```
// -*- C++ -*-
//
// Package:    PromptAna/PromptAnalyzer
// Class:      PromptAnalyzer
//
/**\class PromptAnalyzer PromptAnalyzer.cc PromptAna/PromptAnalyzer/plugins/PromptAnalyzer.cc

 Description: [one line class summary]

 Implementation:
     [Notes on implementation]
*/
//
// Original Author:  Robert Ciesielski
//         Created:  Wed, 27 Jun 2018 16:18:44 GMT
//        Modified:  Luiz Emediato - 24 Nov 2020
//
// system include files
#include <memory>
#include <stdlib.h>
#include <iostream>
#include <string>
#include <sstream>
#include <istream>
#include <fstream>
#include <iomanip>
#include <map>

// user include files
#include "FWCore/Framework/interface/Frameworkfwd.h"
#include "FWCore/Framework/interface/one/EDAnalyzer.h"

#include "FWCore/Framework/interface/Event.h"
#include "FWCore/Framework/interface/EventSetup.h"
#include "FWCore/Framework/interface/MakerMacros.h"

#include "FWCore/ParameterSet/interface/ParameterSet.h"
#include "FWCore/Utilities/interface/InputTag.h"
#include "DataFormats/TrackReco/interface/Track.h"
#include "DataFormats/TrackReco/interface/TrackFwd.h"
#include "DataFormats/Candidate/interface/Candidate.h"
// dEdx
#include "DataFormats/TrackReco/interface/DeDxHit.h"
#include "DataFormats/TrackReco/interface/DeDxHitInfo.h"
#include "DataFormats/TrackReco/interface/DeDxData.h"

#include "DataFormats/VertexReco/interface/Vertex.h"
#include "DataFormats/VertexReco/interface/VertexFwd.h"
#include "SimDataFormats/Vertex/interface/SimVertex.h"

#include "DataFormats/BeamSpot/interface/BeamSpot.h"

#include "DataFormats/HepMCCandidate/interface/GenParticle.h"
#include "DataFormats/HepMCCandidate/interface/GenParticleFwd.h"

#include "FWCore/ServiceRegistry/interface/Service.h"
#include "CommonTools/UtilAlgos/interface/TFileService.h"
#include "TH1.h"
#include "TH2.h"
#include "TMath.h"
#include "TLorentzVector.h"
//#include "DataFormats/Math/interface/deltaR.h"

// pixel clusters
//#include "DataFormats/SiPixelCluster/interface/SiPixelCluster.h"
//#include "DataFormats/TrackerCommon/interface/TrackerTopology.h"
//#include "Geometry/Records/interface/TrackerTopologyRcd.h"

#include "DataFormats/Common/interface/TriggerResults.h"
#include "DataFormats/HLTReco/interface/TriggerObject.h"
#include "FWCore/Common/interface/TriggerNames.h"
#include "DataFormats/HLTReco/interface/TriggerEvent.h"
```

```
#include "HLTrigger/HLTcore/interface/HLTConfigProvider.h"
#include "HLTrigger/HLTcore/interface/HLTPrescaleProvider.h"

//PPS
#include "DataFormats/FWLite/interface/Handle.h"
#include "DataFormats/FWLite/interface/ChainEvent.h"

#include "DataFormats/CTPPSDetId/interface/TotemRPDetId.h"
#include "DataFormats/CTPPSReco/interface/CTPPSLocalTrackLite.h"

// PFCandidates
#include "DataFormats/ParticleFlowCandidate/interface/PFCandidate.h"
#include "DataFormats/ParticleFlowCandidate/interface/PFCandidateFwd.h"
// V0Producer: PFCandidate.h is enough ...Luiz
//#include "DataFormats/Candidate/interface/VertexCompositeCandidate.h"
