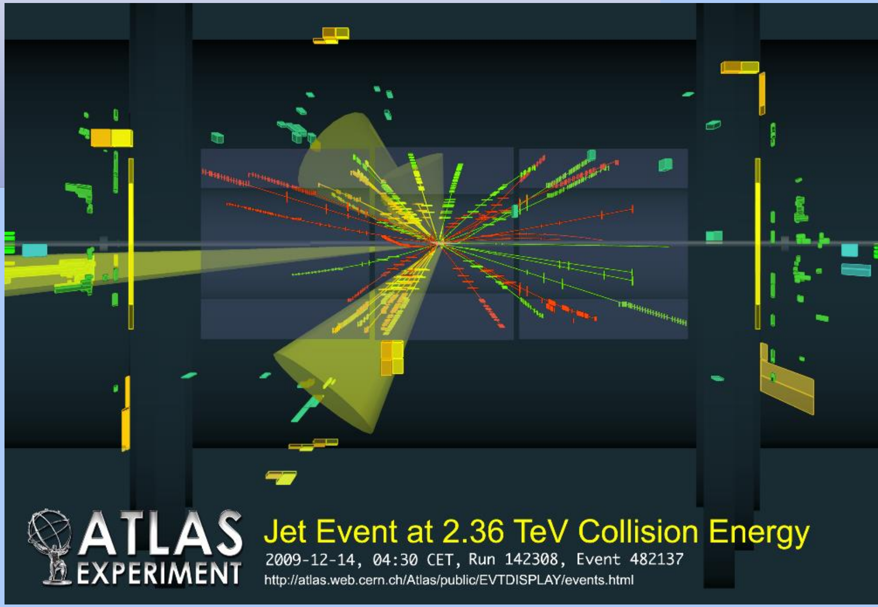
Boosted X → HH → 4b analysis

Christina Nelson

February 12, 2015

Project Introduction & Overview

Overview, what & why…


ATLAS EXPERIMENT
Jet Event at 2.36 TeV Collision Energy
2009-12-14, 04:30 CET, Run 142308, Event 482137
http://atlas.web.cern.ch/Atlas/public/EVTDISPLAY/events.html

* What is X -> HH -> 4b ?

* Resolved & Boosted Analysis

* How data is handled and interpreted

* EventLoop algorithm with RootCore implementation

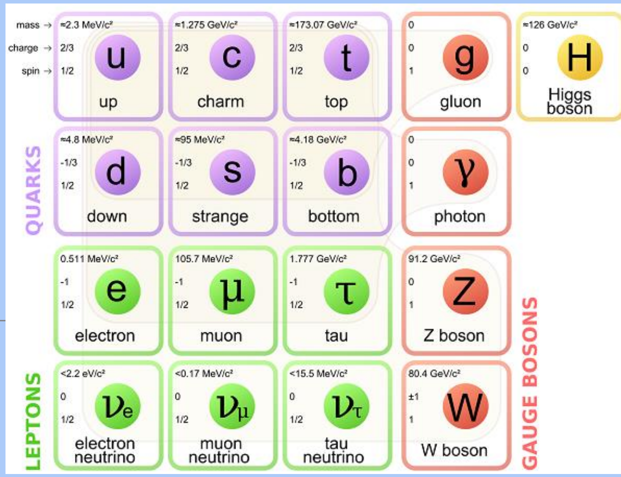
* Example code working-on

* SVN package

* Goal

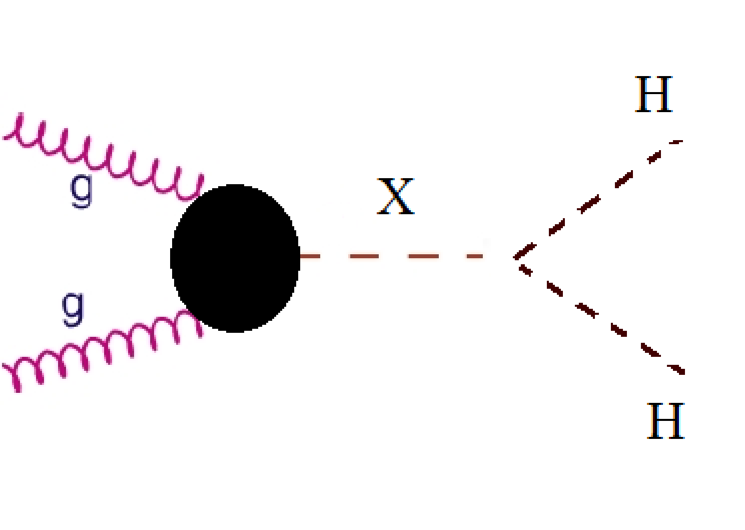
* Preliminary acknowledgement

What is X → HH → 4b ?



Higgs boson has been found and measured properties are consistent with the Standard Model. Many new physics models predict significant Higgs pair production rates at high invariant mass.

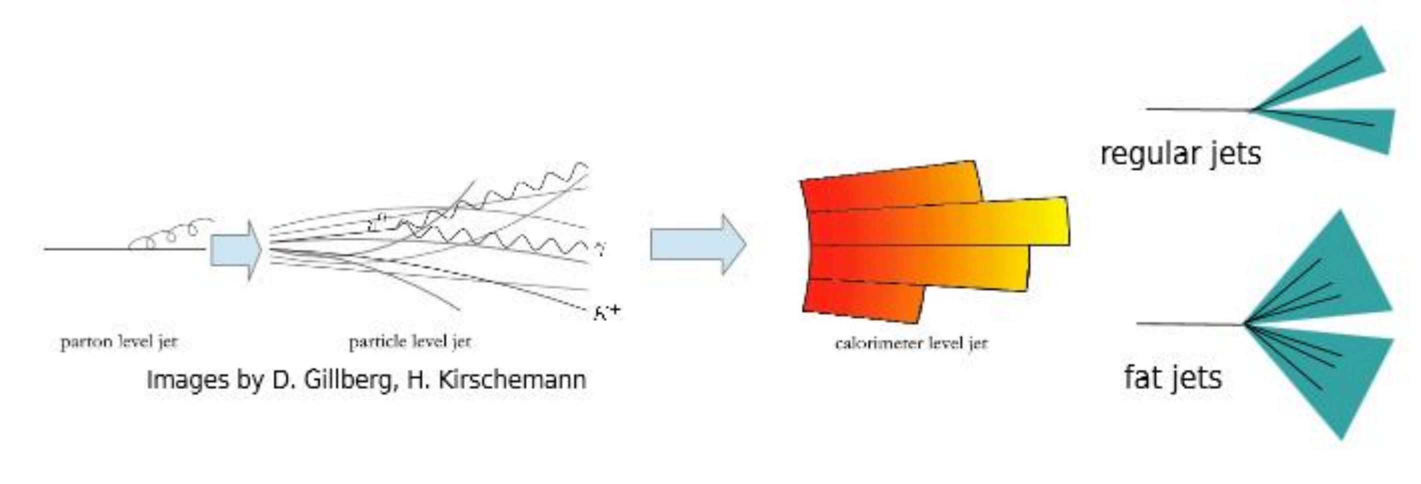
- Owing to the nature of QCD, quarks are never observed as free particles, but are always found confined within hadrons.
- However, in high energy collisions it is quarks that are produced, not hadrons.
- As the result of QCD interaction, the strong interaction field between the quarks produce further quarks and antiquarks through a process called hadronization.



