



FLUKA

# Combinatorial Geometry

FLUKA Beginner's Course

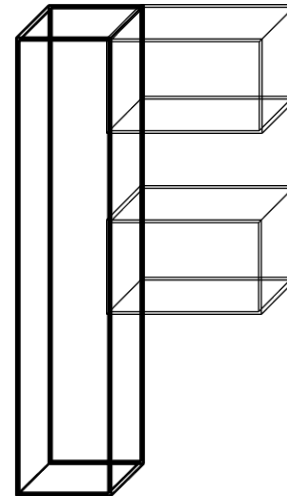
# Introduction

## Principle of Combinatorial Geometry

Basic objects called **bodies** (such as cylinders, spheres, parallelepipeds, etc.) are combined to form more complex objects called **regions**

This combination is done using Boolean operations:  
**union, intersection, and subtraction**

1 complex  
object



3 basic  
objects

# Introduction

Initially, **FLUKA Combinatorial Geometry** was similar to the package developed at ORNL for the neutron and gamma-ray transport program Morse (M.B. Emmett ORNL-4972 1975).

It was based on the original combinatorial geometry by MAGI (Mathematical Applications Group, Inc., W. Guber et al, MAGI-6701 1967).

The present FLUKA CG package has been highly improved:

- addition of "infinite bodies"
- possibility of using **body and region names** instead of numbers and **getting rid of alignment rules**,
- availability of **parentheses**
- **expansion** and **roto-translation** of bodies
- **comments**
- **repetition of patterns** (lattices)
- **voxels**

# Geometry input format

- The input file format for the geometry is different from the one adopted anywhere else in FLUKA (i.e. the number and length of the input fields is different)
- Different formats can be used (due to backward compatibility)
  - **Fixed format**
    - ◆ Alignment is mandatory
    - ◆ Bodies and regions are identified by numbers (rather than names), this makes geometry editing more difficult
  - **Name based format**
    - ◆ **It is the recommended format**
    - ◆ **It is NOT the default!**

# Name based format

It has many advantages, the main are:

- input parameter alignment is not necessary
- bodies and regions are identified by **names**
- possible to edit the input sequence without affecting the region description (e.g. inserting a new body)
- **parentheses can be used to perform complex Boolean operations** in the description of regions.

**Name based format** input is used for both body and region

It is activated

either by a **GLOBAL** command at the beginning of the input file  
or putting **SDUM = COMBNAME** in the **GEOBEGIN** card

<b>GEOBEGIN</b>	Log: ▼ Inp: ▼	Acc: Out: ▼	Opt: ▼ Fmt: COMBNAME ▼
Title:			
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+... <b>GEOBEGIN</b> 0 0			COMBNAME

# Basic Concepts

Four concepts are fundamental in the FLUKA *CG*:

- **Bodies**: basic **convex objects**, plus **infinite planes**, **infinite cylinders**, and **generic quadric surfaces**
- **Zones**: sub-regions defined only via bodies intersection and subtraction
- **Regions**: defined as Boolean operations on bodies (union of zones)
- **Lattices**: duplication of existing objects (translated & rotated), will be explained in a separate lecture

In the original *CG* (Morse) **bodies were convex solid bodies** (i.e. finite portions of space completely delimited by surfaces of first or second degree, i.e. planes or quadrics)

FLUKA has been extended to include: **infinite cylinders** (circular and elliptical), **planes** (half-spaces), and generic **quadrics** (surfaces described by 2<sup>nd</sup> degree equations)

**Use of “infinite bodies” is encouraged**: input are less error-prone and tracking is faster and more accurate.

# Basic Concepts: The Black Hole

All particles entering a black-hole are absorbed (they vanish)

FLUKA geometry MUST be embedded into a **BLCKHOLE** region  
(to avoid tracking particles to infinity)

The outer surface of the **BLCKHOLE** region must be a single closed body (e.g. a sphere).

Further black-hole regions can be defined by the user if desired

# Combinatorial Geometry Input

CG input must respect the following structure:

**GEOBEGIN** card

**VOXELS** card (optional, see advanced geometry lecture)

Geometry title (and reading format options)

Body data

**END** card

Region data

**END** card

**LATTICE** cards (optional, see advanced geometry lecture)

Region volumes (optionally requested by a flag in the Geometry title,  
used together with the **SCORE** command)

**GEOEND** card

Reminder for txt format input files:

an asterisk (\*) in the 1<sup>st</sup> column comments the line



# GEOBEGIN card

<b>GEOBEGIN</b>	Log: ▼ Inp: ▼	Acc: Out: ▼	Opt: ▼ Fmt: COMBNAME ▼
Title:			
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...			
GEOBEGIN			COMBNAME
0	0		

WHAT(1) not used

WHAT(2) set accuracy parameter - **use with extreme care !**

WHAT(3) logical unit for the geometry input.  
Geometry input can be read from file; should be >20.0

WHAT(4) logical unit for the geometry output  
Log feature; should be >20.0 ; Default is standard output

WHAT(5) Parenthesis optimization level (see FLUKA manual)

WHAT(6) not used

SDUM **COMBNAME** or **COMBINAT**

**COMBNAME** selects name based format, **COMBINAT** fixed format

Default: **COMBINAT (!)**

Is overwritten by **WHAT(5)** of a possible **GLOBAL** card

# Geometry TITLE card

This card has no keyword, it is the line that follows the *GEOBEGIN* card (unless voxels are used) and its format is (2I5, 10X, A60)

The card gets three inputs: *IVLFLG*, *IDBG*, *TITLE*

*IVLFLG* (Input VoLumes FLAG) triggers the normalization of the quantities scored in regions by the option *SCORE*

*IVLFLG*= 0 → no normalization applied

*IVLFLG*= 3 → results divided by region volumes given before *GEOEND*.  
Volume value for each region must be provided, format: (7E10.5).

*IDBG* select different kinds of geometry *fixed* format input:

*IDBG* = 0 : default fixed format

*IDBG* = -10 or -100 : high accuracy fixed format

<b>GEOBEGIN</b>	Log: ▼ Inp: 21 ▼	Acc: Out: ▼	Opt: ▼ Fmt: COMBNAME ▼
Title: Geometry title			
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...			
<b>GEOBEGIN</b>	21.		<b>COMBNAME</b>
0 0	Geometry title		

The page features a minimalist design with blue lines and corner markers. A vertical line on the left and a horizontal line at the top intersect at the top-left corner, with a small blue circle at their intersection. A horizontal line at the bottom and a vertical line on the right intersect at the bottom-right corner, also with a small blue circle at their intersection. The word "Bodies" is centered in a large, bold, purple font.

# Bodies

# Bodies

- Each body divides the space into two domains: **inside** and **outside**.
- 3-character code of available bodies:
  - **RPP**: Rectangular Parallelepiped
  - **SPH**: SPHERE
  - **XYP, XZP, YZP**: Infinite half space delimited by a coordinate plane
  - **PLA**: Generic infinite half-space, delimited by a PLANE
  - **XCC, YCC, ZCC**: Infinite Circular Cylinder, parallel to coordinate axis
  - **XEC, YEC, ZEC**: Infinite Elliptical Cylinder, parallel to coordinate axis
  - **RCC**: Right Circular Cylinder
  - **REC**: Right Elliptical Cylinder
  - **TRC**: Truncated Right angle Cone
  - **ELL**: ELLipsoid of revolution
  - **QUA**: QUAdric
- Other (deprecated) bodies: **ARB, RAW, WED, BOX**  
Do not use them, because they can cause rounding problems

# Bodies

The input for each **body** consists of:

- the 3-letter code indicating the body type (RPP, ZCC...)
- a unique "**body name**" (alphanumeric identifier, 8 character maximum, **case sensitive**)
- a set of geometrical quantities defining the body  
(the number depends on the body type, see next slides)

Different items can extend over as many lines as needed  
(Maximum 132 characters per line accepted)

Different items have to be separated by **one or more blanks**,  
or by one of the separators **, / ; :**

**All values are in cm!**

# Infinite half-space parallel to coordinate axis

There are 4 kinds of infinite half-spaces

Three are delimited by planes perpendicular to the coordinate axes:

1. Delimited by a plane  $\perp$  to the  $x$ -axis. **Code: YZP**
2. Delimited by a plane  $\perp$  to the  $y$ -axis. **Code: XZP**
3. Delimited by a plane  $\perp$  to the  $z$ -axis. **Code: XYP**