//#include "DataFormats/Candidate/interface/VertexCompositeCandidateFwd.h"

// Muon Includes
#include "DataFormats/MuonReco/interface/MuonFwd.h"
#include "DataFormats/MuonReco/interface/Muon.h"
#include "DataFormats/MuonReco/interface/MuonSelectors.h"

// Ferenc's PID
#include "UserCode/EnergyLossPID/interface/ParticleType.h"

// Forward Protons ...Luiz
//#include "DataFormats/ProtonReco/interface/ForwardProton.h"
//#include "DataFormats/ProtonReco/interface/ForwardProtonFwd.h"

#define M_LN10 2.30258509299404568402
#define Sqr(x) ((x) * (x))

//------------------------------

// optics
const double v_x_R_1_F = -2.24791387053766;  const double L_x_R_1_F = 0.125396407127792E3;
const double v_y_R_1_F = +0.025781593410852; const double L_y_R_1_F = 238.517247191010E3;
const double v_x_R_2_F = -1.92610996810677;  const double L_x_R_2_F = -3.00655323980445E3;
const double v_y_R_2_F = -0.000000021508565; const double L_y_R_2_F = 271.511335947517E3;

const double v_x_L_1_F = -2.24791387053766;  const double L_x_L_1_F = 0.125396407127792E3;
const double v_y_L_1_F = +0.025781593410852; const double L_y_L_1_F = 238.517247191010E3;
const double v_x_L_2_F = -1.92610996810677;  const double L_x_L_2_F = -3.00655323980445E3;
const double v_y_L_2_F = -0.000000021508565; const double L_y_L_2_F = 271.511335947517E3;


//double m_pi=0.13957;
double m_k =0.493677;
//...Luiz
double m_k0 =0.497611;
//double m_mu = 0.1056583715;
//double m_p =0.93827;
double m_rho = 0.77;
double m_phi = 1.019461;

//...Luiz
// new PDG
double m_pi=0.13957061;
// new PDG
double m_mu = 0.1056583745;
double m_e = 0.0005109989461;
double m_p = 0.9382720813;

//...using Ferenc's PID: pion now is 2, kaon is 3 ...Luiz
//enum EPID { pidUnknown, pidProton, pidKaon, pidPion };
//
//.............0..........1.........2........3
enum EPID { pidUnknown, pidProton, pidPion, pidKaon };

//
// class declaration
//
```

```
// If the analyzer does not use TFileService, please remove
// the template argument to the base class so the class inherits
// from  edm::one::EDAnalyzer<>
// This will improve performance in multithreaded jobs.


//using reco::TrackCollection;

using namespace edm;
using namespace reco;
using namespace std;

class PromptAnalyzer : public edm::one::EDAnalyzer<edm::one::SharedResources>  {
   public:
      explicit PromptAnalyzer(const edm::ParameterSet&);
      ~PromptAnalyzer();

      static void fillDescriptions(edm::ConfigurationDescriptions& descriptions);


   private:
      virtual void beginJob() override;
      virtual void analyze(const edm::Event&, const edm::EventSetup&) override;
      virtual void endJob() override;

      virtual void beginRun(edm::Run const&, edm::EventSetup const&);
      virtual void endRun(edm::Run const&, edm::EventSetup const&);

      bool jsonLocal(int r, int ls);

      // ----------member data ---------------------------
  //      edm::EDGetTokenT<TrackCollection> tracksToken_;  //used to select what tracks to read f
rom configuration file
  edm::EDGetTokenT<reco::TrackCollection> trkToken_;