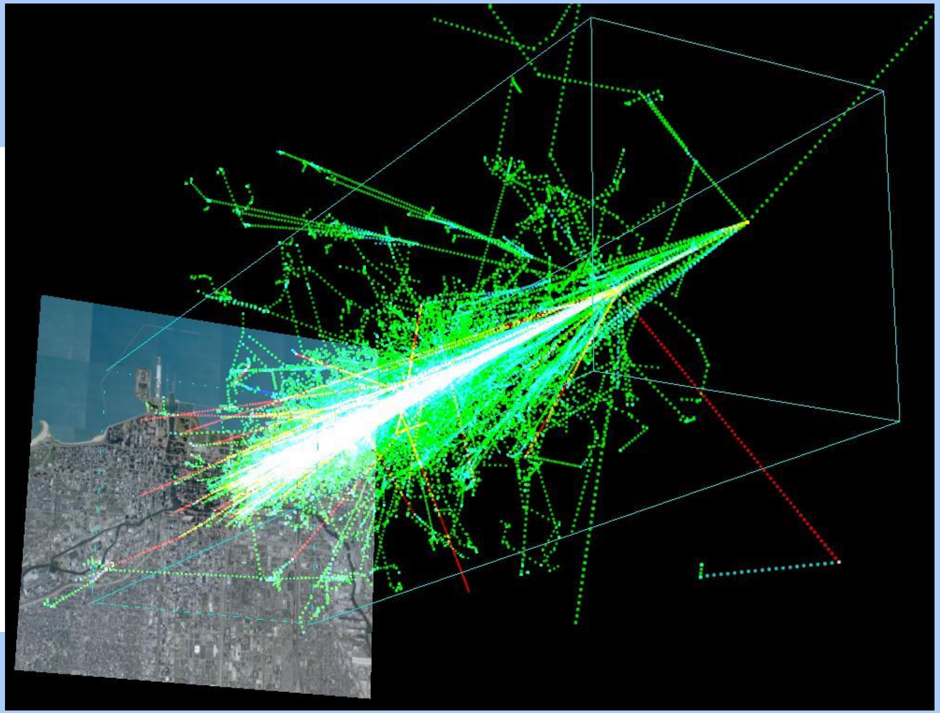
- From hadronization, each quark produced in a collision produces a jet of hadrons
- In general, it is not possible to tell which flavor quark was produced, or even whether the jet originated from a quark or gluon.
- However, if a b-quark is produced, the hadronization process will create a jet of hadrons, one of which will contain the b-quark.
- The b quarks are relatively long lived with lifetimes of order 1.5 x 10^-12 s

- New resonances : Kaluza-Klein graviton in Randall-Sundrum framework
- Extended Higgs sectors : 2 Higgs Decay Model (2DHM)