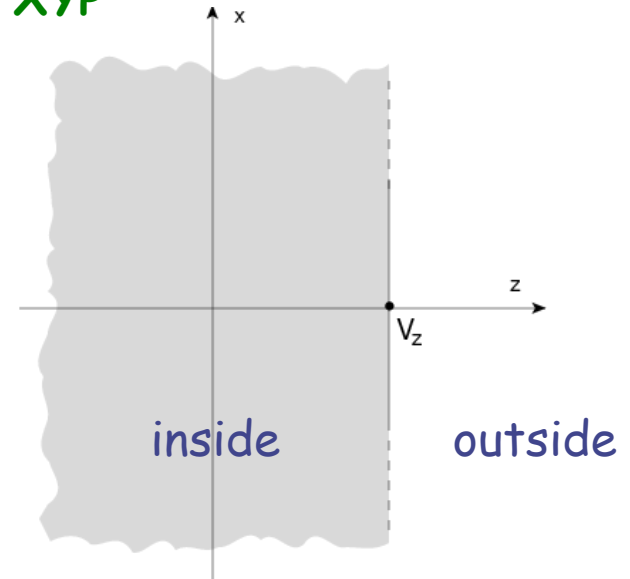
All defined by a single number:

$V_x$  (resp.  $V_y$ , or  $V_z$ ),  
coordinate of the plane on the  
corresponding perpendicular axis

Points for which:

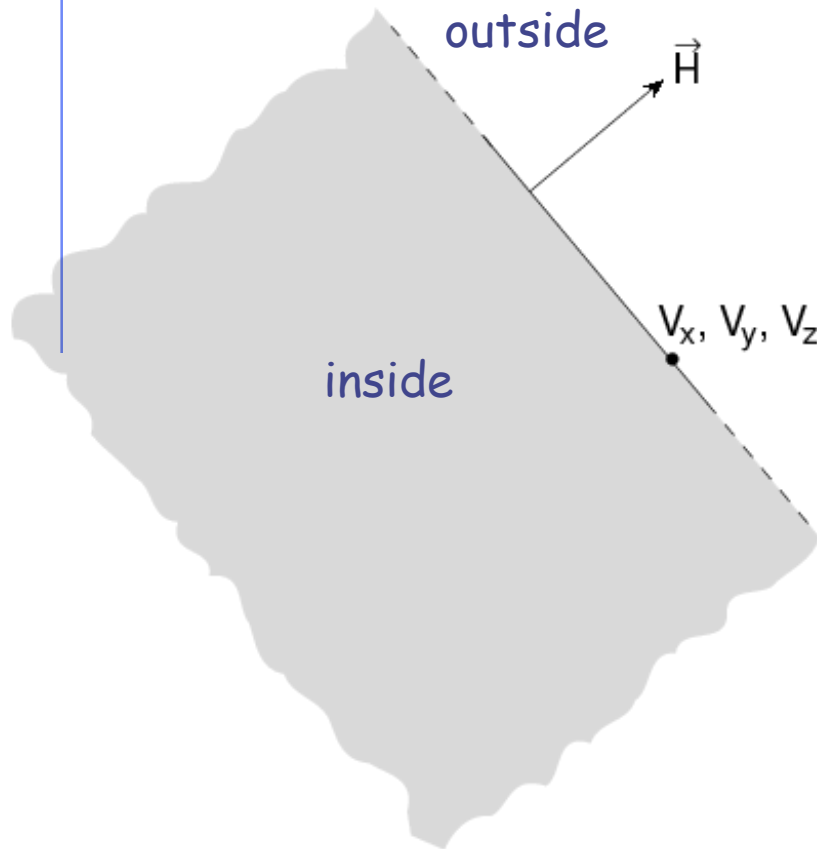
$x < V_x$  (resp.  $y < V_y$ , or  $z < V_z$ )

are "inside the body"



```
XYP          eg_XYP          z:10.0
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...
XYP eg_XYP    10.0
```

# Arbitrarily orientated infinite half-space: **PLA**



A **PLA** defines the infinite half space delimited by a generic plane

A **PLA** is defined by 6 numbers:  
 $H_x, H_y, H_z$  (vector  $\perp$  to the plane, arbitrary length);

$V_x, V_y, V_z$  (any point lying on the plane)

The half-space "**inside the body**" is that from which the vector is pointing (i.e. **the vector points "outside"**).

```
PLA eg_PLA Nx:0.0 Ny:1.0 Nz:1.0
          x:1.  y:2.  z:3.
+...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...
PLA eq_PLA 0.0 1.0 1.0 1. 2. 3.
```

# Rectangular Parallelepiped: **RPP**

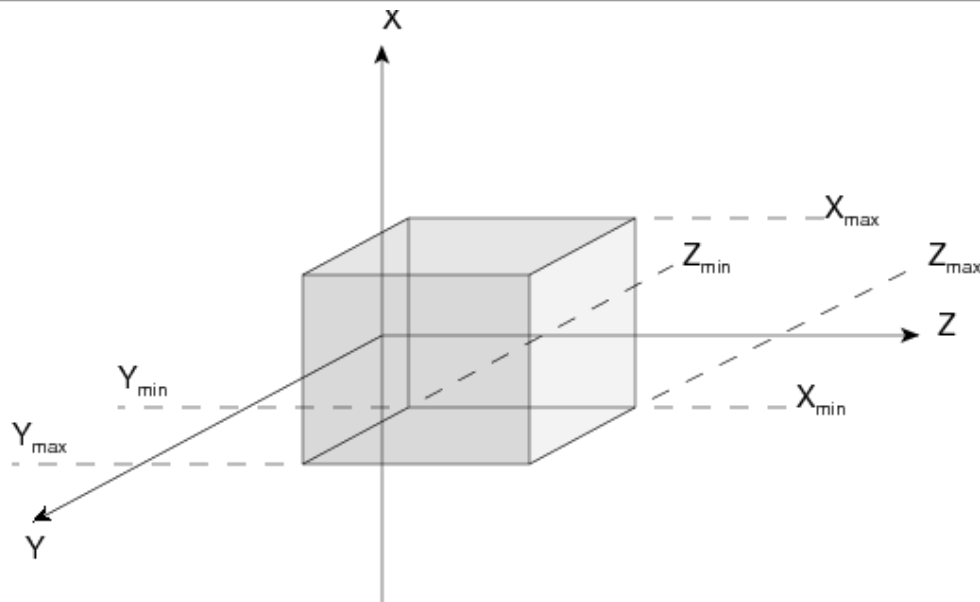
An **RPP** has its edges parallel to the coordinate axes  
It is defined by 6 numbers in the **following order**:

$X_{\min}$ ,  $X_{\max}$ ,  $Y_{\min}$ ,  $Y_{\max}$ ,  $Z_{\min}$ ,  $Z_{\max}$

(min and max coordinates delimiting the parallelepiped)

```
RPP      eg_RPP      Xmin: -5.0      Xmax: 5.0
          Ymin: 0.0      Ymax: 10.0
          Zmin: -30.0     Zmax: 30.0

*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...
RPP eg_RPP      -5.0 5.0 0.0 10.0 -30.0 30.0
```





# Rectangular Parallelepiped: **RPP**

```

RPP      eg_RPP      Xmin: -5.0      Xmax: 5.0
          Ymin: 0.0      Ymax: 10.0
          Zmin: -30.0     Zmax: 30.0
  
```

---

```

* . . . + . . . 1 . . . + . . . 2 . . . + . . . 3 . . . + . . . 4 . . . + . . . 5 . . . + . . . 6 . . . + . . . 7 . . . + . . .
RPP eg_RPP      -5.0 5.0 0.0 10.0 -30.0 30.0
  
```

An **RPP** definition extends over one single card in default fixed format, or over two cards in high-accuracy body fixed format

Example in **fixed format** (the comment lines shown are allowed input lines):

```

* . . . + . . . 1 . . . + . . . 2 . . . + . . . 3 . . . + . . . 4 . . . + . . . 5 . . . + . . . 6 . . . + . . . 7 . . . + . . .
RPP      4      -20.0      +20.0      -50.0      +50.0      -38.5      +38.5
  
```

\* (a parallelepiped centered on the origin)

