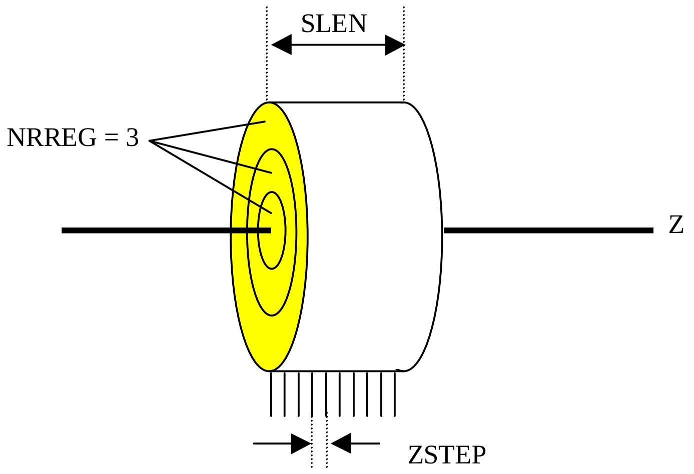
Regions and Radial Subregions

A ***region*** is a cylinder coaxial with the Z direction (in most cases the beam direction).

As a whole, it is characterized by only three user-supplied parameters:

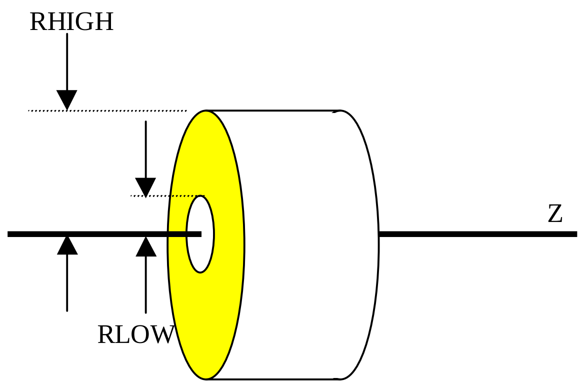
• **SLEN**	Its ***length*** along the Z axis, in meters.
• **NRREG**	The ***number of radial subregions*** within the region (1-4).
• **ZSTEP**	The ***particle simulation step size*** within the region, in meters.



Each ***region*** is made up of 1 to 4 ***radial subregions***. Each radial subregion is characterized by inner and outer radii, an electromagnetic field type, and a material type and geometry.

Three more user-supplied parameters are attached to each radial sub-region (or r-region):

• **RREGN**	The sub-region number (1-4). check this
• **RLOW**	The inner radius of the sub-region (the radius of the “hole”), in meters.
• **RHIGH**	The outer radius of the sub-region, in meters.



That defines the shape of the sub-regions, but tells us neither what kind of fields inhabit them, nor what kinds of materials are to be found in them. So for each sub-region, we also specify:

The name of the type of field (FTAG) that inhabits the sub-region. Among the possibilities:

NONE	No fields at all
ACCE	Linear accelerator fields (various kinds of RF cavities)
BACK	A background field for the cell containing this region
BSOL	Bent solenoid
COIL	Coil
DEFL	Deflection cavity
DIP	Sector dipole
FOFO	Solenoidal FOFO lattice element
QUAD	Quadrupole field
ROD	Axial current carrying rod
SEX	Sextupole field
SHEE	Sum of fields from annular current sheets
SOL	Solenoid
STUS	User-specified static magnetic field, tabulated on a grid
TROD	Tapered current carrying rod

Following this field type specifier is a list of 15 parameters (FPARM). What those parameters mean depends on the field type. Some types of field use very few parameters, or none. Others use all 15. All unused parameters should be set to zero; they cannot just be omitted.

Most of the field types listed above have several subtypes or ***models***. A ***model number*** is specified as the first parameter, and the meaning of the remaining parameters will usually depend on which model you are using. All the gory details are described in the ICOOL User's Manual, pp. 31-50.