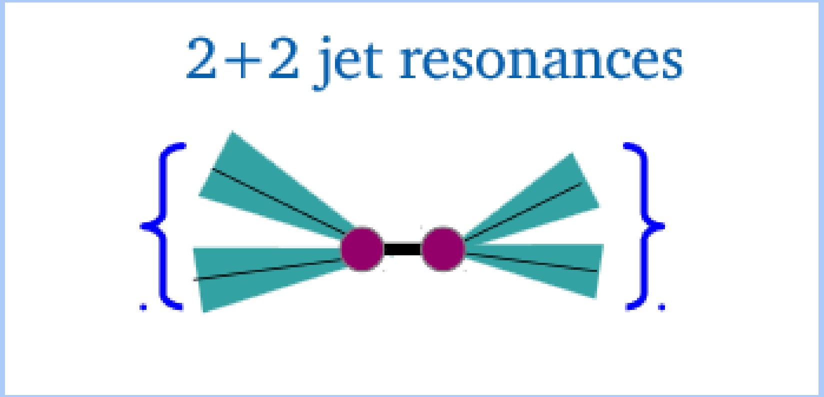
Resolved &Boosted Analysis



Images by D. Gillberg, H. Kirschemann

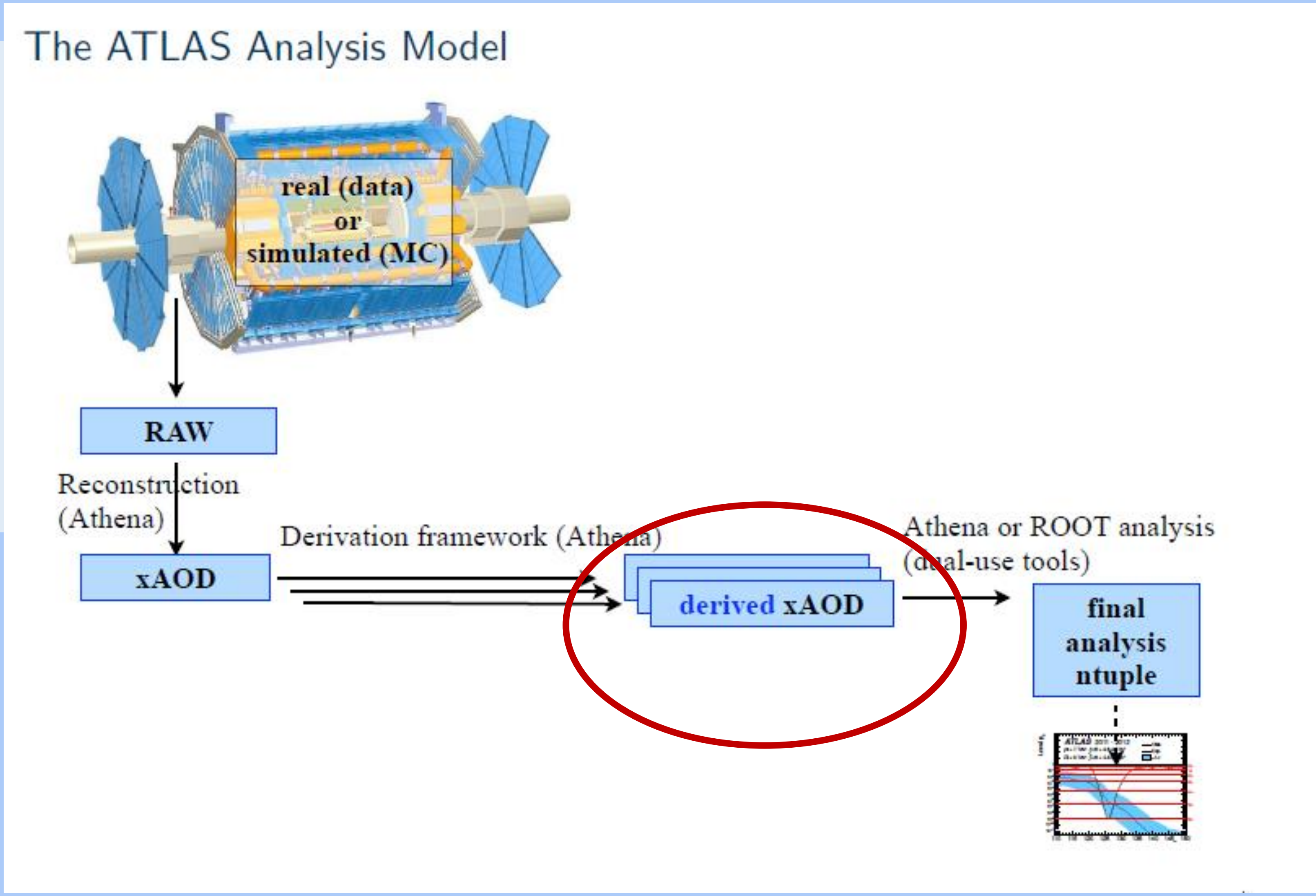


- The identification of b-quark jets relies on the ability to resolve the secondary vertices from the primary vertex.
- Resolved analysis reconstructs Higgs boson candidates from pairs of anti-k_T R = 0.4 jets that are each b-tagged, and offers good efficiency over a wide range of P_T. Sensitivity is particularly good in the range 500 <= m_x <= 1500 GeV.
- Boosted analysis reconstructs as a single, trimmed anti-k_T R = 1.0 jet which must be associated with two b-tagged anti-k_T R = 0.3 track-jets.
- The use of a smaller R parameter track jets allows for higher P_T Higgs bosons to be reconstructed.



How data is handled and interpreted

- Event data
  - * Data : detector read out, RAW
  - * reconstructed quantities
  - * stored in event files
  - * MC : Monte Carlo generators and simulated detector responses
- Meta data
  - * “Data about the data”
  - * Configuration and conditions
  - * stored in event files or databases



The ATLAS Analysis Model

real (data) or simulated (MC)

RAW

Reconstruction (Athena)

xAOD

Derivation framework (Athena)

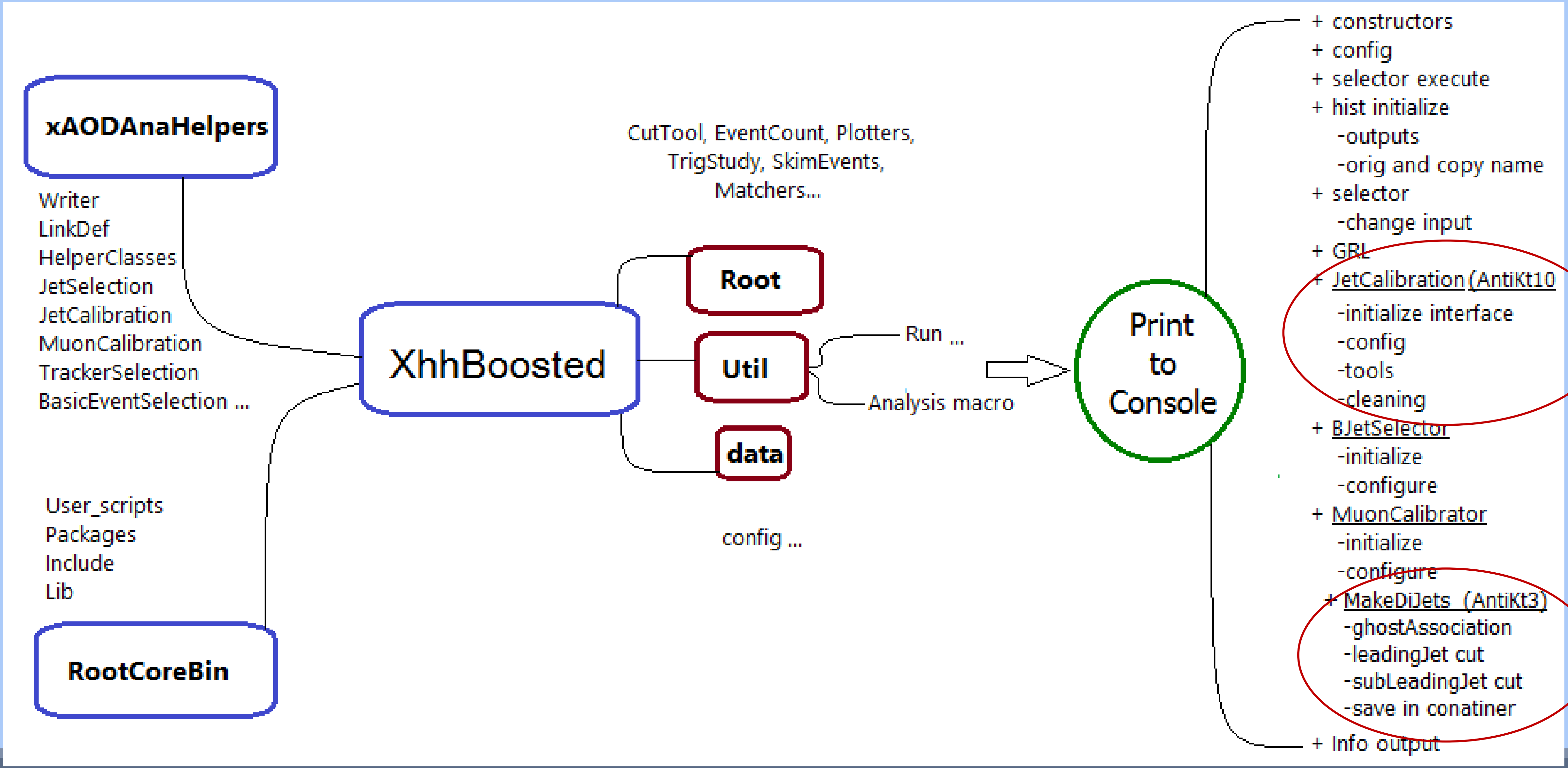
derived xAOD

Athena or ROOT analysis (dual-use tools)

final analysis ntuple

- xAOD = Analysis Object Data
- Athena = reconstruction software
- Data Challenge 2014 (DC14)
- Reprocessing of Run-I data and MC with new xAOD EDM
- Run-II MC with $sqrt{s}$ 13TeV

EventLoop algorithm with RootCore implementation



xAODAnaHelpers

Writer
LinkDef
HelperClasses
JetSelection
JetCalibration
MuonCalibration
TrackerSelection
BasicEventSelection ...

User_scripts
Packages
Include
Lib

RootCoreBin

XhhBoosted

CutTool, EventCount, Plotters,
TrigStudy, SkimEvents,
Matchers...

Root

Util

data

config ...