  edm::EDGetTokenT<vector<CTPPSLocalTrackLite> > RPtrkToken_;
  edm::EDGetTokenT<reco::VertexCollection> vtxToken_;
  edm::EDGetTokenT<reco::BeamSpot> beamspotToken_;
  edm::EDGetTokenT<edm::TriggerResults>  trigToken_;
  // V0 ...Luiz
  edm::EDGetTokenT<reco::VertexCompositeCandidateCollection> kshortsToken_;
  edm::EDGetTokenT<reco::VertexCompositeCandidateCollection> lambdasToken_;
  //edm::EDGetTokenT<reco::ForwardProtonCollection> RecoProtonsSingleRPToken_;
  //edm::EDGetTokenT<reco::ForwardProtonCollection> RecoProtonsMultiRPToken_;

  HLTConfigProvider hltConfig_;

  map<string,TH1F*> histosTH1F;
  map<string,TH2F*> histosTH2F;

};

//

// constants, enums and typedefs

//


//

// static data member definitions

//


//

// constructors and destructor ...including kshorts and lambdas and reco RP protons ...Luiz

//
// TAGs from python configuration : TOKEN <-> TAG
```

```
PromptAnalyzer::PromptAnalyzer(const edm::ParameterSet& iConfig)
  :
  trkToken_(consumes<reco::TrackCollection>(iConfig.getParameter<edm::InputTag>("tracks")))
  ,RPtrkToken_(consumes<vector<CTPPSLocalTrackLite> >(iConfig.getParameter<edm::InputTag>("RPtrac
ks")))
  ,vtxToken_(consumes<reco::VertexCollection>(iConfig.getParameter<edm::InputTag>("vertices")))
  ,beamspotToken_(consumes<reco::BeamSpot>(iConfig.getParameter<edm::InputTag>("beamspot")))
  ,trigToken_(consumes<edm::TriggerResults>(iConfig.getParameter<edm::InputTag>("triggers")))
  // V0 ...Luiz

  ,kshortsToken_(consumes<reco::VertexCompositeCandidateCollection>(iConfig.getParameter<edm::Inp
utTag>("kshorts")))
  ,lambdasToken_(consumes<reco::VertexCompositeCandidateCollection>(iConfig.getParameter<edm::Inp
utTag>("lambdas")))
  // reco RP protons ...Luiz
  //,RecoProtonsSingleRPToken_(consumes<reco::ForwardProtonCollection>(iConfig.getParameter<edm::
InputTag>("tagRecoProtonsSingleRP")))
  //,RecoProtonsMultiRPToken_(consumes<reco::ForwardProtonCollection>(iConfig.getParameter<edm::I
nputTag>("tagRecoProtonsMultiRP")))
{

  //now do what ever initialization is needed

  //usesResource("TFileService");
  //edm::Service<TFileService> fs;

  /*

  //----------------------------------

  //-------------------------------------

  */

}


PromptAnalyzer::~PromptAnalyzer()
{

   // do anything here that needs to be done at desctruction time
   // (e.g. close files, deallocate resources etc.)

}

////edm::Service<TFileService> fs;

//
// member functions
//

// ------------ method called for each event  ------------
void
PromptAnalyzer::analyze(const edm::Event& iEvent, const edm::EventSetup& iSetup)
{

  //   using namespace edm;

   //--------------------------

  edm::Handle<TrackCollection> tracks;
  edm::Handle<vector<CTPPSLocalTrackLite> > RPtracks;
  edm::Handle<VertexCollection> vertices;
  edm::Handle<reco::BeamSpot> beamspot;
  edm::Handle<edm::TriggerResults> triggers;
  // kshorts and lambdas ...Luiz
  edm::Handle<reco::VertexCompositeCandidateCollection> kshorts;
  edm::Handle<reco::VertexCompositeCandidateCollection> lambdas;
  // reco RP protons ...Luiz