Now all that remains is to specify the material type and material geometry inhabiting the region. That's done in about the same manner just employed for fields. One line specifies a material type (MTAG); among the choices:

VAC	vaccuum
LH	liquid hydrogen
LI, BE, B, C, AL, TI, FE, CU, W, HG	various solid elements
LIH	lithium hydride
FURB	cat furballs

Finally comes the material geometry (MGEOM); among the choices are:

• NONE	used for vaccuum
• CBLO	Simple cylindrical block
• WEDG	Asymmetric wedge
• PWED	Asymmetric polynomial wedge
• ASPW	Azimuthally symmetric wedge

So what does a complete ***Region Definition Block*** look like? Something like:

one line specifying SLEN, NRREG, and ZSTEP, the Region Descriptors
for each of NRREG radial sub-regions . . .

> one line specifying RREGN, RLOW and RHIGH), the Subregion Descriptors
> one line holding a Field Type specifier (FOFO, QUAD, ROD etc.)
> one line (or a couple of lines) specifying the 15 Field Type Parameters
> one line holding a Material Type specifier (VAC, LH, BE etc.)
> one line holding a Material Geometry specifier (CBLO, WEDG etc.)
> one line (or a couple of lines) specifying the 10 Material Geometry Parameters.

In the illustrations below, this entire region definition block will be drawn as:

$$\boxed{\text{Region Definition Block}}$$

Lists of Regions and Subregions

Though we eagerly await the design that will perform the whole muon production and cooling task in a single region, in the meantime we are obliged to building more complex devices employing a whole bunch of regions strung out along the beam direction (increasing Z). You can follow one region definition block by another, and another.... Remember that you specified the region's length in Z, so ICOOL puts the left edge of region 2 at the right edge of region 1, and so on. If you need to put "empty space" between the active regions – those that bend or focus or accelerate or absorb or whatever – just put in a "drift space" of the proper length – a region with 1 radial subregion, field=NONE, material type=VAC, and material geometry=NONE.

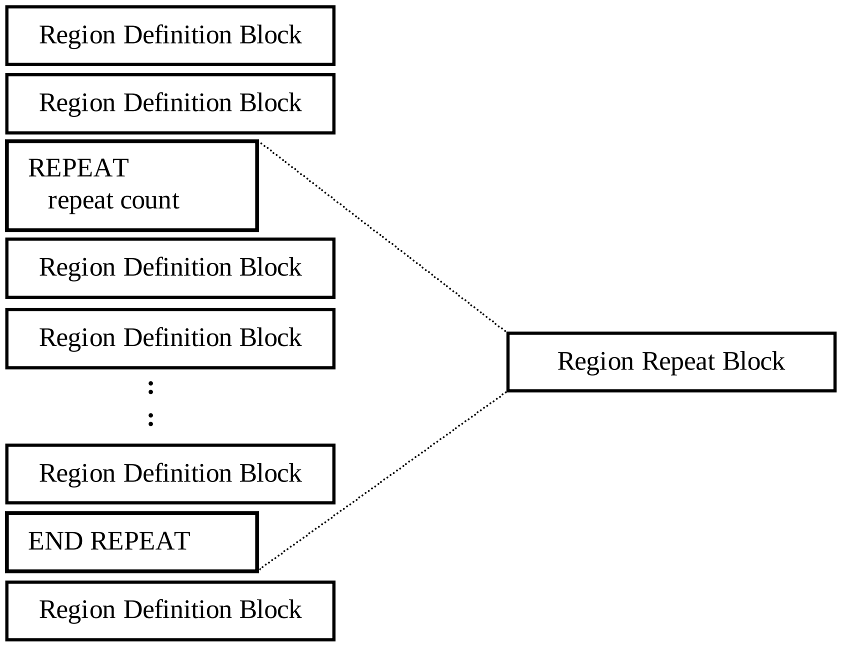
In the drawings to follow, a list of region definition blocks – a ***Region Definition List*** is drawn:

$$\boxed{\text{Region Definition List}}$$

Though it is quite possible to design a useful apparatus whose geometry can be described by a simple one-time-through list of regions and subregions, most muon cooling apparatus is far more complex. However, as we delve into more complex region definitions, is to well to keep in mind that before particles are propagated through the regions, all apparatus specifications are converted internally to exactly this kind of one-time-through region definition list. The conversion from the more compact human-readable form of region description to the simpler but lengthy expanded form used by ICOOL is done (transparently to the user) during the setup of the simulation.