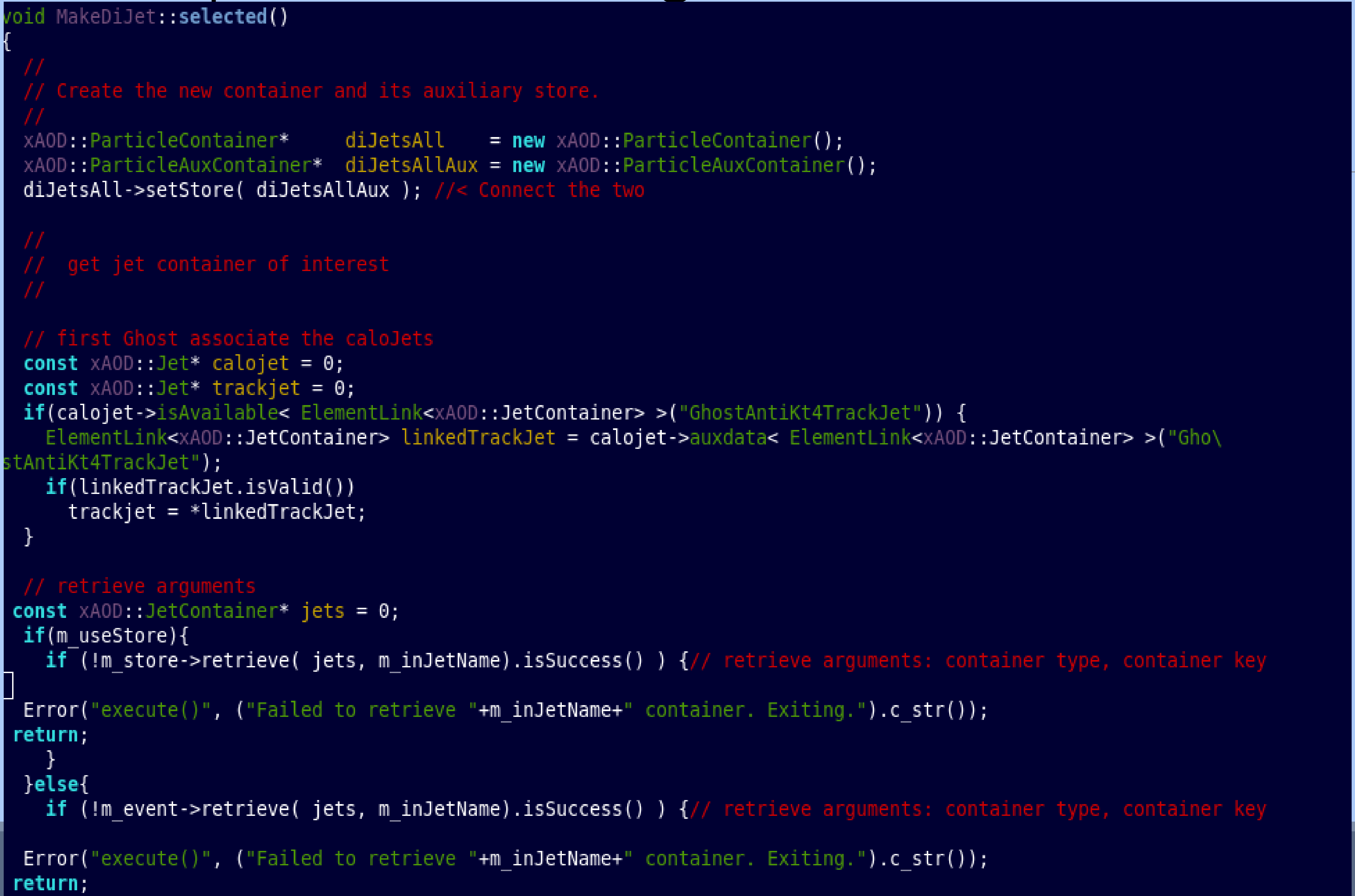
Run ...

