of the electric field at the point (X, Y, Z) in the listing file using the current interpolation method. Enter the coordinates in units set by the current value of *DUnit*.

ESCAN (X1, Y1, Z1) (X2, Y2, Z2)
ESCAN (0.0, 0.0, 12.0) (5.0, 0.0, 12.0)
List electric field values along the scan line from (X_1, Y_1, Z_1) to (X_2, Y_2, Z_2). Enter the coordinates in units set by *DUnit*.

15.3 Magnetic field diagnostics

The following commands record values of applied magnetic field if a magnetic solution has been loaded with the *BFile* command. The values reflect the actions of scaling and shift operations.

BDUMP = FPrefix
BDUMP = SolShift
Record a file of the magnetic field values used in the simulation with the name `FPrefix.BOU` in the current directory. The file may be loaded into **BStat** or **Trak** for plotting and analysis. The command functions only if a solution has been loaded with the *BFile* command.

BPOINT = (X, Y, Z)
BPOINT = (3.0, 3.0, 0.0)
List the value magnetic field calculated from all sources at the point (X, Y, Z). Enter coordinates in units set by the current value of *DUnit*. Depending on the commands of the *Fields* section, the calculation may include contributions from a finite-element mesh, a table of on-axis values, uniform field components, or an azimuthal field created by a wire.

BSCAN (X1, Y1, Z1) (X2, Y2, Z2)
BSCAN (2.0, 0.0, -3.0) (2.0, 0.0, 15.0)
List magnetic field values calculated from all sources along the scan line from (X_1, Y_1, Z_1) to (X_2, Y_2, Z_2). Enter the coordinates in units set by *DUnit*.

15.4 Diagnostics of beam-generated magnetic fields

The following commands write values of the beam-generated magnetic field. They function only in *RelMode* calculations.

BBDUMP = FPrefix
BBDUMP = HTest
Write node values used for the calculation of a beam-generated magnetic field to the file `FPrefix.BBD` in the current directory. The file has a format similar to the output files of **EStat** and **PerMag**. For a cylindrical simulation the recorded quantities are B_θ (in tesla) and the enclosed current at the node (in A). The quantities in a planar simulation are B_z and the enclosed linear current in A/m. You can load the file in **Trak** to create plots of the beam-generated magnetic field.

BBPOINT = (X, Y, Z)
BBPOINT: 5.5 6.2 12.3
List the beam-generated magnetic field at the point (X, Y, Z). Enter coordinates in units set by *DUnit*.

BBSCAN = (X1, Y1, Z1) (X2, Y2, Z2)
BBSCAN = (2.0, 0.0, -5.0) (2.0, 0.0, 13.0)
Lists values of the beam-generated magnetic field along the scan line from (X_1, Y_1, Z_1) to (X_2, Y_2, Z_2). Enter the coordinates in units set by *DUnit*.

BBBOUNDARY NReg
BBBOUNDARY(5)
Write a list of facet currents along a collector surface composed of nodes with region number *NReg*. **Trak** locates the intersection of the collector surface with the axis and then moves radially outward calculating the total enclosed current and current density from the associated facet currents and areas. The routine can provide useful calculations of current density variations on target surfaces. You may want to define special collector surfaces and/or apply the *RelBeam* mode in non-relativistic calculations to utilize the capabilities of this command.

CDENS ZPos Rmax NR
CDENS XPos Ymax NY
CDENS 5.0 1.5 20
Employ information on the beam-generated magnetic field stored on the electric field mesh to generate a transverse scan of the beam current density. Sometimes, it may be useful to simulate a non-relativistic beam in the *RelBeam* in order to use this diagnostic. In cylindrical geometries, the parameters are Z_{pos} (the z position for the scan), R_{max} (the outer radius for the scan) and N_R (the number of radial intervals). The inner scan radius is always taken as the minimum radius of the solution volume (where $B_\theta = 0.0$). In planar geometries, the parameters are X_{pos} (the x position for the calculation), Y_{max} (the outer displacement for the calculation) and N_Y (the number of intervals in y). **Trak** issues an error message if $Y_{min} \leq 0.0$. The calculated values may be subject to large numerical errors if the number of model particles is small or if there are accumulated errors in the orbit calculations. Table 15 shows an example of the listing for a cylindrical simulation:

Table 15: Listing created by the *CDens* command

```
Current density derived from the beam-generated magnetic field
Axial position, X:  5.00000E+00
     r          Jz
======================
1.5000E-02  4.2624E+06
4.5000E-02  3.2274E+06
7.5000E-02  2.9988E+06
1.0500E-01  2.8971E+06
1.3500E-01  2.8448E+06
1.6500E-01  2.9201E+06
1.9500E-01  3.1195E+06
2.2500E-01  3.6695E+06
2.5500E-01  3.3201E+06
2.8500E-01  3.0915E+05
```

15.5 Orbit diagnostics

The final set of commands writes information about particle orbits (or field lines) to the listing file. The final positions used for the calculation may depend on the presence of *Diag* or *Stop* commands in the *Particles* section.

PARTLIST [SymType]
PARTLIST Rect
Make a formatted listing of initial and final positions and momenta for a particle simulation. The list contains only coordinate values in a field line simulation. The parameter *SymType* may have the values $RECT$ or $CYLIN$. In the default mode ($RECT$) components are given in Cartesian coordinates. If $SymType = CYLIN$, then values are converted to cylindrical components relative to the z axis.

PARTFILE FPrefix
PARTFILE = Part02
Write a file `FPrefix.PRT` of final orbit parameters to the current directory in the standard particle file format. This file can be used as input to a subsequent **Trak** run or for distribution analyses in **GenDist**. This command does not function in the *FLine* mode.