  //edm::Handle<reco::ForwardProtonCollection> ProtonsSingleRP;
  //edm::Handle<reco::ForwardProtonCollection> ProtonsMultiRP;
```

```
iEvent.getByToken(trkToken_,tracks);
iEvent.getByToken(RPtrkToken_,RPtracks);
iEvent.getByToken(vtxToken_,vertices);
iEvent.getByToken(beamspotToken_,beamspot);
iEvent.getByToken(trigToken_,triggers);
// kshorts and lambdas ...Luiz
iEvent.getByToken(kshortsToken_,kshorts);
iEvent.getByToken(lambdasToken_,lambdas);
// reco RP protons ...Luiz
//iEvent.getByToken(RecoProtonsSingleRPToken_,ProtonsSingleRP);
//iEvent.getByToken(RecoProtonsMultiRPToken_,ProtonsMultiRP);


//----------------------------------------------------------------------
/*
std::cout<<"check triggers"<<std::endl;

if (triggers.isValid()) {
  edm::TriggerNames triggerNames = iEvent.triggerNames(*triggers);
  unsigned int size = triggers->size();

  for(unsigned ij = 0; ij<size; ++ij) {
    std::string name = triggerNames.triggerName(ij);
    //      const char* variab1 = name.c_str();

    if( triggers->accept(ij) ){
      std::cout<<ij<<") "<<name<<" accept="<<triggers->accept(ij)<<std::endl;
    }
  }
}
*/
//----------------------------------------------------------------------
//TB/BT or TT/BB are now separately in TOTEM2 or TOTEM4
// 0 = for TB in TOTEM2 or TT in TOTEM4
// 1 = for BT in TOTEM2 or BB in TOTEM4

int runnr =  iEvent.id().run();
//  int eventnr =  iEvent.id().event();
int LS = iEvent.luminosityBlock();

//LS histos were here

int ntrk0 = 0;
int ntrk  = 0;

int totcharge=0;

//tracks in 2track-events (npixelhits>0)
//TLorentzVector pi1(0.,0.,0.,0.);
//TLorentzVector pi2(0.,0.,0.,0.);
TLorentzVector pipiRec(0.,0.,0.,0.);
//...Luiz
TLorentzVector Pi1(0.,0.,0.,0.);
TLorentzVector Pi2(0.,0.,0.,0.);

//tracks in 4track-events (npixelhits>0)
TLorentzVector pi4pos1(0.,0.,0.,0.);
TLorentzVector pi4pos2(0.,0.,0.,0.);
TLorentzVector pi4neg1(0.,0.,0.,0.);
TLorentzVector pi4neg2(0.,0.,0.,0.);
TLorentzVector pi4Rec(0.,0.,0.,0.);

int ntrk4pos=0;
int ntrk4neg=0;

double xBS,yBS,zBS;
if (beamspot.isValid()){
  xBS = beamspot->x0();
  yBS = beamspot->y0();
  zBS = beamspot->z0();
}else{
```

```
  std::cout<<"sorry, no beamspot"<<std::endl;
  xBS=0;
  yBS=0;
  zBS=0;
}

// my cuts ...Luiz
bool fiducialRegion4 = false;
bool fiducialRegionPt4 = false;

bool fiducialRegionK4 = false;
bool fiducialRegionPtK4 = false;

double etaCut = 2.5;
double ptCut = 0.0;

// my tracks in 4track-events (npixelhits>0)
// pions & kaons ...Luiz
TLorentzVector pi1(0.,0.,0.,0.);
TLorentzVector pi2(0.,0.,0.,0.);
TLorentzVector pi3(0.,0.,0.,0.);
TLorentzVector pi4(0.,0.,0.,0.);
//
TLorentzVector piA(0.,0.,0.,0.);
TLorentzVector piB(0.,0.,0.,0.);
TLorentzVector piC(0.,0.,0.,0.);
TLorentzVector piD(0.,0.,0.,0.);
//
TLorentzVector k1(0.,0.,0.,0.);
TLorentzVector k2(0.,0.,0.,0.);
TLorentzVector k3(0.,0.,0.,0.);
TLorentzVector k4(0.,0.,0.,0.);
//
TLorentzVector kA(0.,0.,0.,0.);
TLorentzVector kB(0.,0.,0.,0.);
TLorentzVector kC(0.,0.,0.,0.);
TLorentzVector kD(0.,0.,0.,0.);
//
TLorentzVector pi1pi2Rec(0.,0.,0.,0.);