Analysis macro

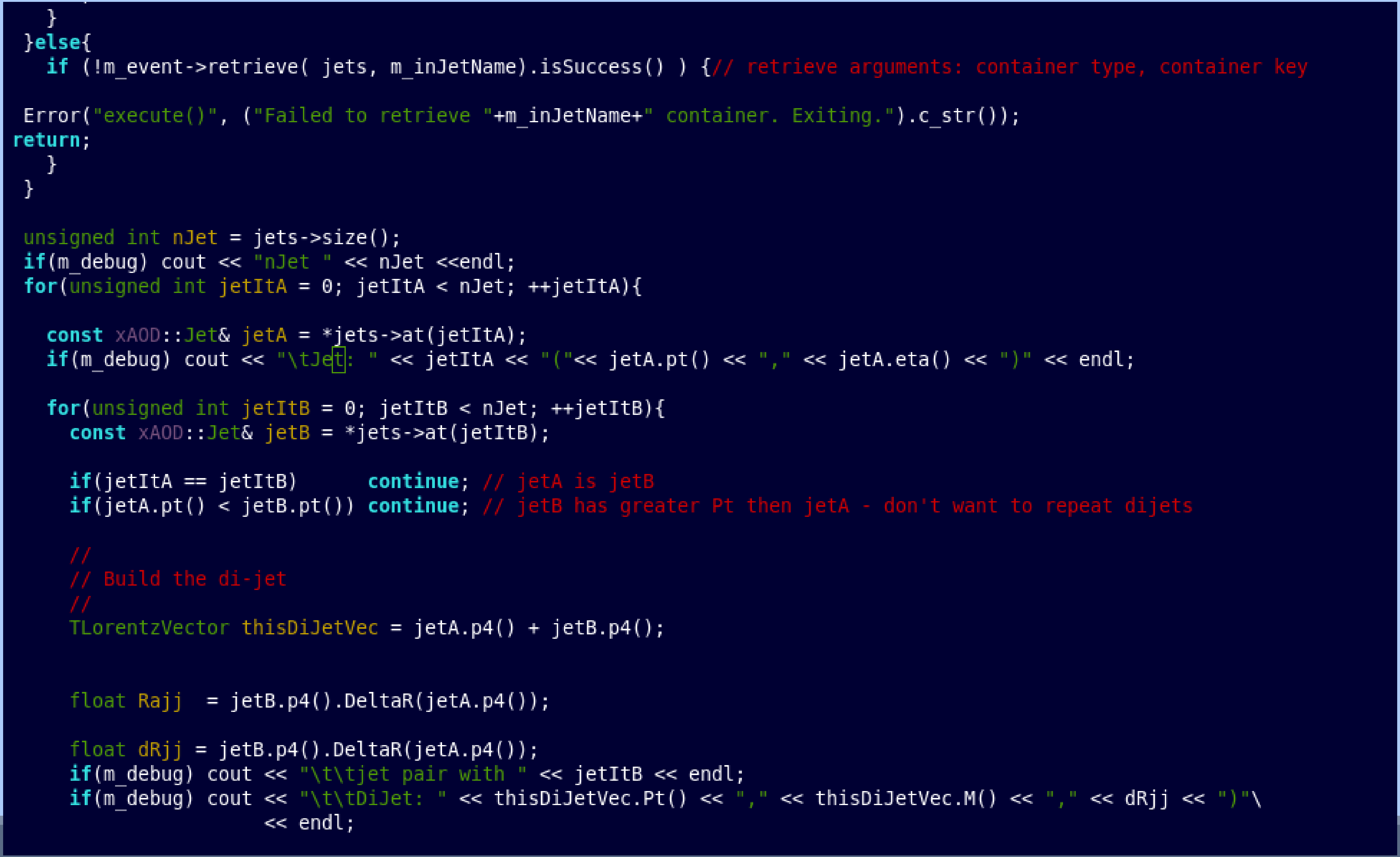
Print
to
Console

+ constructors
+ config
+ selector execute
+ hist initialize
  -outputs
  -orig and copy name
+ selector
  -change input
+ GRL
+ JetCalibration (AntiKt10
  -initialize interface
  -config
  -tools
  -cleaning
+ BJetSelector
  -initialize
  -configure
+ MuonCalibrator
  -initialize
  -configure
+ MakeDiJets (AntiKt3)
  -ghostAssociation
  -leadingJet cut
  -subLeadingJet cut
  -save in conatiner
+ Info output