## High-accuracy fixed format

```

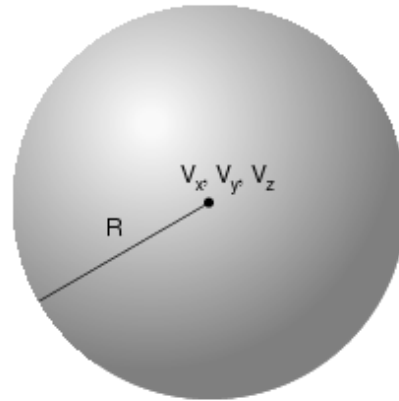
* . . . + . . . 1 . . . + . . . 2 . . . + . . . 3 . . . + . . . 4 . . . + . . . 5 . . . + . . . 6 . . . + . . . 7 . . . + . . .
RPP      4      -20.0      +20.0      -50.0
          +50.0      -38.5      +38.5
  
```

# Sphere

## Sphere: SPH

A SPH is defined by 4 numbers:

$V_x$ ,  $V_y$ ,  $V_z$  (coordinates of the centre),  $R$  (radius)



SPH	eg_SPH	x:0.0	y:0.0	z:0.0
		R:10.0		

```
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...
SPH eg_SPH 0.0 0.0 0.0 10.0
```

# Circular cylinder

## Right circular cylinder: RCC

An **RCC** can have any orientation in space

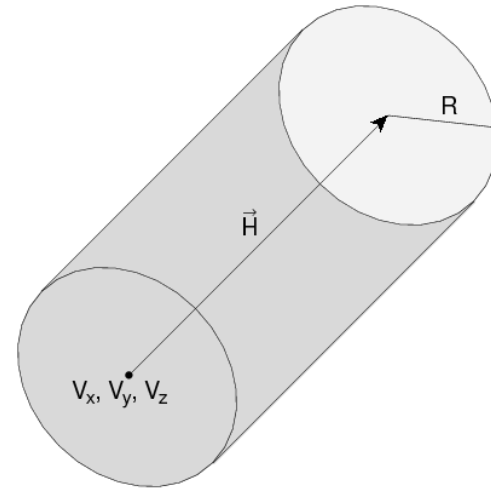
Limited by a cylindrical surface and two plane faces  $\perp$  to its axis.

Each **RCC** is defined by 7 numbers:

$V_x, V_y, V_z$  (centre of one face);

$H_x, H_y, H_z$  (vector corresponding to the cylinder height, pointing toward the other face);

$R$  (cylinder radius).



<b>RCC</b>	eg_RCC	x:0.0	y:0.0	z:0.0
		Hx:0.0	Hy:10.0	Hx:10.0
		R:2.0		

```
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...
RCC eg_RCC 0.0 0.0 0.0 0.0 10.0 10.0 2.0
```

# Infinite cylinders

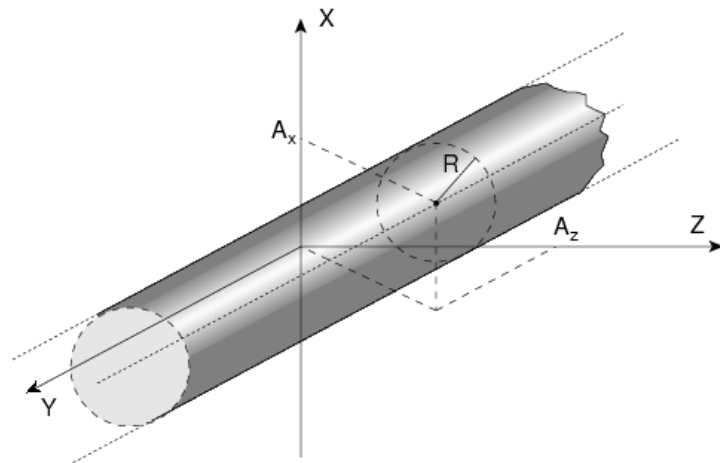
Infinite Circular Cylinder parallel to coordinate axis: **XCC, YCC, ZCC**

Each **XCC** (**YCC, ZCC**) is defined by 3 numbers:

**A<sub>y</sub>, A<sub>z</sub>** for **XCC**

(**A<sub>z</sub>, A<sub>x</sub>** for **YCC**, **A<sub>x</sub>, A<sub>y</sub>** for **ZCC**)  
(coordinates of the cylinder axis),

**R** (cylinder radius)



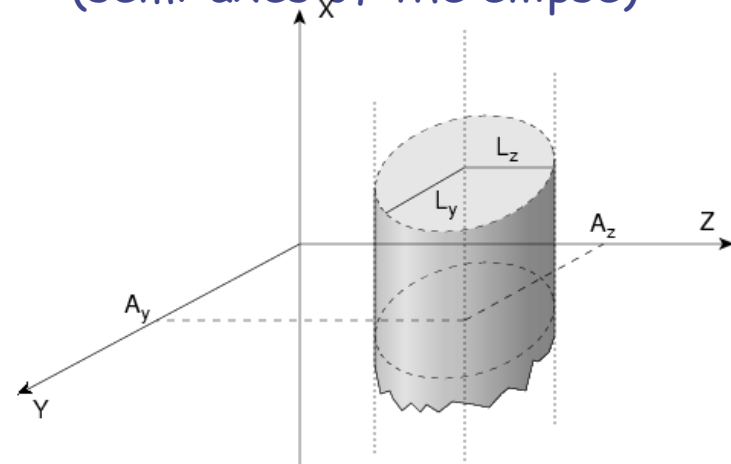
Infinite Elliptical Cylinder parallel to coordinate axis: **XEC, YEC, ZEC**

Each **XEC** (**YEC, ZEC**) is defined by 4 numbers: **A<sub>y</sub>, A<sub>z</sub>** for **XEC**

(**A<sub>z</sub>, A<sub>x</sub>** for **YEC**, **A<sub>x</sub>, A<sub>y</sub>** for **ZEC**)  
(coordinates of the cylinder axis);

**L<sub>y</sub>, L<sub>z</sub>** for **XEC**

(**L<sub>z</sub>, L<sub>x</sub>** for **YEC**, **L<sub>x</sub>, L<sub>y</sub>** for **ZEC**)  
(semi-axes of the ellipse)



```
ZCC      eg_ZCC      x:0.0      y:0.0      R:20.0
+...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...
ZCC eg_ZCC      0.0 0.0 20.0
```

# Arbitrary generic quadric: **QUA**

A **QUA** allows to insert a quadric surface

This is defined by a 2<sup>nd</sup> degree equation  $F(x,y,z) = 0$

Each **QUA** is defined by 10 numbers:

$A_{xx}, A_{yy}, A_{zz}, A_{xy}, A_{xz}, A_{yz}, A_x, A_y, A_z, A_0$

corresponding to the equation:

$$A_{xx} x^2 + A_{yy} y^2 + A_{zz} z^2 + A_{xy} xy + A_{xz} xz + A_{yz} yz + \\ + A_x x + A_y y + A_z z + A_0 = 0$$

A **QUA** definition extends over two cards in default fixed format,  
and over 4 cards in high-accuracy body fixed format.

The slide features a decorative layout of thin blue lines. A vertical line on the left and a horizontal line at the top intersect at a small circle in the top-left corner. Another horizontal line is positioned below the top one, and a vertical line on the right intersects it at a small circle in the bottom-right corner. The word "Regions" is centered between the two horizontal lines.

# Regions

# Concept

Regions are defined as combinations of bodies obtained by boolean operations:

	Union	Subtraction	Intersection
Name based format		-	+
Fixed format	OR	-	+
Mathematically	$\cup$	-	$\cap$

Regions are not necessarily simply connected (they can be the union of two or more non contiguous or partially overlapping zones) but must be of homogeneous material composition

Each point of space must belong to one and only one region!

# Regions

Input for each region starts on a new line and extends on as many continuation lines as are needed. It is of the form:

```
REGNAME NAZ boolean-zone-expression
```

or

```
REGNAME NAZ boolean-zone-expression | boolean-zone-expression | ...
```

- **REGNAME** is the **region "name"**  
alphanumeric identifier, 8 character maximum, **case sensitive**  
Must start by an alphabetical character
- **NAZ** See next slide
- **"boolean-zone-expression"** See next slides



# Regions

- **NAZ** (Number of **A**djacent **Z**ones) is a rough estimate of the number of zones a particle can enter when leaving the current region zones (5 by default). What actually matters is the NAZ sum over all regions, defining the size of the *contiguity list*
- While tracking, the program searches in the contiguity list for the neighbor zones of each zone. If the zone is not yet in the list, the whole geometry is scanned and it is added to the list with its neighbor zones
- When the NAZ limit is reached (i.e. the list is full) the code prints a warning: GEOMETRY SEARCH ARRAY FULL. This is not lethal: the calculation continues but with a reduced efficiency
- If you have more than 1000 regions, you must issue a **GLOBAL** card putting in **WHAT(1)** a higher limit (not beyond 10000)

# Regions Boolean expression

- **zones** are defined by intersections and/or subtraction of **bodies**, each body being preceded by its + or - sign
- | (OR) operators are used to join **zones**
- In evaluating the expressions, the highest operator precedence is given to parentheses, followed by +, - and the | operator
- Parentheses are available in name based format (use with care!)
- If one line is not sufficient, any number of continuation lines can be added
- Blanks are ignored

# Meaning of + and - operators

If a **body** number (or name) appears in a **zone** description preceded by a + operator, it means that the **zone** being described is **wholly contained inside** the **body**.