TLorentzVector pi3pi4Rec(0.,0.,0.,0.);
TLorentzVector pi1pi3Rec(0.,0.,0.,0.);
TLorentzVector pi2pi4Rec(0.,0.,0.,0.);
TLorentzVector pi1pi4Rec(0.,0.,0.,0.);
TLorentzVector pi2pi3Rec(0.,0.,0.,0.);
//
TLorentzVector k1k2Rec(0.,0.,0.,0.);
TLorentzVector k3k4Rec(0.,0.,0.,0.);
TLorentzVector k1k3Rec(0.,0.,0.,0.);
TLorentzVector k2k4Rec(0.,0.,0.,0.);
TLorentzVector k1k4Rec(0.,0.,0.,0.);
TLorentzVector k2k3Rec(0.,0.,0.,0.);

// checking
TLorentzVector pi1234Rec(0.,0.,0.,0.);
TLorentzVector pi1324Rec(0.,0.,0.,0.);
TLorentzVector pi1423Rec(0.,0.,0.,0.);

//...combining pions and kaons for the event selection type = 11 (one primary & one Vee)
 /*
...first combining, then select the Q_pairs=0
pi1pi2 pi3k4
pi1pi3 pi2k4
pi2pi3 pi1k4

pi1pi2 k3pi4
pi1pi4 k3pi2
pi2pi4 k3pi1

pi1k2 pi3pi4
pi3k2 pi1pi4
pi4k2 pi1pi3
```

```
k1pi2 pi3pi4
k1pi3 pi2pi4
k1pi4 pi2pi3
 */

//
TLorentzVector pi3k4Rec(0.,0.,0.,0.);
TLorentzVector pi2k4Rec(0.,0.,0.,0.);
TLorentzVector pi1k4Rec(0.,0.,0.,0.);
//
TLorentzVector k3pi4Rec(0.,0.,0.,0.);
TLorentzVector k3pi2Rec(0.,0.,0.,0.);
TLorentzVector k3pi1Rec(0.,0.,0.,0.);
//
TLorentzVector pi1k2Rec(0.,0.,0.,0.);
TLorentzVector pi3k2Rec(0.,0.,0.,0.);
TLorentzVector pi4k2Rec(0.,0.,0.,0.);
//
TLorentzVector k1pi2Rec(0.,0.,0.,0.);
TLorentzVector k1pi3Rec(0.,0.,0.,0.);
TLorentzVector k1pi4Rec(0.,0.,0.,0.);
//
TLorentzVector pipipipiRec(0.,0.,0.,0.);
TLorentzVector kkkkRec(0.,0.,0.,0.);
//

int charray[4]={0,0,0,0};
double chi2array[4]={0.,0.,0.,0.};
double d0array[4]={0.,0.,0.,0.};
double dzarray[4]={0.,0.,0.,0.};
int pidarray[4]={0,0,0,0};
double vtxdxyarray[4]={0.,0.,0.,0.};
double vtxdzarray[4]={0.,0.,0.,0.};
double phiarray[4]={0.,0.,0.,0.};
//...ordering
int arraych[4]={0,0,0,0};
double arraychi2[4]={0.,0.,0.,0.};
double arrayd0[4]={0.,0.,0.,0.};
double arraydz[4]={0.,0.,0.,0.};
int arraypid[4]={0,0,0,0};
double arrayvtxdxy[4]={0.,0.,0.,0.};
double arrayvtxdz[4]={0.,0.,0.,0.};
double arrayphi[4]={0.,0.,0.,0.};

     // ...transverse impact parameter distribution
     double d01 = 0.0;
     double d02 = 0.0;
     double d03 = 0.0;
     double d04 = 0.0;
     // ...longitudinal impact parameter distribution
     double dz1 = 0.0;
     double dz2 = 0.0;
     double dz3 = 0.0;
     double dz4 = 0.0;
     // ...transverse impact parameter distribution
     double vtxdxy1 = 0.0;
     double vtxdxy2 = 0.0;
     double vtxdxy3 = 0.0;
     double vtxdxy4 = 0.0;
     // ...longitudinal impact parameter distribution
     double vtxdz1 = 0.0;
     double vtxdz2 = 0.0;
     double vtxdz3 = 0.0;
     double vtxdz4 = 0.0;

//...Luiz
// Get primary vertex PV
//if (vertices->empty()) return;  // skip the event if no PV found
//const reco::Vertex& pv = vertices->begin()->position();

  //...Luiz
  //...to avoid 'not declared in this scope'
```

```
    //math::XYZPoint ppv(0.,0.,0.);
    math::XYZPoint pv(0.,0.,0.);
    double pvx = 0.;
    double pvy = 0.;
    double pvz = 0.;
    //int visfake = vertices->begin()->isFake();

    //...what does it mean vertices->empty() ???