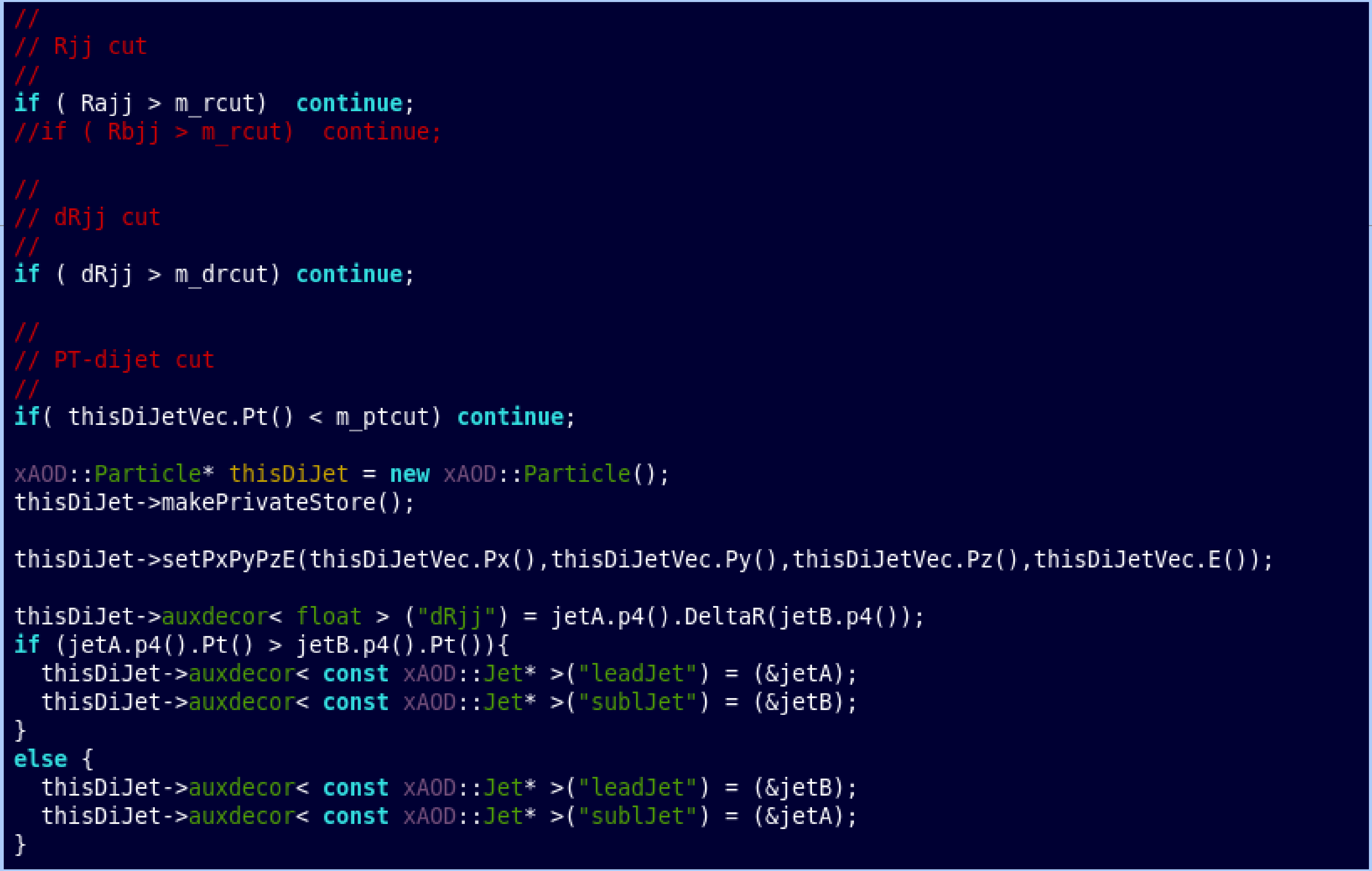
Example Code working-on



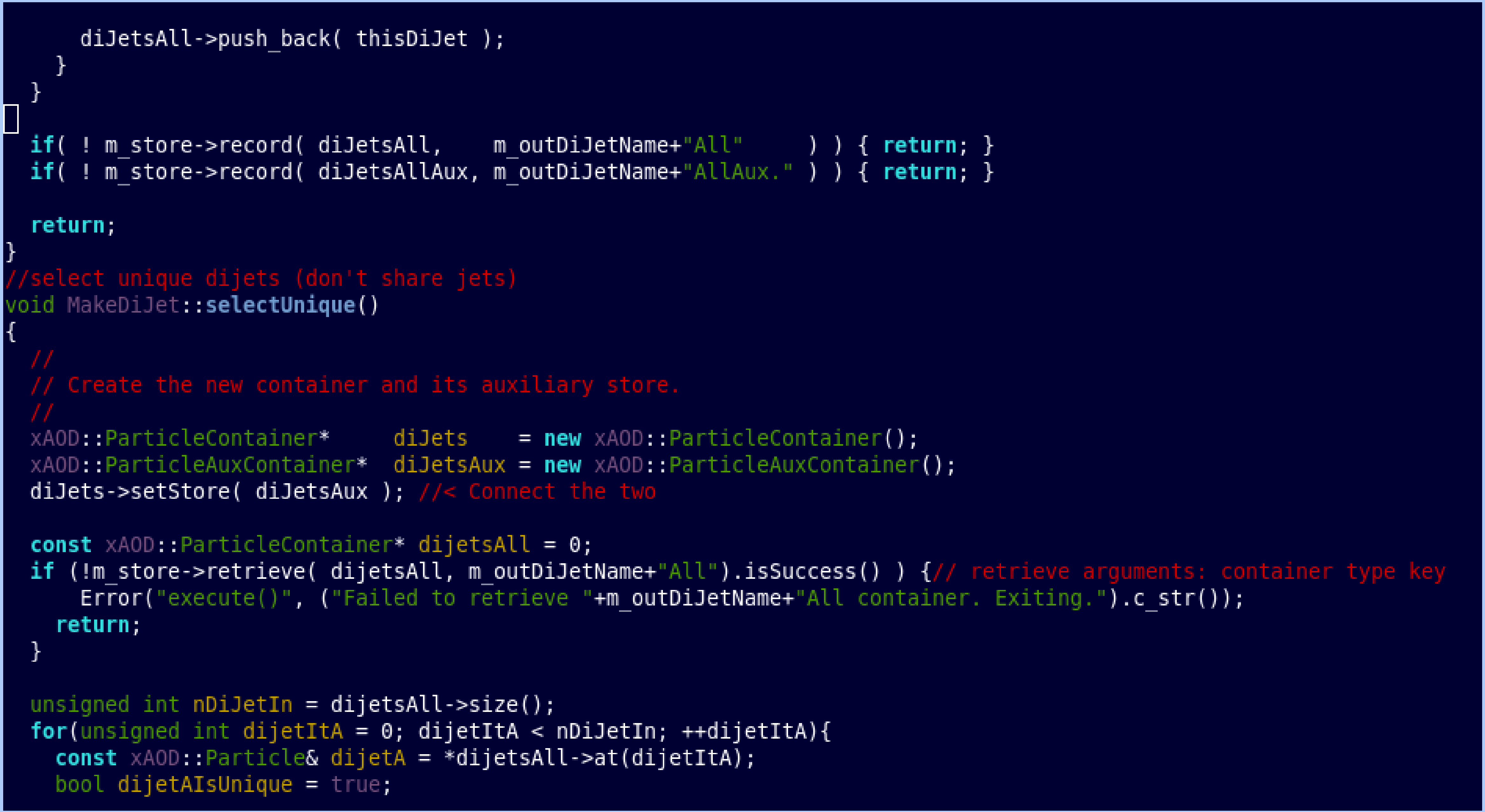
```
void MakeDiJet::selected()
{
  //
  // Create the new container and its auxiliary store.
  //
  xAOD::ParticleContainer*     diJetsAll    = new xAOD::ParticleContainer();
  xAOD::ParticleAuxContainer*  diJetsAllAux = new xAOD::ParticleAuxContainer();
  diJetsAll->setStore( diJetsAllAux ); //< Connect the two

  //
  //  get jet container of interest
  //

  // first Ghost associate the caloJets
  const xAOD::Jet* calojet = 0;
  const xAOD::Jet* trackjet = 0;
  if(calojet->isAvailable< ElementLink<xAOD::JetContainer> >("GhostAntiKt4TrackJet")) {
    ElementLink<xAOD::JetContainer> linkedTrackJet = calojet->auxdata< ElementLink<xAOD::JetContainer> >("Gho\
stAntiKt4TrackJet");
    if(linkedTrackJet.isValid())
      trackjet = *linkedTrackJet;
  }

  // retrieve arguments
 const xAOD::JetContainer* jets = 0;
  if(m_useStore){
    if (!m_store->retrieve( jets, m_inJetName).isSuccess() ) {// retrieve arguments: container type, container key

  Error("execute()", ("Failed to retrieve "+m_inJetName+" container. Exiting.").c_str());
 return;
    }
  }else{
    if (!m_event->retrieve( jets, m_inJetName).isSuccess() ) {// retrieve arguments: container type, container key

  Error("execute()", ("Failed to retrieve "+m_inJetName+" container. Exiting.").c_str());
 return;
```



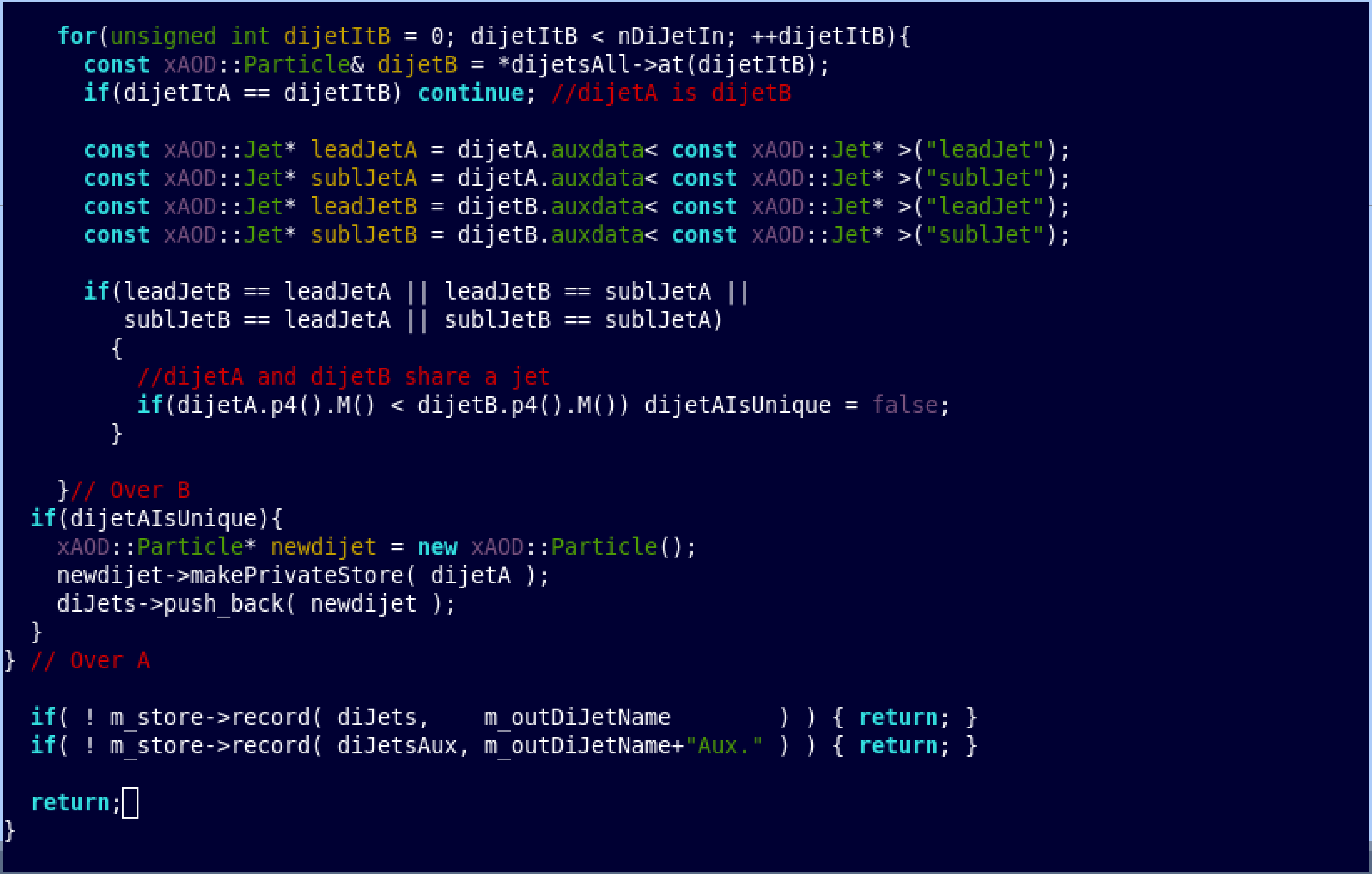
```
   }
 }else{
   if (!m_event->retrieve( jets, m_inJetName).isSuccess() ) {// retrieve arguments: container type, container key

 Error("execute()", ("Failed to retrieve "+m_inJetName+" container. Exiting.").c_str());
return;
   }
 }

 unsigned int nJet = jets->size();
 if(m_debug) cout << "nJet " << nJet <<endl;
 for(unsigned int jetItA = 0; jetItA < nJet; ++jetItA){

   const xAOD::Jet& jetA = *jets->at(jetItA);
   if(m_debug) cout << "\tJet: " << jetItA << "("<< jetA.pt() << "," << jetA.eta() << ")" << endl;

   for(unsigned int jetItB = 0; jetItB < nJet; ++jetItB){
     const xAOD::Jet& jetB = *jets->at(jetItB);

     if(jetItA == jetItB)      continue; // jetA is jetB
     if(jetA.pt() < jetB.pt()) continue; // jetB has greater Pt then jetA - don't want to repeat dijets

     //
     // Build the di-jet
     //
     TLorentzVector thisDiJetVec = jetA.p4() + jetB.p4();


     float Rajj  = jetB.p4().DeltaR(jetA.p4());

     float dRjj = jetB.p4().DeltaR(jetA.p4());
     if(m_debug) cout << "\t\tjet pair with " << jetItB << endl;
     if(m_debug) cout << "\t\tDiJet: " << thisDiJetVec.Pt() << "," << thisDiJetVec.M() << "," << dRjj << ")"\
                      << endl;
```



```
//
// Rjj cut
//
if ( Rajj > m_rcut)  continue;
//if ( Rbjj > m_rcut)  continue;

//
// dRjj cut
//
if ( dRjj > m_drcut) continue;

//
// PT-dijet cut
//
if( thisDiJetVec.Pt() < m_ptcut) continue;

xAOD::Particle* thisDiJet = new xAOD::Particle();
thisDiJet->makePrivateStore();

thisDiJet->setPxPyPzE(thisDiJetVec.Px(),thisDiJetVec.Py(),thisDiJetVec.Pz(),thisDiJetVec.E());

thisDiJet->auxdecor< float > ("dRjj") = jetA.p4().DeltaR(jetB.p4());
if (jetA.p4().Pt() > jetB.p4().Pt()){
  thisDiJet->auxdecor< const xAOD::Jet* >("leadJet") = (&jetA);
  thisDiJet->auxdecor< const xAOD::Jet* >("sublJet") = (&jetB);
}
else {
  thisDiJet->auxdecor< const xAOD::Jet* >("leadJet") = (&jetB);
  thisDiJet->auxdecor< const xAOD::Jet* >("sublJet") = (&jetA);
}
```