Repeating Yourself, Part 1: The *Repeat* Statement

Any contiguous group of region definition blocks may be placed inside a ***repeat / end repeat block***. Schematically:



The rules for a ***region repeat block*** are very simple:

Any number of contiguous region definition blocks can be included inside a repeat block.

The ***repeat*** statement is followed by a line which specifies the number of times the repeat block is to be repeated.

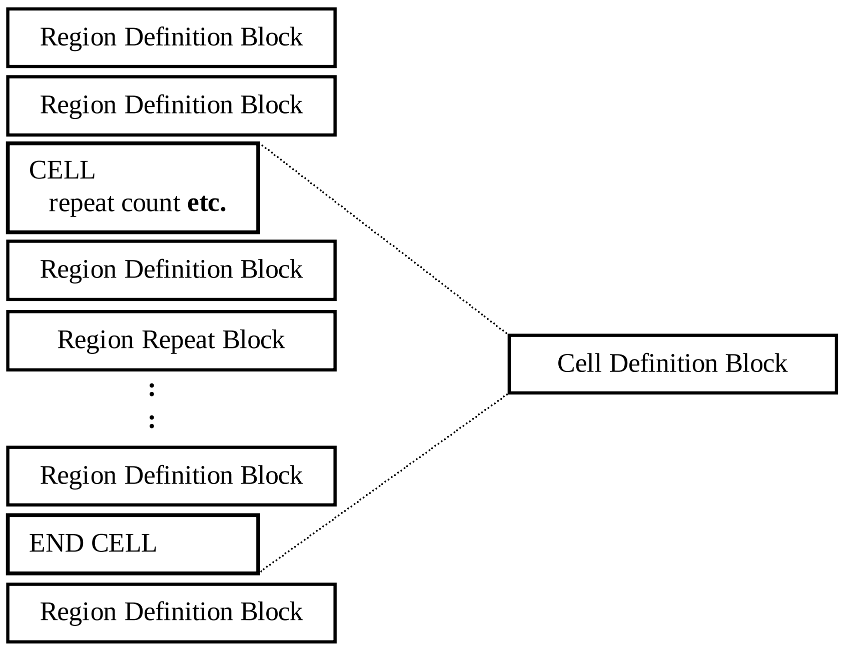
When ***end repeat*** is reached, if the repeat count has not been exhausted, the program goes back to the frist region definition block in the repeat block and does it all again, adding more regions in increasing Z.

There may be any number of repeat blocks describing your apparatus, but they <u>must not overlap</u>. In particular, they <u>cannot be nested</u>; there cannot be a repeat block within another repeat block. (However, there are other ways to skin this cat, as we shall soon see.)

In rare instances, there are cases in which the most trivial repeat block – one with a single region definition block inside – may be useful. For example, suppose that you have put a meter-long chunk of kryptonite in the apparatus, and you want to plot what effect this has on the beam each quarter meter. But the more usual case is that a repeat block produces multiple copies of a more complex (virtual) apparatus.

Repeating Yourself, Part 2: Cells

Another way to group contiguous region descriptor blocks into a single unit is to place them in a ***cell***. Cells, like repeat blocks, come with a repetition count, so that all the region descriptor blocks inside the cell can be executed again and again. In addition, a ***cell*** can contain any number of embedded ***repeat blocks***; this provides one way to get a "two-level do-loop".



If you are almost pathologically observant, you may have noticed that the ***Cell*** statement above is followed by a repetition count **etc.** In fact, there are four lines of additional information following the ***cell*** command; the remaining three (besides the repetition count) specify a cell-wide field, very much like the fields that may inhabit individual regions and sub-regions. When a particle passes through a region in a cell, it sees both the fields specific to the region (or subregion) and the fields deriving from the cell. This is frequently very useful. Suppose you have some complex absorber, with many regions, buried in a solenoid. You could define the solenoid field region by region, but it's a lot easier to lump the absorber regions into a cell, and just define the field once for the whole cell. Naturally, it's essential to maintain consistency here; if part of your absorber happens to be iron or mu-metal, and you're reading in the cell field from a file, ICOOL isn't going to be smart enough to handle the field alteration caused by the included material. But regions can be filled with many substances that *don't* alter the fields appreciably, and cells are extremely useful in these cases.