If a **body** number appears in a **zone** description preceded by a - operator, it means that the **zone** being described is **wholly outside** the **body**.

**Zones** must be finite: normally in the description of each **zone** and hence of each **region** the symbol + must appear at least once.

# Meaning of the + and – operators

Target 5 +Outer +CutPlane

- \* (the above region is the part of space common to bodies Outer and CutPlane)

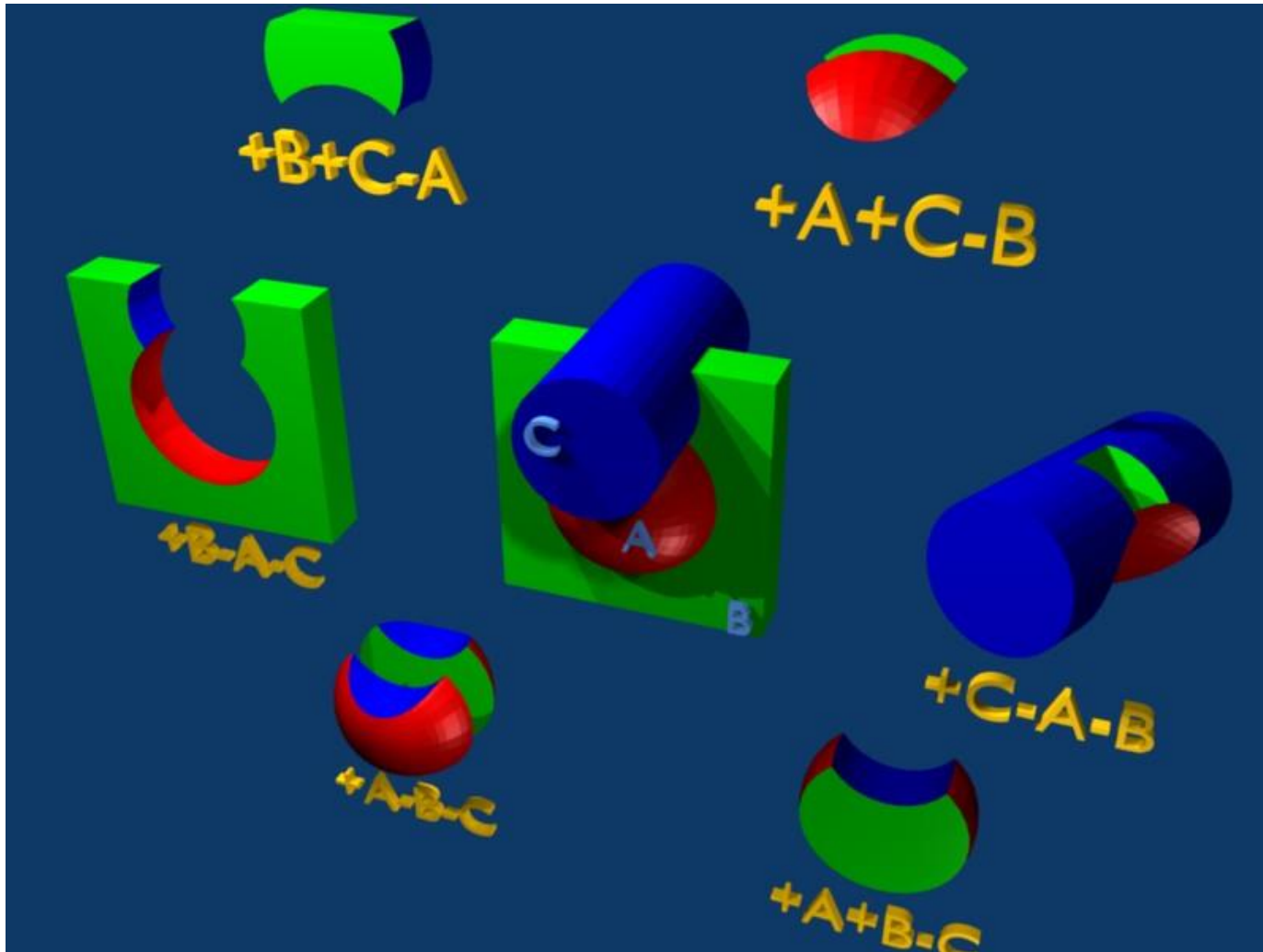
Regex 5 +Red -Blue -Green +Yellow

- \* (the above region is the part of space common to body Red and Yellow,
- \* excluding however what is inside body Blue and what is inside Green)

Air 5 +Room

- \* (the latter region coincides entirely with body Room)

# Meaning of + and - operators



# Meaning of the | (OR) operator

The | (or OR) operator is used as a Boolean **union** operator in order to combine **zones** (**sub regions** partially overlapping or not).

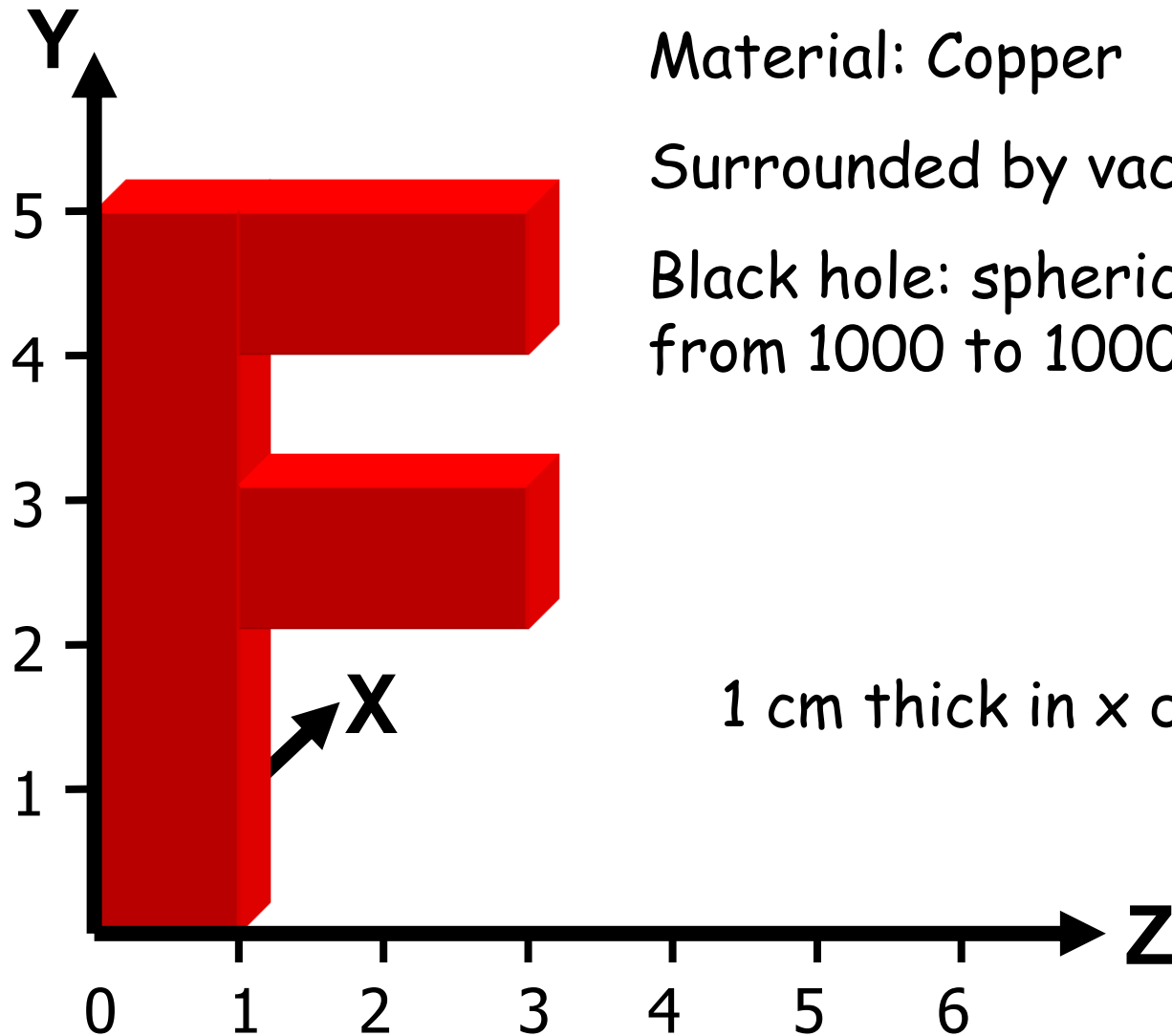
**Zones** are formed as body intersections or subtractions as previously explained

The **regions** are formed by the union of these **zones**.

