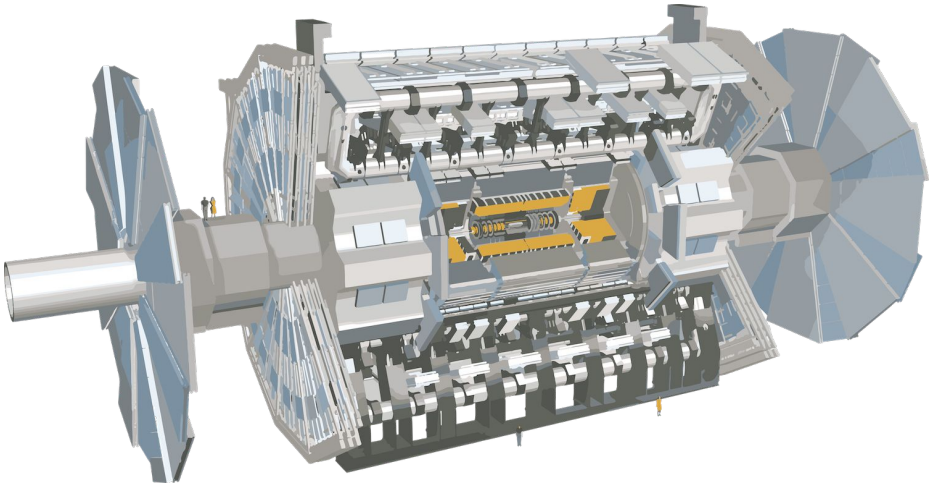
# ROOT Part 3

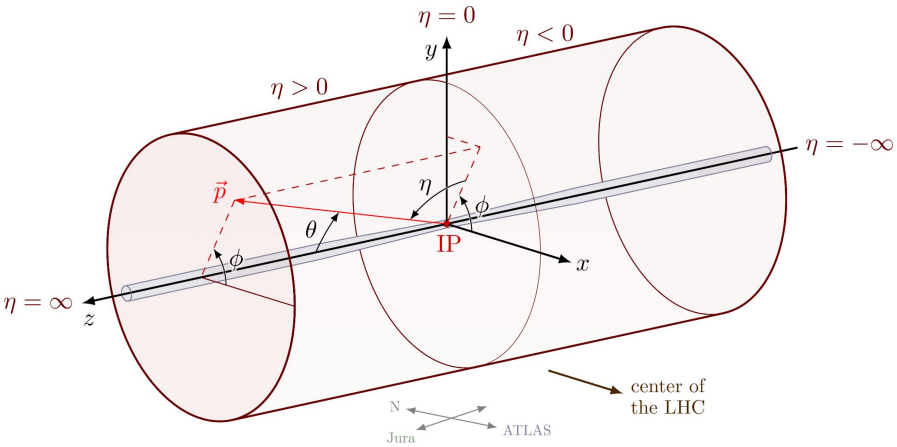
# Recap

- ROOT files (*.root) can hold any type of ROOT object
- A TTree stores information in a set of branches for every entry
  - Able to easily browse and draw information in a TTree
- TBrowser allows browsing files and drawing histograms and branches
- C++ ROOT macros can be used to perform complex ROOT analyses
  - Interpreted using CLING
- TLorentzVector provides methods for relativistic calculations

# Detector Coordinates



https://atlas.cern/Discover/Detector




https://tikz.net/axis3d_cms/

# Angular distances

- 4-vectors are defined with η and φ angular directions
- Distances in detector space are defined as angular difference
  - Consistently defined at any radius from the interaction point

$$\Delta R = ((\Delta\eta)^2 + (\Delta\varphi)^2)^{1/2}$$

- TLorentzVector provides DeltaR() and DeltaPhi() methods

```
TLV1.DeltaR(TLV2)
```

- Note: for distance calculations, Δφ must be [-π,π]
  - By-hand calculations often fall outside of this range and need to be adjusted

# Save objects to output ROOT file

- ROOT objects need to be added to files explicitly
- Create a new output file:

```
TFile *outFile = new TFile("output.root","RECREATE");
```

- The TObject Write() function saves object to current directory
- Latest directory (or file) to be used is the current directory
- It is useful to call file->cd() before calling Write()
- If writing same object to file multiple times, multiple snapshots are saved

# Create and save an output TTree

- Storing information in a TTree is useful for later analysis
  - Generally useful when simplifying information for quickly repeatable analysis
    - Significantly reduce amount of information and number of events once and then analyze remaining information multiple times
- Declare new branches with Branch() and populate tree with Fill()

```
float dRHH;
outTree->Branch("dRHH", &b_dRHH);
for(...) {
        dRHH = H1.DeltaR(H2);
        outTree->Fill();
}
outTree->Write();
```

# Setting object directory

- The directory where a TObject lives can be modified using SetDirectory()
- Argument is generally a TFile to assign object to the file
- SetDirectory(0) disconnects the object from any file
    - Very useful for retrieving objects from a file and then closing the file

# TString

- ROOT provides its own implementation of strings: TString
- TString provides all of the functionality of std::string and more
- Some useful methods:
  - Append() and Prepend()
  - Insert()
  - Replace()
  - Length()
  - First()
- Data() returns a char array - often necessary when passing as an argument

# Histogram errors

- Histograms can hold statistical errors
  - Defined as entry weights added in quadrature
- Use Sumw2() to create structure to hold errors
  - Needs to be called before entries are added
- Error bars are drawn by default
- Draw just the bin contents with:

```
myHist->Draw("HIST");
```

# Histogram scaling

- Uniformly change the integral of a histogram with Scale()

```
myHist->Scale(3); // multiply by 3
```

- Scaling preserves the shape of a histogram
- Scale histograms to have integral = 1 to compare shape of distributions
  - Commonly done when comparing signal and background to choose selection cuts

```
myHist->Scale(1 / myHist->Integral());
```

# Multi-histogram operations

- Numerous ways to interact with multiple histograms simultaneously
- Add two histograms with scale factors:

```
h1.Add(h2,3); // add 3*h2 to h1
h1.Add(h2,1.3,h3,7); // set h1 to be 1.3*h2 + 7*h3
```

- Take the ratio of two histograms:

```
h1.Divide(h2);
```

- Overlay histograms on the same canvas:

```
h2->Draw("same");
```

- Stack histograms using THStack

# pyROOT

- ROOT provides python bindings
- Check supported version of python with root-config --python-version
- Commands are intuitive python implementations of C++ methods

```
import ROOT
h = ROOT.TH1F("myHist", "myTitle", 64, -4, 4)
h.FillRandom("gaus")
h.Integral()
```

- More details available here: https://root.cern/manual/python/

# ROOT Documentation

- Extensive documentation available on ROOT website
  - https://root.cern/manual/basics/ - good starting point
  - https://root.cern/doc/master/ - provides all class definitions
  - https://root.cern/doc/master/group__Tutorials.html - good tutorials
  - https://root-forum.cern.ch/ - ask questions to experts (or find existing questions)
- ROOT naming conventions:
  - Class/namespace and member functions are in UpperCamelCase (a.k.a. PascalCase)
  - Most classes/namespaces begin with T
  - Non-class types end in _t
- When using Google, begin search with “CERN ROOT”
  - ROOT refers to the top level directory in a file system or the name of an admin account