    //...notice that *vertex* IS NOT *primary vertex*
    //   this difference is clear in the software
    //   vertex is the interaction point (0,0,0)
    //   track referencePoint() is the PCA w.r.t. vertex or (0,0,0)
    //   track referencePoint(pv) is the PCA w.r.t. primary vertex

    //...there are no vertices->empty events
    //...I am disabling the IF statement

    //...vertices are primary vertices from the collection and we do have events with no primarie
s

    //if ( !( vertices->empty() ) ){
    //math::XYZPoint pvtx = itTrack->referencePoint();  // PCA w.r.t. the center of CMS or beamsp
ot (0,0,0)
    //math::XYZPoint ppv = (*vertices)[0].position();   // w.r.t. primary vertex
    //math::XYZPoint pv = vertices->begin()->position();// w.r.t. primary vertex
    //ppv = (*vertices)[0].position();

    pvx =  vertices->begin()->x();
    pvy =  vertices->begin()->y();
    pvz =  vertices->begin()->z();

    math::XYZPoint mypv(pvx,pvy,pvz); //...important only within the IF

    pv = mypv; //...scrumptious
    //}

    /*
    std::cout << " --- primary vertex position ----------------" << std::endl;
    std::cout << "pv  = " << pv  << std::endl;
    std::cout << "pvx = " << pvx << std::endl;
    std::cout << "pvy = " << pvy << std::endl;
    std::cout << "pvz = " << pvz << std::endl;
    //std::cout << "ppv = " << ppv  << std::endl;
    */

    //...checking!
    if ( vertices->empty() ){
      std::cout << " +++ empty +++++++++++++++++  " << std::endl;
    }

  for(TrackCollection::const_iterator itTrack = tracks->begin();itTrack != tracks->end();++itTrac
k) {

    int looper = itTrack->isLooper();
    //    double p = itTrack->p();
    double pt = itTrack->pt();
    double pz = itTrack->pz();
    double eta = itTrack->eta();
    double phi = itTrack->phi();
    int charge = itTrack->charge();
    int npixelhits = itTrack->hitPattern().numberOfValidPixelHits();
    //    int nstriphits = itTrack->hitPattern().numberOfValidStripHits();
    int algo = itTrack->algo();
    double chi2 = itTrack->normalizedChi2();
    double d0 = itTrack->d0();
    double dz = itTrack->dz();

    // Bene: One can use TransientTracks to estimate track impact parameters with
    // respect to the beam line or primary vertex, taking into account the curvature of the track
.
```

```
    //...Luiz
    //math::XYZPoint pv = itTrack->referencePoint(); // PCA w.r.t. the center of CMS or beamspot
(0,0,0)
    //double pvx =  itTrack->vx();
    //double pvy =  itTrack->vy();
    //double pvz =  itTrack->vz();
    double vdxy = itTrack->dxy(pv); // vtxdxy : transverse impact parameter
    double vdz =  itTrack->dz(pv);  // vtxdz  : longitudinal impact parameter

    //...Bene: be sure the pt of the tracks is bigger than 0.5 GeV (?)

    /*
    std::cout.precision(10);
    std::cout << " --- primary vertex ----------------" << std::endl;
    std::cout << "pv  = " << pv  << std::endl;
    std::cout << "pvx = " << pvx << std::endl;
    std::cout << "pvy = " << pvy << std::endl;
    std::cout << "pvz = " << pvz << std::endl;
    */

    /*
    std::cout << " *** transverse i.p.   *** " << std::endl;
    std::cout << "vdxy = " << vdxy << std::endl;
    std::cout << " --- longitudinal i.p. --- " << std::endl;
    std::cout << "vdz  = " << vdz << std::endl;
    */

    /*
    //...considering the track curvature
    //
    // with respect to any specified vertex, such as primary vertex
    GlobalPoint vert(pv.x(), pv.y(), pv.z());
    TrajectoryStateClosestToPoint  traj = itTrack->trajectoryStateClosestToPoint(vert );
    double dd0 = traj.perigeeParameters().transverseImpactParameter();
    double zz0 = traj.perigeeParameters().longitudinalImpactParameter();
    */

    //...attention here!