Examples:

```
Ground 5 | +Body9 | +Body15 | +Body1 | +Body8 -Body2 | +Body8 -Body3
*          <- 1st -><- 2nd -><- 3rd -><----- 4th -----><----- 5th ---->
* (continuation line)
          | +Body8 +Body18 | +Body12 -Body10 -Body11 -Body13 -Body14
*          <----- 6th -----><----- 7th and last zone ----->
```

# Geometry Example "F" shaped target



Material: Copper

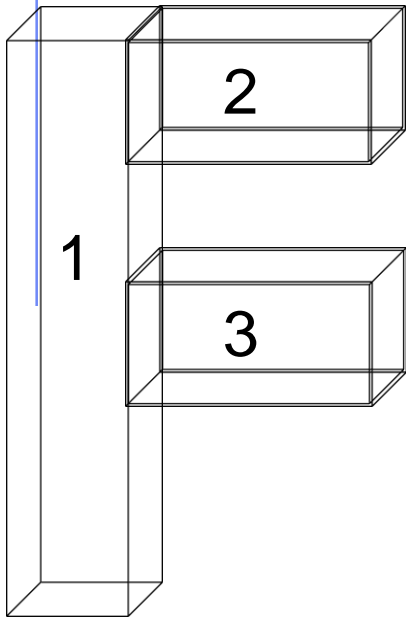
Surrounded by vacuum

Black hole: spherical shell  
from 1000 to 10000 cm radius

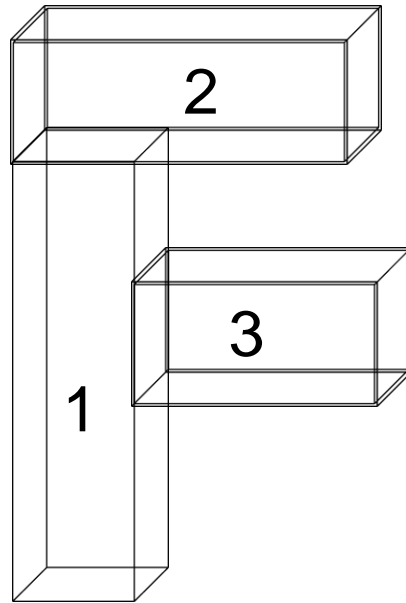
1 cm thick in x direction

# Geometry example "F": Bodies

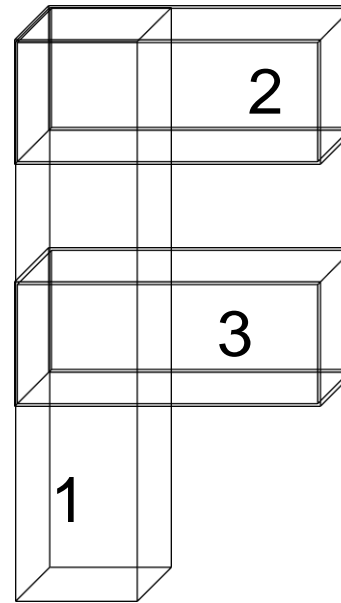
Several possibilities for bodies, e.g.:



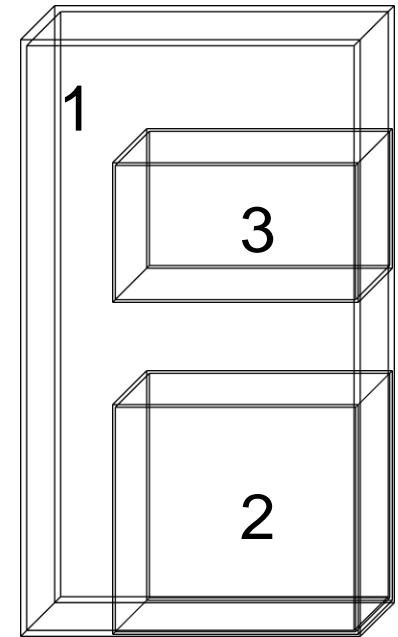
(A) Separated bodies



(B) Separated bodies



(C) overlapping

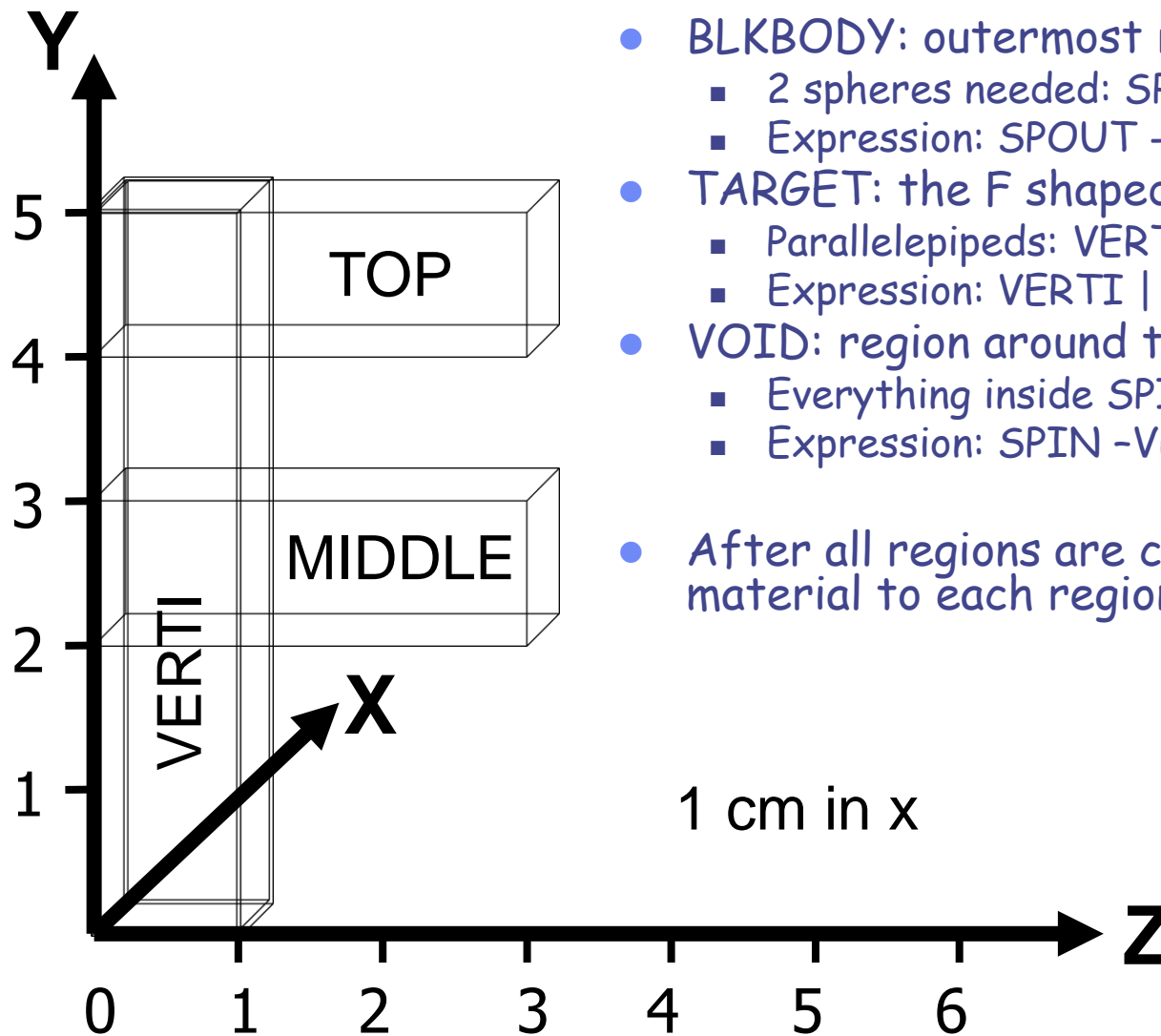


(D) subtraction

The 4 options are equivalent. We will use C.



