Tutorial 4 More Analysis Tools

This tutorial builds directly on Tutorial 3 A Simple Analysis using Reconstructed Particles, so all the requirements apply.

The Code

```
import hep.physics.jet.EventShape;
import hep.physics.jet.FixNumberOfJetsFinder;
import hep.physics.jet.JetFinder;
import hep.physics.vec.BasicHepLorentzVector;
import hep.physics.vec.HepLorentzVector;
import hep.physics.vec.VecOp;

import java.util.ArrayList;
import java.util.List;

import org.lcsim.event.EventHeader;
import org.lcsim.event.MCParticle;
import org.lcsim.event.ReconstructedParticle;
import org.lcsim.event.util.JetDriver;
import org.lcsim.util.Driver;
import org.lcsim.util.aida.AIDA;

public class SimpleAnalysis extends Driver{
        int ievt = 0;
        private AIDA aida = AIDA.defaultInstance();
        String reconname = "PPRReconParticles";


        public SimpleAnalysis() {

        }

        // process events
        protected void process(EventHeader event) {
                super.process(event);
                // the particle lists
                List<MCParticle> mcparticles = event
                                .get(MCParticle.class, "MCParticle");
                List<ReconstructedParticle> jets = event.get(
                                ReconstructedParticle.class, "Jets");
                List<ReconstructedParticle>recparticles =
                event.get(ReconstructedParticle.class,"PPRReconParticles");




                for (MCParticle particle : mcparticles) {
                         aida.cloud1D("MC particle energy").fill(particle.getEnergy());
                         if ( particle.getGeneratorStatus()==MCParticle.FINAL_STATE) {
                                 aida.cloud1D("MC particle energy final").fill(particle.getEnergy());
                         }
                         if (Math.abs(particle.getPDGID())==22) {
                                 aida.cloud1D("MC particle energy photon").fill(particle.getEnergy());
                         }
                }
                ArrayList<HepLorentzVector> recothrustlist =new ArrayList<HepLorentzVector>();
                for (ReconstructedParticle particle : recparticles) {
                        recothrustlist.add(particle.asFourVector());
                }
                EventShape myrecoshape = new EventShape();
                myrecoshape.setEvent(recothrustlist);
                aida.cloud1D("Rec Particle Thrust").fill(myrecoshape.thrust().x());

                // print out a status line
                if (ievt % 50 == 0) {
                        System.out.println("Processed Events  " + ievt);
                }
                ievt++;
        }
}
```

There are two more advanced schemes build in here , looking at MC particle detailsand using the event shape utilities

MCParticle Analysis

We can query Mc particles for their ID or if they are stable or not

```
 for (MCParticle particle : mcparticles) {
        aida.cloud1D("MC particle energy").fill(particle.getEnergy());
         if ( particle.getGeneratorStatus()==MCParticle.FINAL_STATE) {
                 aida.cloud1D("MC particle energy final").fill(particle.getEnergy());
        }
        if (Math.abs(particle.getPDGID())==22) {
                aida.cloud1D("MC particle energy photon").fill(particle.getEnergy());
        }
}
```

We loop over all particles in the MCParticle collection, with

```
 particle.getGeneratorStatus()==MCParticle.FINAL_STATE)
```

we ask, whether the particle is stable and with

```
 Math.abs(particle.getPDGID())==22
```

we ask whether this particle is a photon (PDG ID=22). all the methods are well documented in the LCIO documentation.

Thrust from the EventShape Package

The EventShape package from FreeHep Libraryalso contains a Thrust implementation, which is used in this example

```
ArrayList<HepLorentzVector> recothrustlist =new ArrayList<HepLorentzVector>();
for (ReconstructedParticle particle : recparticles) {
        recothrustlist.add(particle.asFourVector());
}

EventShape myrecoshape = new EventShape();
myrecoshape.setEvent(recothrustlist);
aida.cloud1D("Rec Particle Thrust").fill(myrecoshape.thrust().x());
```

The EventShape package needs a List of HepLorentzVectors, which we provide by looping over all reconstructed particles

```
 ArrayList<HepLorentzVector> recothrustlist =new ArrayList<HepLorentzVector>();
for (ReconstructedParticle particle : recparticles) {
        recothrustlist.add(particle.asFourVector());
}
```

then we make a new EventShape Object, pass it the ArrayList and plot the Event Thrust (the first component of 3 member array)

```
EventShape myrecoshape = new EventShape();
myrecoshape.setEvent(recothrustlist);
aida.cloud1D("Rec Particle Thrust").fill(myrecoshape.thrust().x());
```