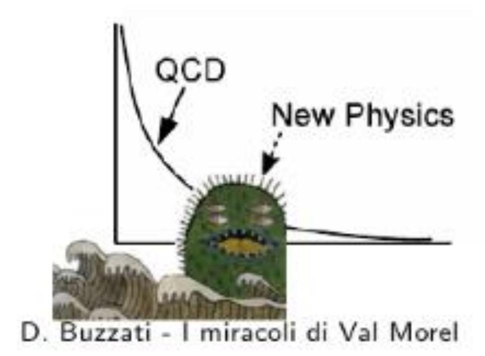


```
      diJetsAll->push_back( thisDiJet );
    }
  }

  if( ! m_store->record( diJetsAll,    m_outDiJetName+"All"     ) ) { return; }
  if( ! m_store->record( diJetsAllAux, m_outDiJetName+"AllAux." ) ) { return; }

  return;
}
//select unique dijets (don't share jets)
void MakeDiJet::selectUnique()
{
  //
  // Create the new container and its auxiliary store.
  //
  xAOD::ParticleContainer*     diJets    = new xAOD::ParticleContainer();
  xAOD::ParticleAuxContainer*  diJetsAux = new xAOD::ParticleAuxContainer();
  diJets->setStore( diJetsAux ); //< Connect the two

  const xAOD::ParticleContainer* dijetsAll = 0;
  if (!m_store->retrieve( dijetsAll, m_outDiJetName+"All").isSuccess() ) {// retrieve arguments: container type key
      Error("execute()", ("Failed to retrieve "+m_outDiJetName+"All container. Exiting.").c_str());
    return;
  }

  unsigned int nDiJetIn = dijetsAll->size();
  for(unsigned int dijetItA = 0; dijetItA < nDiJetIn; ++dijetItA){
    const xAOD::Particle& dijetA = *dijetsAll->at(dijetItA);
    bool dijetAIsUnique = true;
```



```
    for(unsigned int dijetItB = 0; dijetItB < nDiJetIn; ++dijetItB){
      const xAOD::Particle& dijetB = *dijetsAll->at(dijetItB);
      if(dijetItA == dijetItB) continue; //dijetA is dijetB

      const xAOD::Jet* leadJetA = dijetA.auxdata< const xAOD::Jet* >("leadJet");
      const xAOD::Jet* sublJetA = dijetA.auxdata< const xAOD::Jet* >("sublJet");
      const xAOD::Jet* leadJetB = dijetB.auxdata< const xAOD::Jet* >("leadJet");
      const xAOD::Jet* sublJetB = dijetB.auxdata< const xAOD::Jet* >("sublJet");

      if(leadJetB == leadJetA || leadJetB == sublJetA ||
         sublJetB == leadJetA || sublJetB == sublJetA)
        {
          //dijetA and dijetB share a jet
          if(dijetA.p4().M() < dijetB.p4().M()) dijetAIsUnique = false;
        }

    }// Over B
  if(dijetAIsUnique){
    xAOD::Particle* newdijet = new xAOD::Particle();
    newdijet->makePrivateStore( dijetA );
    diJets->push_back( newdijet );
  }
} // Over A

  if( ! m_store->record( diJets,    m_outDiJetName        ) ) { return; }
  if( ! m_store->record( diJetsAux, m_outDiJetName+"Aux." ) ) { return; }

  return;
}
```

SVN package : XhhBoosted



* SVN stands for a subversion repository that facilitates collaborative development of software and documents.