REGLIST
This command initiates a list of particle collisions with material volumes organized by region. Information includes the total number of hits, the weighted number of particle collisions in secondary simulations, and the total deposited current. Note that region 0 includes all other particle termination events (*e.g.*, left the solution volume, exceeded T_{max}, ...).

PARTDIST

Write a list of beam distribution properties using the final orbit parameters. Calculations are performed relative to the z axis, independent of the symmetries of the field solutions. Quantities include the root-mean-squared widths, angular divergences and emittances in the x and y directions. **Trak** makes a special emittance listing for cylindrical beams, calculating the value corresponding to a uniform azimuthal distribution of model particles. If the simulation involves space charge, the distributions are weighted by the model particle currents. You can perform similar calculations interactively in **GenDist** by creating a particle output file with the *PartFile* command.

DISPLIST

Make a simple listing of initial and final orbit displacements from the axis. In cylindrical simulations the tabulated quantities are r_i and r_f, while y_i and y_f are recorded for planar simulations. This information can be useful to assess the optical quality of a charged-particle gun. If the gun produces a laminar beam, a plot of $r_i - r_f$ follows a straight line.

JSOURCE

Record a list of smoothed current density on emission facets ordered by region. Table 16 shows an example. The smoothing routine uses a least-squares-fit to a fourth-order polynomial. The independent variable is distance from the start point of the region. Only even terms in r (or y) are used if the emission surfaces contacts an axis of symmetry. In the case of cylindrical solutions, a weighting function proportional to r is applied so that errors in the calculated current density for low-current particles near the axis do not skew the results. Figure 35 illustrates the fitted function for a cylindrical beam.

Table 16: *JSource* command listing

```
---  Smoothed emission-facet current density ---
  N   NReg    ZAvg       RAvg       JAvg
=========================================
   1     2  1.009E+00  5.000E-02  4.020E+02
   2     2  1.009E+00  1.500E-01  4.020E+02
   3     2  1.009E+00  2.500E-01  4.020E+02
   4     2  1.008E+00  3.500E-01  4.020E+02
   5     2  1.008E+00  4.500E-01  4.021E+02
   6     2  1.007E+00  5.500E-01  4.021E+02
   7     2  1.007E+00  6.500E-01  4.021E+02
   8     2  1.006E+00  7.500E-01  4.022E+02
   9     2  1.005E+00  8.500E-01  4.023E+02
  10     2  1.005E+00  9.500E-01  4.023E+02
  11     2  1.004E+00  1.050E+00  4.024E+02
  12     2  1.004E+00  1.150E+00  4.024E+02
  13     2  1.004E+00  1.250E+00  4.024E+02
  14     2  1.003E+00  1.350E+00  4.024E+02
  15     2  1.003E+00  1.450E+00  4.025E+02
  16     2  1.003E+00  1.550E+00  4.024E+02
  17     2  1.003E+00  1.650E+00  4.024E+02
  18     2  1.002E+00  1.750E+00  4.023E+02
  19     2  1.001E+00  1.850E+00  4.023E+02
  20     2  1.000E+00  1.950E+00  4.021E+02
```

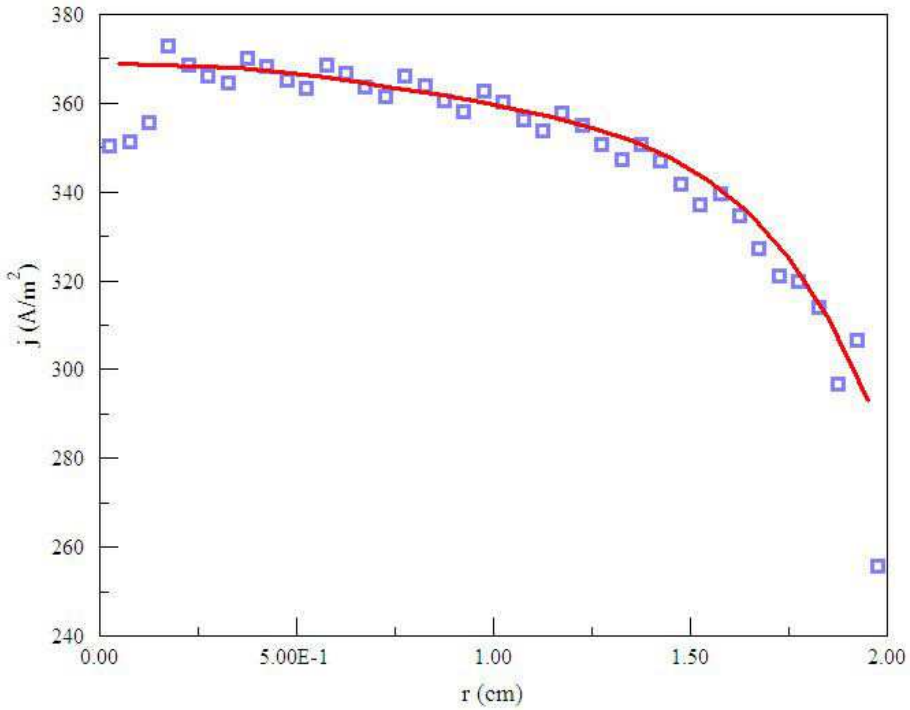

Figure 35: Weighted current-density smoothing for a cylindrical beam. Blue squares represent values for individual model particles, the red line is a fourth-order fit. Note the effects of small interpolation errors at element boundaries and near the axis.

Index