# Geometry example "F": REGIONS



- BLKBODY: outermost region like a shell
  - 2 spheres needed: SPOUT, SPIN
  - Expression: SPOUT - SPIN
- TARGET: the F shaped object
  - Parallelepipeds: VERTI, TOP, MIDDLE
  - Expression: VERTI | TOP | MIDDLE
- VOID: region around the target
  - Everything inside SPIN that is not target
  - Expression: SPIN -VERTI - TOP -MIDDLE
- After all regions are created: assign material to each region with ASSIGNMA

# Geometry example "F": input file

COMBNAME

- ```
GEOBEGIN
      0      0      The copper F
SPH SPIN      0.0 0.0 0.0 1000.
SPH SPOUT     0.0 0.0 0.0 10000.
RPP VERTI     0.0 1. 0.0 5. 0.0 1.
RPP TOP       0.0 1. 4. 5. 0.0 3.
RPP MIDDLE    0.0 1. 2. 3. 0.0 3.
END
* Black hole
BLKBODY      5  +SPOUT -SPIN
* Void around
VOID         5  +SPIN -TOP -VERTI -MIDDLE
* Target
TARGET       5  +TOP | +VERTI | +MIDDLE
END
GEOEND
ASSIGNMA     BLCKHOLE   BLKBODY
ASSIGNMA     VACUUM     VOID
ASSIGNMA     COPPER     TARGET
```

# Geometry example "F": input Flair

The screenshot shows the 'Input' window of a software application. The window title is 'Input' and it contains a list of geometric bodies and regions. The left sidebar shows a tree view with 'Input' expanded to 'Bodies'. The main area displays the following data:

| Body/Region           | Material      | X            | Y         | Z         | Other Parameters          |
|-----------------------|---------------|--------------|-----------|-----------|---------------------------|
| <b>GEOBEGIN</b>       |               |              |           |           | Title: The copper F       |
| <b>SPH blkbody</b>    | blkbody       | 0.0          | 0.0       | 0.0       | R: 100000.0               |
| <b>SPH void</b>       | void          | 0.0          | 0.0       | 0.0       | R: 10000.0                |
| <b>RPP VERTI</b>      |               | Xmin: 0.0    | Ymin: 0.0 | Zmin: 0.0 | Xmax: 1, Ymax: 5, Zmax: 1 |
| <b>RPP TOP</b>        |               | Xmin: 0.0    | Ymin: 4   | Zmin: 0.0 | Xmax: 1, Ymax: 5, Zmax: 3 |
| <b>RPP MIDDLE</b>     |               | Xmin: 0.0    | Ymin: 2   | Zmin: 0.0 | Xmax: 1, Ymax: 3, Zmax: 3 |
| <b>END</b>            |               |              |           |           |                           |
| <b>REGION BLKBODY</b> |               |              |           |           | Neigh: 5, Volume:         |
| <b>REGION VOID</b>    |               |              |           |           | Neigh: 5, Volume:         |
| <b>REGION TARGET</b>  |               |              |           |           | Neigh: 5, Volume:         |
| <b>END</b>            |               |              |           |           |                           |
| <b>GEOEND</b>         |               |              |           |           |                           |
| <b>ASSIGNMA</b>       | Mat: BLCKHOLE | Reg: BLKBODY | to Reg:   | Field:    |                           |
| <b>ASSIGNMA</b>       | Mat: VACUUM   | Reg: VOID    | to Reg:   | Field:    |                           |
| <b>ASSIGNMA</b>       | Mat: COPPER   | Reg: TARGET  | to Reg:   | Field:    |                           |

# Geometry example "F": input Flair

The screenshot displays the Geometry software interface. The main workspace is divided into four viewports: Front, U:V, VOID Left, and VOID Back. The Front view shows a 2D projection of the block with a central void, labeled 'VOID'. The U:V view shows a 3D perspective of the block. The VOID Left and VOID Back views show the block from the left and back perspectives, respectively, with the void highlighted. The Properties panel on the left shows the following details:

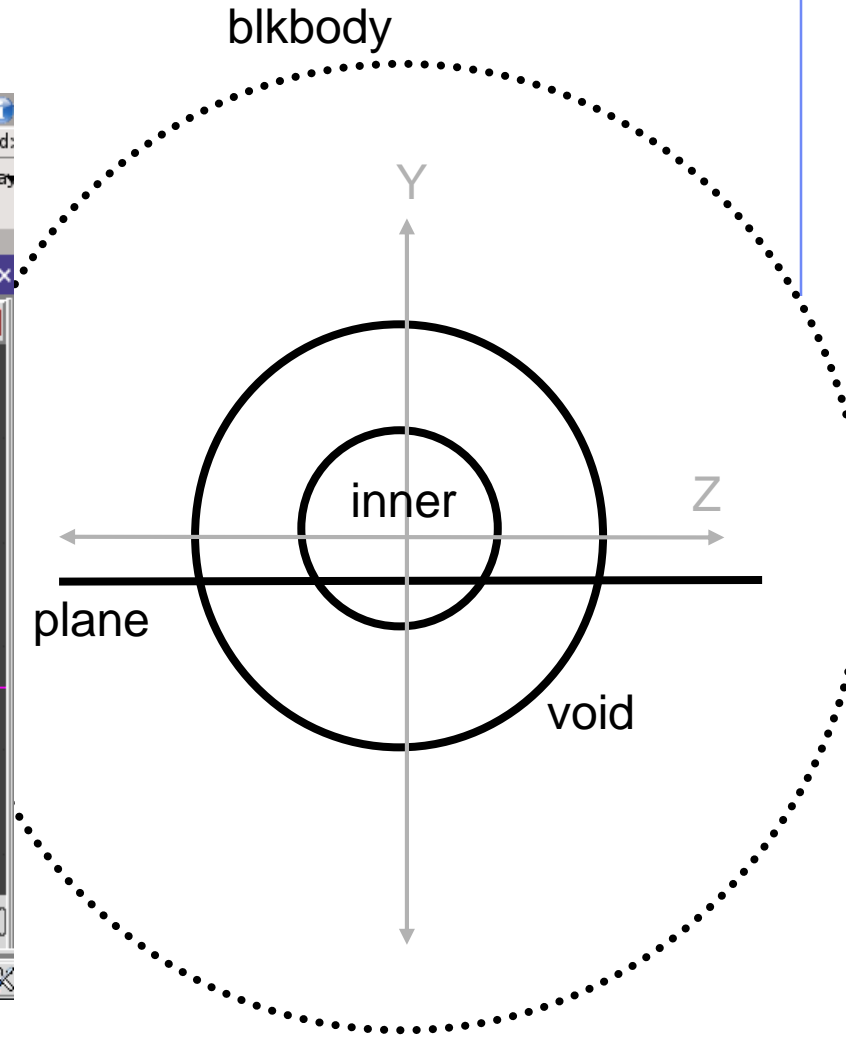
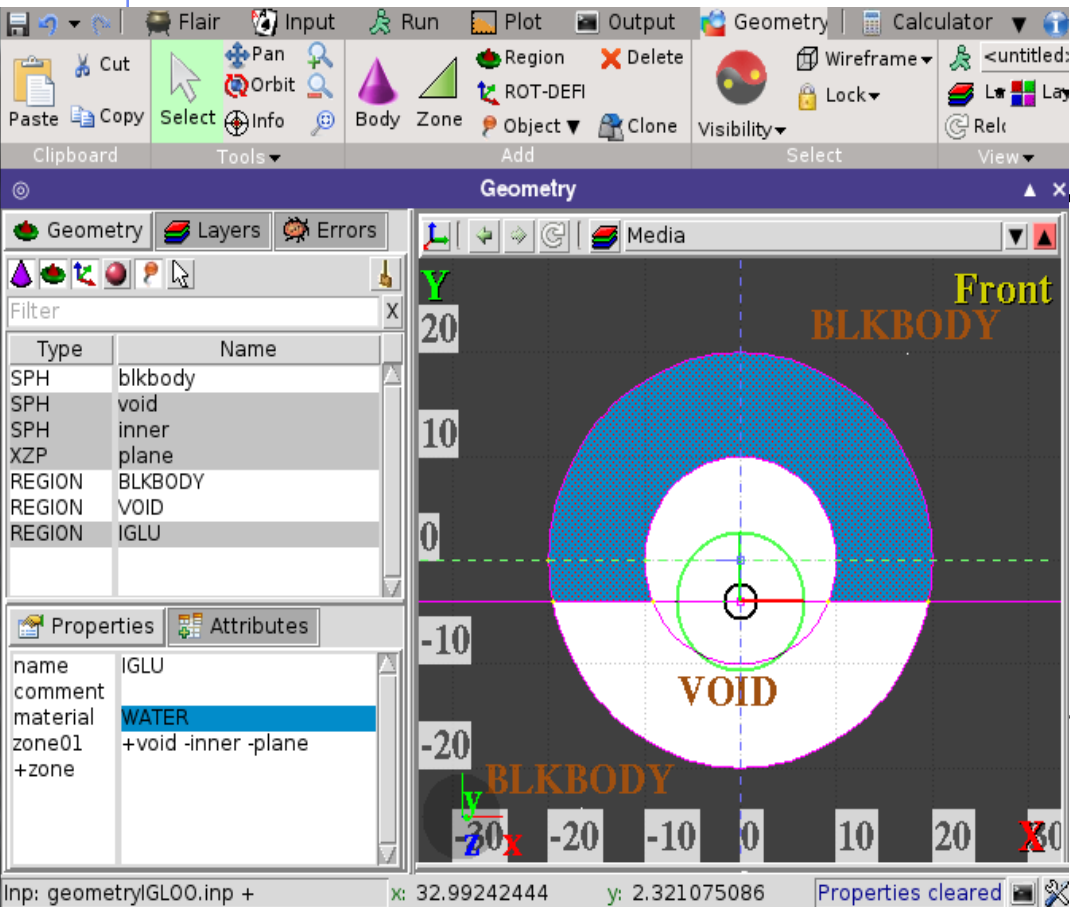
| Type   | Name    |
|--------|---------|
| SPH    | blkbody |
| SPH    | void    |
| RPP    | VERTI   |
| RPP    | TOP     |
| RPP    | MIDDLE  |
| REGION | BLKBODY |
| REGION | VOID    |
| REGION | TARGET  |