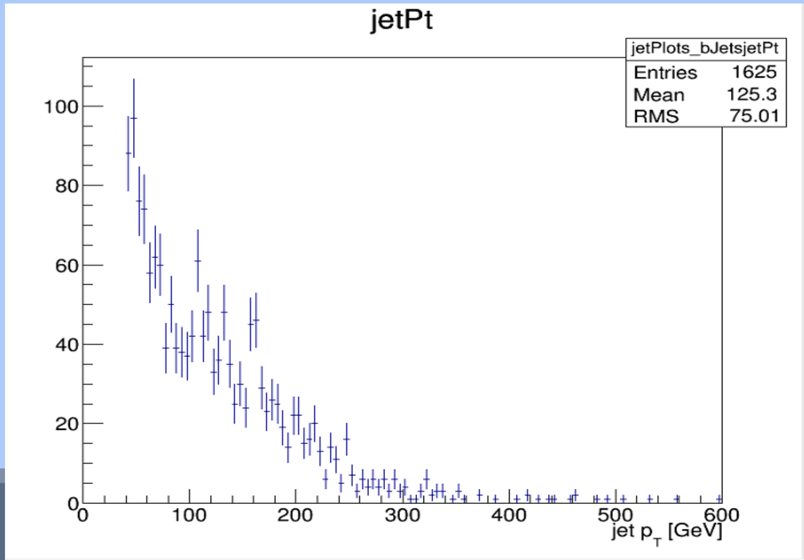
* developing package: XhhBoosted

* can be found through svnweb.cern.ch

root/Institutes/Uchicago/johnda/EventLoopsAlgs/XhhBoosted/trunk

- Look for bumps over QCD
- Key to analysis: good control of calorimeter behavior

and jet performance at high energies → jet reconstruction



Goals

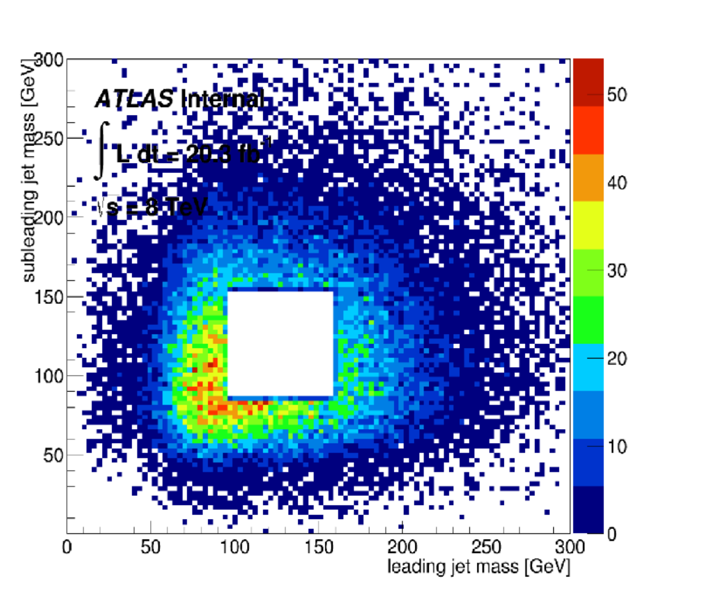
*** Have a working XhhBoosted package ready for analysis with DC14 at 13TeV in preparation for Run2. (asap, hopefully end of the week or next week)**

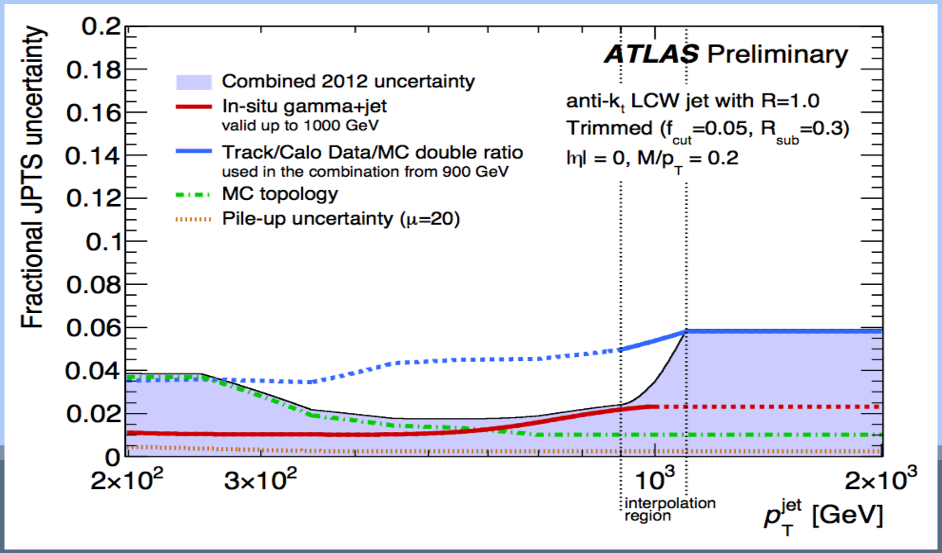
*** Generate xAOD analysis in ROOT with DC14 at 13TeV**

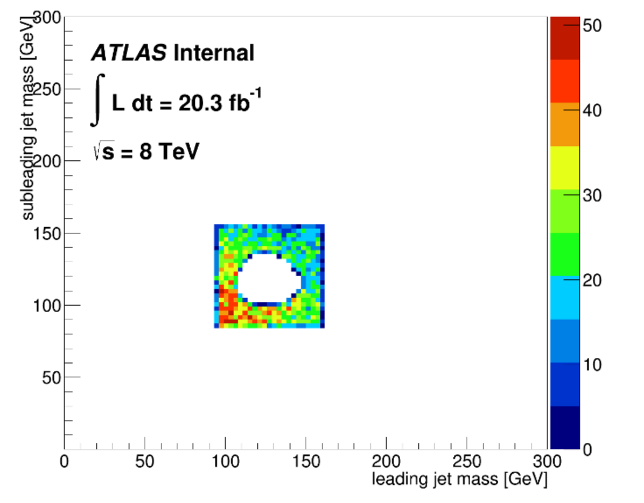
- Background estimation : define sideband, signal and control regions
- Use control region to predict QCD and ttbar
- Study systematics & uncertainties

* Write up analysis

* Improve on data processing if possible







Preliminary Acknowledgment to :

The Lounsbery Foundation, & Professor Krisch & Professor Neal

canelson@hawaii.edu

christina.nelson@cern.ch

References

https://indico.cern.ch/event/295317/contribution/5/material/slides/0.pdf

Caterina Doglioni : Jet + X searcher for Run II, 06/ 01/ 2015

https://indico.cern.ch/event/345626/session/8/contribution/26/material/slides/0.pdf

Christian Ohm : Data Preparation & Physics Analysis, 19/01/2015

https://indico.cern.ch/event/345626/session/7/contribution/24/material/slides/0.pdf

Louise Heelan : ATLAS Analysis Model: How analysis is done, 19/01/2015

M. Bellomo, M. Kagan, E. Thompson, L. Zhou : Boosted X → HH → 4b Channel, 27/ 01 /2015

Mark Thompson, Modern Particle Physics. Cambridge Press 2013, pp 21-25