    //// comment this line out only for the M(K0) window technique
    if(npixelhits>0){

      histosTH1F["hpt"]->Fill(pt);
      histosTH1F["heta"]->Fill(eta);
      histosTH1F["hphi"]->Fill(phi);
      histosTH1F["halgo"]->Fill(algo);
      histosTH1F["hnhits"]->Fill(npixelhits);

      histosTH1F["hlooper"]->Fill(looper);
      histosTH1F["hchi2"]->Fill(chi2);
      histosTH1F["hd0"]->Fill(d0);
      histosTH1F["hdz"]->Fill(dz);

      math::XYZPoint point(xBS,yBS,zBS);
      double d0BS = itTrack->dxy(point);
      double dzBS = itTrack->dz(point);

      histosTH1F["hd0BS"]->Fill(d0BS);
      histosTH1F["hdzBS"]->Fill(dzBS);

      //...temporarily to avoid crash ...Luiz
      //double vdxy = d0BS ;
      //double vdz = dzBS ;

      totcharge += charge;

      // pions
      double ene=TMath::Sqrt(pt*pt+pz*pz+m_pi*m_pi);
      TLorentzVector trk_lorentz(itTrack->px(),itTrack->py(),itTrack->pz(),ene);

      //--------------------------------------
      // 2 trk
      //double ene=TMath::Sqrt(pt*pt+pz*pz+m_pi*m_pi);
```

```
//TLorentzVector trk_lorentz(itTrack->px(),itTrack->py(),itTrack->pz(),ene);
pipiRec += trk_lorentz;

//if(ntrk==0) pi1 = trk_lorentz;
//if(ntrk==1) pi2 = trk_lorentz;
//...Luiz
if(ntrk==0) Pi1 = trk_lorentz;
if(ntrk==1) Pi2 = trk_lorentz;

//--------------------------------------
// 4trk

pi4Rec += trk_lorentz;

if(charge>0){
  if(ntrk4pos==0) pi4pos1 = trk_lorentz;
  if(ntrk4pos==1) pi4pos2 = trk_lorentz;

  ntrk4pos++;
}

if(charge<0){
  if(ntrk4neg==0) pi4neg1 = trk_lorentz;
  if(ntrk4neg==1) pi4neg2 = trk_lorentz;

  ntrk4neg++;
}
//--------------------------------------

//--------------------------------------
// my 4trk definitions ...Luiz

     pipipipiRec += trk_lorentz;

     //...first, tagging by track number
     if(ntrk==0) piA = trk_lorentz;
     if(ntrk==1) piB = trk_lorentz;
     if(ntrk==2) piC = trk_lorentz;
     if(ntrk==3) piD = trk_lorentz;

     /*
     //...Luiz
     EPID pid2 = ParticleType::guess( itTrack->p() , itTrack->dedx() );
     //...V0
     EPID pidV0 = ParticleType::sure( itTrack->p() , itTrack->dedx() );

     std::cout << " ..^..^.. " << std::endl;
     std::cout << "pid2 = " << pid2 << std::endl;
     std::cout << "pidV0 = " << pidV0 << std::endl;
     */


     //...temporarily ...no PID yet ...Luiz
     int pid2 = 3;


   //...beware of the index here
     if(ntrk==0){
       charray[0]=charge;
       chi2array[0]=chi2;
       d0array[0]=d0;
       dzarray[0]=dz;
       pidarray[0]=pid2;
       vtxdxyarray[0]=vdxy;
       vtxdzarray[0]=vdz;
       phiarray[0]=phi;
     }
     if(ntrk==1){
       charray[1]=charge;
       chi2array[1]=chi2;
       d0array[1]=d0;
       dzarray[1]=dz;
```

```
      pidarray[1]=pid2;
      vtxdxyarray[1]=vdxy;
      vtxdzarray[1]=vdz;
      phiarray[1]=phi;
    }
    if(ntrk==2){
      charray[2]=charge;
      chi2array[2]=chi2;
      d0array[2]=d0;
      dzarray[2]=dz;
      pidarray[2]=pid2;
      vtxdxyarray[2]=vdxy;
      vtxdzarray[2]=vdz;
      phiarray[2]=phi;
    }
    if(ntrk==3){
      charray[3]=charge;
      chi2array[3]=chi2;
      d0array[3]=d0;
      dzarray[3]=dz;
      pidarray[3]=pid2;
      vtxdxyarray[3]=vdxy;
      vtxdzarray[3]=vdz;
      phiarray[3]=phi;
    }

//...ordering pions and kaons by momentum, index=1 is the highest Pt
//...we need to include kaons for the selection 11 : one primary and one Vee

vector<Double_t> piVec = { piA.Pt(), piB.Pt(), piC.Pt(), piD.Pt() };
//
sort(piVec.begin(), piVec.end());

//...ordering by Pt and connecting the charges & PID's to the particles...tricky!

