

Tutorial 2 PPFA and Jet Clustering

This is a driver example to run the PPFA from Ron Cassell, cluster some Jets and write and a LCIO file for analysis. The Perfect PFA runs over real GEANT4/SLIC simulated events and uses a perfect pattern recognition, compared to the FastMC tool, which is purely parametric.

Requirements

1. JAS3 installed
2. latest release of org.lcsim and GeomConverter and made available in JAS3 (see [here](#))
3. an Z uds [slcio](#) from the SLAC repository

The Code

```

import java.io.File;

import hep.physics.jet.FixNumberOfJetsFinder;
import hep.physics.jet.JetFinder;

import org.lcsim.contrib.Cassell.recon.Cheat.PPRReconDriver;
import org.lcsim.event.EventHeader;
import org.lcsim.event.util.JetDriver;
import org.lcsim.util.Driver;
import org.lcsim.util.aida.AIDA;
import org.lcsim.util.loop.LCIODriver;

//sample Driver for R.Cassells Perfect PFA
public class PPFADriver extends Driver {

    String reconname = "PPRReconParticles";
    String jetlistname = "Jets";
    private AIDA aida = AIDA.defaultInstance();

    int ievt = 0;

    public PPFADriver() {
        PPRReconDriver rd = new PPRReconDriver();
        add(rd);

        // set up Jet Finder

        JetDriver j = new JetDriver();
        j.setInputCollectionName(reconname);
        j.setOutputCollectionName(jetlistname);
        JetFinder twojet = new FixNumberOfJetsFinder(2);
        j.setFinder(twojet);
        add(j);

        File output = new File(System.getProperty("user.home"), "slicZuds.slcio");

        LCIODriver writer = new LCIODriver(output);
        writer.getWriter().addWriteOnly("MCParticle");
        writer.getWriter().addWriteOnly(jetlistname);
        writer.getWriter().addWriteOnly(reconname);
        add(writer);

    }

    protected void process(EventHeader event) {
        // give some status on events

        super.process(event);

        if (ievt % 50 == 0) {
            System.out.println("Processed Events  " + ievt);
        }
        ievt++;
    }

}

```

The PPFA

The PPFA call is very simple, as everything is nicely encapsulated

```
PPRReconDriver rd = new PPRReconDriver();
    add(rd);
```

that is all that need's doing. Careful, since this cod ein development, there maybe some changes

The JetDriver

The Jet driver is used in the same fashion as in [Tutorial 1 FastSim and Jet Clustering](#)

The LCIO Writer

The LCIO driver use a nice feature, the selctive write of only a few collections, which will be used in Analysis

```
LCIODriver writer = new LCIODriver(output);
    writer.getWriter().addWriteOnly("MCParticle");
    writer.getWriter().addWriteOnly(jetlistname);
    writer.getWriter().addWriteOnly(reconname);
    add(writer);
```

the addWriteOnly method does this for you, and you'll end up with a much smaller LCIO file, which contains merely the MC truth and reconstructed particles