The Properties panel also shows the following details:

| Properties | Attributes |
|------------|------------|
| name       | TARGET     |
| comment    |            |
| material   | COPPER     |
| zone01     | +TOP       |
| zone02     | +VERTI     |
| zone03     | +MIDDLE    |
| +zone      |            |

The status bar at the bottom indicates the input file is geometry.F.inp and provides coordinates: x: -1.049774245, y: 2.336612972, z: 1.721566. The Zones are listed as zone03.

# Example: Igloo



# Example: Igloo

The screenshot displays the Flair software interface for editing an input file. The main window shows the following content:

```

-----TITLE ... BEAMPOS : 4 cards-----
GEOBEGIN
  Title:
  SPH blkbody x: 0.0 y: 0.0 z: 0.0
             R: 100000.0
  SPH void x: 0.0 y: 0.0 z: 0.0
           R: 20.0
  SPH inner x: 0.0 y: 0.0 z: 0.0
           R: 10.0
  XZP inner1 y: -4.0
END
REGION BLKBODY Neigh: 5 Volume:
  expr: +blkbody -void
REGION VOID Neigh: 5 Volume:
  expr: +inner
        | +void +inner1 -inner
REGION IGLU Neigh: Volume:
  expr: +void -inner -inner1
END
GEOEND
ASSIGNMA Mat: BLK HOLE Reg: BLK BODY to Reg:
         Mat(Decay): Step: Field:
ASSIGNMA Mat: VACUUM Reg: VOID to Reg:
         Mat(Decay): Step: Field:
ASSIGNMA Mat: WATER Reg: IGLU to Reg:
         Mat(Decay): Step: Field:
-----+-----1-----+-----2-----+-----3-----+-----4-----+-----5-----+-----6-----+-----7-----+-----
GEOBEGIN COMBNAME
  0 0
  
```

The status bar at the bottom indicates: Inp: geometry\IGLOO.inp + Card:5 Displayed:17 Total:21

The page features a decorative layout with blue lines and corner markers. A vertical line on the left and a horizontal line at the top intersect at a small circle in the top-left corner. Another horizontal line is positioned below the top one, and a vertical line on the right intersects it at a small circle in the bottom-right corner. The word "Remarks" is centered between the top and middle horizontal lines.

# Remarks

# Important Notes

- Whenever it is possible, the following bodies should be preferred:

PLA, RPP, SPH, XCC, XEC, XYP  
XZP, YCC, YEC, YZP, ZCC, ZEC, QUA

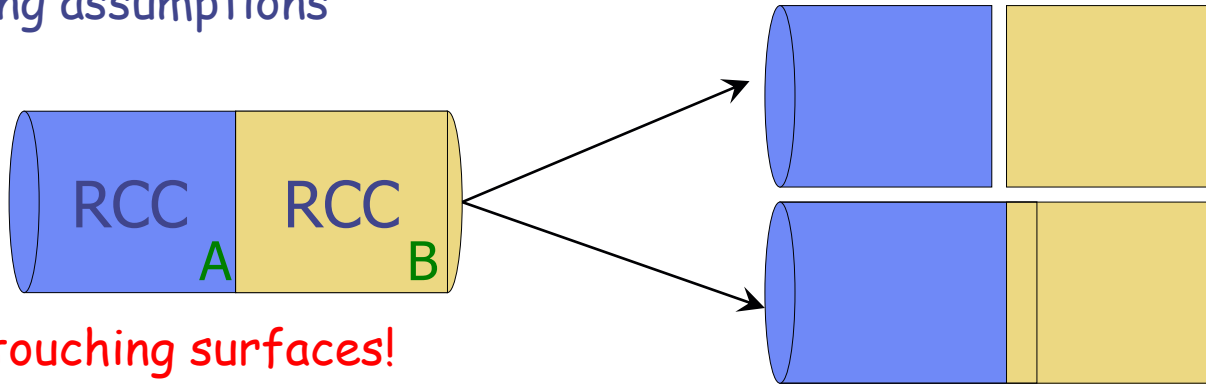
These bodies make the tracking faster, since for them extra coding ensures that unnecessary boundary intersection calculations are avoided when the length of the next step is smaller than the distance to any boundary of the current **region**.

- Always **use as many digits as possible** in the definition of the body parameters, particularly for body heights (RCC, REC, TRC), and for direction cosines of bodies with slant surfaces. The name based format or the high-accuracy fixed format should always be used in these cases.



# Precision Errors

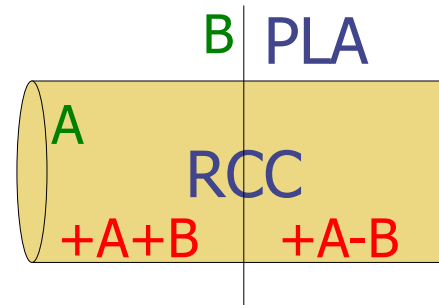
Modeling assumptions



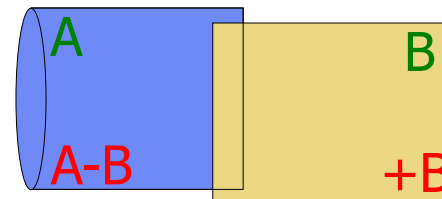
**Avoid touching surfaces!**

When floating point operations are involved.

- Use cutting surface **B** instead



- Or force partial overlap of bodies



# Tracking accuracy

- ❑ FLUKA uses systematically double precision mathematic (i.e. 16 significant digits)
- ❑ GEOBEGIN's WHAT(2)\*10<sup>-6</sup>cm is the *absolute accuracy (AA)* requested for tracking and boundary identification
- ❑ The *relative accuracy (RA)* achievable in double precision is of the order of 10<sup>-14</sup>-10<sup>-15</sup>.
- ❑ AA should be larger than  $RA * L$  , being  $L$  the largest coordinate value in the considered problem, i.e. the whole geometry size (The outer blackhole shell containing the system does not count)
- ❑ For very large and very small geometries, you may, respectively, need to increase or decrease the WHAT(2) default value of 0.0001.

