LCFIPlus jet clustering

Matthias Weber (CERN)

LCFIPlus jet clustering

LCFIPlus is supposed to assign vertices and corresponding tracks to jets, clustered with the VLC algorithm → outputs are so-called refined jets with corresponding matched vertices

Idea: use now these refined jets in analysis, compare with original subjet,

Potential issue: realized energy changes sometimes quite substantially

Printout of all particles of one of these refined jets:

```
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 0 E/px/py/pz 6.04565/4.197/-1.82904/3.94591 PDG 211
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 1 E/px/py/pz 12.9354/8.53676/-3.65458/9.00517 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 2 E/px/py/pz 3.32803/2.15205/-0.991208/2.33708 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 3 E/px/py/pz 2.27612/1.72994/-0.333248/1.44118 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 4 E/px/py/pz 0.881595/0.600208/-0.136829/0.631061 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 5 E/px/py/pz 11.1868/7.68948/-3.20962/7.46292 PDG 211
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 6 E/px/py/pz 11.1868/7.68948/-3.20962/7.46292 PDG 211
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 7 E/px/py/pz 4.42074/2.90832/-1.44602/2.99569 PDG -211
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 8 E/px/py/pz 100.277/67.24/-29.9308/68.1064 PDG -13
[ VERBOSE "MyHZAnalyzer"] jet E 252.816 particles in sj1_rfj1 9 E/px/py/pz 100.277/67.24/-29.9308/68.1064 PDG -13
```

Charged particles in the jet are at times double counted → for this jet almost double the total jet energy, so issue not only in assignment of particles to jets

Behavior tested on all builds of ILCSoft from April

First look: assignment of particles to jets

Issue is related only to charged particles

→ In fact issue related to tracks from vertices, see code at https://github.com/lcfiplus/LCFIPlus/blob/master/src/LCIOStorer.cc#L982-L1010

```
// associate particles
double charge = 0.;
for (unsigned int ntr = 0; ntr < flajet->getTracks().size(); ntr++) {
  const lcfiplus::Track* flatr = flajet->getTracks()[ntr];
  lcio::ReconstructedParticle* lciotr = _trackLCIORel[const_cast<lcfiplus::Track*>(flatr)];
  charge += flatr->getCharge();
  lciojet->addParticle(lciotr);
  //cout << "LCIOStorer::ConvertJet: add track: id = " << flatr->getId() << ", energy = " << flatr->E() << flush;
  //cout << ", lcio energy = " << lciotr->getEnergy() << endl;
}
for (unsigned int nneut = 0; nneut < flajet->getNeutrals().size(); nneut++) {
  const lcfiplus::Neutral* flaneut = flajet->getNeutrals()[nneut];
  lcio::ReconstructedParticle* lcioneut = _neutralLCIORel[const_cast<lcfiplus::Neutral*>(flaneut)];
  lciojet->addParticle(lcioneut);
}
for (unsigned int nvtx = 0; nvtx < flajet->getVertices().size(); nvtx++) {
  const lcfiplus::Vertex* flavtx = flajet->getVertices()[nvtx];
  // first, extract all vertex tracks
  for (unsigned int ntr = 0; ntr < flavtx->getTracks().size(); ntr++) {
    const lcfiplus::Track* flatr = flavtx->getTracks()[ntr];
    lcio::ReconstructedParticle* lciotr = _trackLCIORel[const_cast<lcfiplus::Track*>(flatr)];
    charge += flatr->getCharge();
    lciojet->addParticle(lciotr);
  }
```

If track assigned to vertex, it is double counted

Fix in LCFIPlus prerun function of JetFinder

In prerun tracks and neutrals are checked for association to a vertex jets. If the association criteria is met, tracks are assigned to the jet. So far no check was performed if the track is in fact part of the vertex

```
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 0 E/px/py/pz 6.04565/4.197/-1.82904/3.94591 PDG 211
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 1 E/px/py/pz 12.9354/8.53676/-3.65458/9.00517 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 2 E/px/py/pz 3.32803/2.15205/-0.991208/2.33708 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 3 E/px/py/pz 2.27612/1.72994/-0.333248/1.44118 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 4 E/px/py/pz 0.881595/0.600208/-0.136829/0.631061 PDG 22
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 5 E/px/py/pz 100.277/67.24/-29.9308/68.1064 PDG -13
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 6 E/px/py/pz 11.1868/7.68948/-3.20962/7.46292 PDG 211
[ VERBOSE "MyHZAnalyzer"] jet E 141.352 particles in sj1_rfj1 7 E/px/py/pz 4.42074/2.90832/-1.44602/2.99569 PDG -211
```

Tracks assigned only once, jet energy changes naturally too

→ Might have an impact on B-tagging as well, since for some categories properties of the jet used as well together with vertex information

Fix in JetFinder file: prerun function

Before associating track to vertex jet, check if track already part of the vertex
→ If track already counted as part of the vertex, don't add it (avoids double counting)

14 src/JetFinder.cc

```
@@ -367,7 +367,19 @@ vector<Jet*> JetFinder::prerun(TrackVec& tracks, NeutralVec& neutrals, VertexVec
      }

      if (jetToAssoc) {
-       jetToAssoc->add(tracks[i]);
+       bool veto_track=false;
+       for (unsigned int k=0; k<jetToAssoc->getVertices().size(); k++) {
+         const Vertex* vtx = jetToAssoc->getVertices()[k];
+         for (unsigned int n=0;n<vtx->getTracks().size();n++){
+           if(vtx->getTracks()[n]->Angle(tracks[i]->Vect())==0){
+             veto_track=true;
+             break;
+           }
+         }
+       }
+       if(!veto_track){
+         jetToAssoc->add(tracks[i]);
+       }
        usedTracks[i] = true;
      }
    }
```