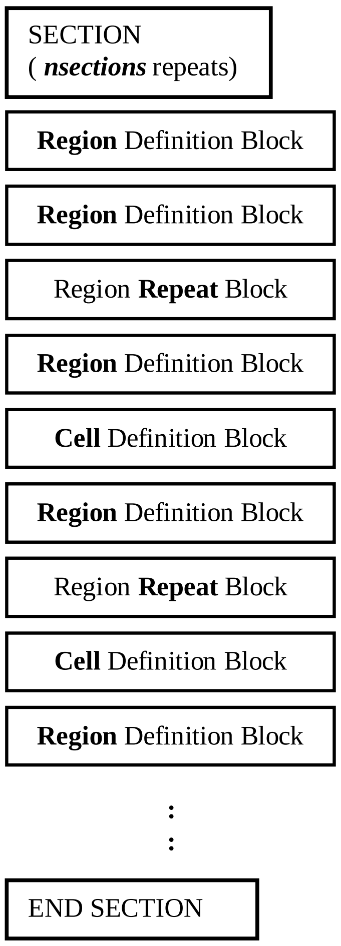
Repeating Yourself, Part 3: Sections

Or to be more exact, ***section***. It is ICOOL convention to precede the first region definition information with a **SECTION** line, and terminate the region definition (and the file) with an **END SECTION** line. I suspect that, once upon a time, sections were conceived of as something there might be more than one of. At the moment, the code does not support multiple sections, though it wouldn't take much work to make it do so. But ***sections*** also come with a repetition count, so you can always set it to something other than 1, which says "repeat the whole region definition shebang". In principle this enables a three-level do-loop:

```
DO section_repetition = 1, nsections
    DO cell_repetition = 1, cell_repetition_count
        DO region_list_repetition = 1, region_list_repetition_count
            region descriptor block #1 (within the list)
            region descriptor block #2 . . . .
```

The section repetition count ***nsections*** is one of the control variables in the ***cont*** namelist, near the beginning of the input file. It defaults to 1.

Schematically summarizing the entire region definition portion of ICOOL's main input file***forr001.dat:***


SECTION
(***nsections*** repeats)
Region Definition Block
Region Definition Block
Region **Repeat** Block
Region Definition Block
Cell Definition Block
Region Definition Block
Region **Repeat** Block
Cell Definition Block
Region Definition Block
:
:
END SECTION

We start with SECTION. Inside the section are regions, repeat blocks, and cells in any order. We end with END SECTION. Repeat blocks specify a list of regions and a repeat count. Cells specify lists of regions and/or repeat blocks of regions, and a cell repetition count of their own.

Again, it's important to remember what the higher-level commands – the repeats, the cells, etc. – are doing. Regions are added to the apparatus in increasing order of Z. Every time a region definition block is encountered, a region of specified Z length is added at the "right hand end" of the apparatus, and the Z pointer is bumped ahead accordingly. Repeating items simply loop through all the region definition elements they contain, for the specified number of repetitions, extending the apparatus yet further in the +Z direction.

In fact, that is ***exactly*** how ICOOL builds its region definition file (for007.dat, an internal file). As ICOOL works its way through the user's input file, it "in-lines" all the repeating elements, making a region-by-weary-region list for the whole apparatus. This file can easily be many thousands of lines long. That having been done, beam particles are generated, and propagated through the regions. First, all the particles are propagated through region 1, then region 2, etc. Region by region, the particles are bent, focused, accelerated, decelerated, absorbed, scattered etc., perhaps even cooled. Along the way, the remaining particle ensemble may be characterized in various ways. File 7 grinds relentlessly on, region after region.... Finally it runs out of user-authorized ways to torture the particles, the survivors are entered into the output beam statistics, and the program ends.