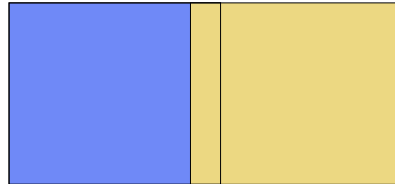
The slide features a decorative layout of blue lines and corner markers. A vertical line on the left and a horizontal line at the top intersect at the top-left corner, with a small blue circle at their intersection. A horizontal line at the bottom and a vertical line on the right intersect at the bottom-right corner, also with a small blue circle at their intersection. The title "Geometry Errors and Debugging" is centered in the upper half of the slide.

# Geometry Errors and Debugging

# Geometry Errors

During execution the code always needs to know the region where a particle is located at every step

- The program will **stop** only if a particle position **does not belong to any region**  
An error message will be printed in the **.err** file with the particle position
- **IMPORTANT!** It will **not stop** if a particle position **belongs to more than one region**. It will accept the first region it finds but the results will completely unreliable!!



# Geometry Errors

## Further types of errors

- Problem space not enclosed by a black body region
- Never start a primary particle along a surface. You could get a geometry error even if the geometry is correct because FLUKA cannot determine the region
- Precision errors
- Lattice replica  $\leftrightarrow$  basic cell mismatch  
(see advanced geometry lecture)

## Debugging Tools

- **GEOEND** card with the **DEBUG** option  
(handled through a dedicated Flair frame)
- Error messages during simulation in the **.err** file
- Geometry plotting by Flair  
(automatically invoking the **PLOTGEOM** card)
- **FLAIR Geometry Editor** (very powerful! See dedicated lecture)
- Simplegeo

# Debugging with FLUKA

**GEOEND** card activates the geometry debugger. Detects both undefined or multiple defined points in a selected X,Y,Z mesh

- Two cards are needed  
First card

WHAT(1)=X<sub>max</sub>  
WHAT(4)=X<sub>min</sub>  
SDUM = DEBUG

WHAT(2)=Y<sub>max</sub>  
WHAT(5)=Y<sub>min</sub>

WHAT(3)=Z<sub>max</sub>  
WHAT(6)=Z<sub>min</sub>

- Second Card

WHAT(1)=Nx  
SDUM = &

WHAT(2)=Ny

WHAT(3)=Nz

|               |             |             |             |             |             |             |              |
|---------------|-------------|-------------|-------------|-------------|-------------|-------------|--------------|
| <b>GEOEND</b> | <b>Xmax</b> | <b>Ymax</b> | <b>Zmax</b> | <b>Xmin</b> | <b>Ymin</b> | <b>Zmin</b> | <b>DEBUG</b> |
| <b>GEOEND</b> | <b>Nx</b>   | <b>Ny</b>   | <b>Nz</b>   |             |             |             | <b>&amp;</b> |

|               |  |       |  |       |  |  |                |
|---------------|--|-------|--|-------|--|--|----------------|
| <b>GEOEND</b> |  |       |  |       |  |  | <b>DEBUG</b> ▼ |
|               |  | Xmin: |  | Xmax: |  |  | NX:            |
|               |  | Ymin: |  | Ymax: |  |  | NY:            |
|               |  | Zmin: |  | Zmax: |  |  | NZ:            |

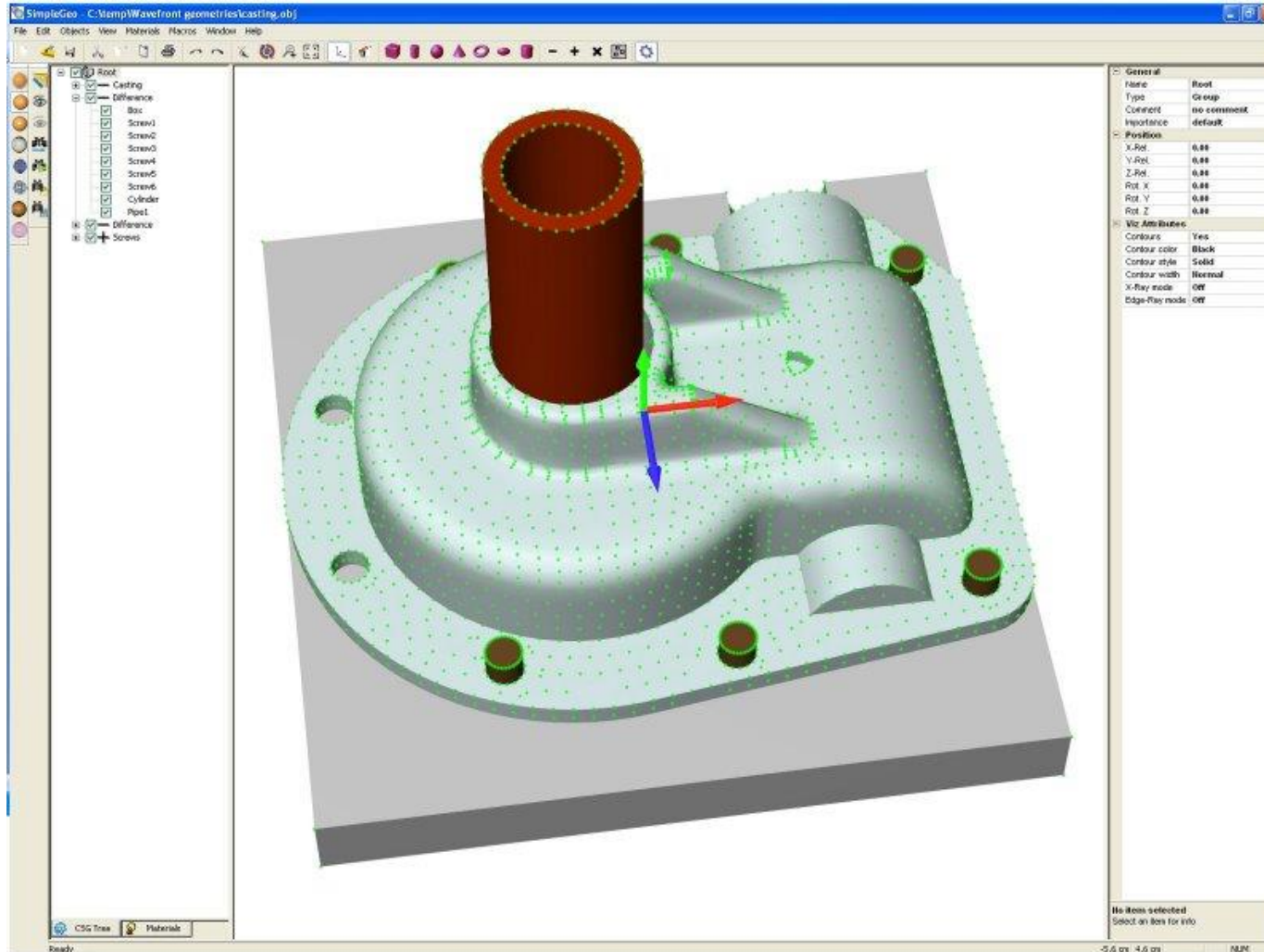
## WARNING!

The program stops if too many errors are found  
A message will be issued on the output unit

# Debugging with FLUKA

- If **no error** is found, no **.err** file will be created
- **REMINDER**: If the debugger does not find any error it does not mean that the geometry is error free!
- One has to test, changing the **GEOEND** settings especially for critical and complicated regions
- Errors will be listed in the **.err** file in the form:
  - \*\*\*\* Lookdb: Geometry error found \*\*\*\*  
\*\*\*\* The point: -637.623762 -244.554455 -96.039604 \*\*\*\*
  - **Point is contained in more than one region**  
\*\*\*\* is contained in more than 1 region \*\*\*\*  
\*\*\*\* (regions: 6 7) \*\*\*\*
  - **Not contained in any region**  
\*\*\*\* is not contained in any region

# Auxiliary program: *SimpleGeo*





# Auxiliary program: *SimpleGeo*

- SimpleGeo is an interactive solid modeler which allows for flexible and easy creation of the models via drag & drop, as well as on-the-fly inspection
- Imports existing geometries for viewing
- Creating new geometries from scratch
- Export to various formats (FLUKA, MCNP, MCNPX)
- Download, Tutorials, etc.:  
<http://theis.web.cern.ch/theis/simplegeo>
- Operating system: Windows only