if(piVec[3]!=0.0 && piVec[3]==piA.Pt()){ pi1 = piA ;
  arraych[0]=charray[0];
arraychi2[0]=chi2array[0];
  arrayd0[0]=d0array[0];
  arraydz[0]=dzarray[0];
 arraypid[0]=pidarray[0];
 arrayphi[0]=phiarray[0];
 arrayvtxdxy[0]=vtxdxyarray[0];
 arrayvtxdz[0]=vtxdzarray[0];}

if(piVec[3]!=0.0 && piVec[3]==piB.Pt()){ pi1 = piB ;
  arraych[0]=charray[1];
arraychi2[0]=chi2array[1];
  arrayd0[0]=d0array[1];
  arraydz[0]=dzarray[1];
 arraypid[0]=pidarray[1];
 arrayphi[0]=phiarray[1];
 arrayvtxdxy[0]=vtxdxyarray[1];
 arrayvtxdz[0]=vtxdzarray[1];}

if(piVec[3]!=0.0 && piVec[3]==piC.Pt()){ pi1 = piC ;
  arraych[0]=charray[2];
arraychi2[0]=chi2array[2];
  arrayd0[0]=d0array[2];
  arraydz[0]=dzarray[2];
 arraypid[0]=pidarray[2];
 arrayphi[0]=phiarray[2];
 arrayvtxdxy[0]=vtxdxyarray[2];
 arrayvtxdz[0]=vtxdzarray[2];}

if(piVec[3]!=0.0 && piVec[3]==piD.Pt()){ pi1 = piD ;
  arraych[0]=charray[3];
arraychi2[0]=chi2array[3];
  arrayd0[0]=d0array[3];
  arraydz[0]=dzarray[3];
 arraypid[0]=pidarray[3];
 arrayphi[0]=phiarray[3];
```

```
 arrayvtxdxy[0]=vtxdxyarray[3];
 arrayvtxdz[0]=vtxdzarray[3];}

//
if(piVec[2]!=0.0 && piVec[2]==piA.Pt()){ pi2 = piA ;
  arraych[1]=charray[0];
arraychi2[1]=chi2array[0];
  arrayd0[1]=d0array[0];
  arraydz[1]=dzarray[0];
 arraypid[1]=pidarray[0];
 arrayphi[1]=phiarray[0];
 arrayvtxdxy[1]=vtxdxyarray[0];
 arrayvtxdz[1]=vtxdzarray[0];}

if(piVec[2]!=0.0 && piVec[2]==piB.Pt()){ pi2 = piB ;
  arraych[1]=charray[1];
arraychi2[1]=chi2array[1];
  arrayd0[1]=d0array[1];
  arraydz[1]=dzarray[1];
 arraypid[1]=pidarray[1];
 arrayphi[1]=phiarray[1];
 arrayvtxdxy[1]=vtxdxyarray[1];
 arrayvtxdz[1]=vtxdzarray[1];}

if(piVec[2]!=0.0 && piVec[2]==piC.Pt()){ pi2 = piC ;
  arraych[1]=charray[2];
arraychi2[1]=chi2array[2];
  arrayd0[1]=d0array[2];
  arraydz[1]=dzarray[2];
 arraypid[1]=pidarray[2];
 arrayphi[1]=phiarray[2];
 arrayvtxdxy[1]=vtxdxyarray[2];
 arrayvtxdz[1]=vtxdzarray[2];}

if(piVec[2]!=0.0 && piVec[2]==piD.Pt()){ pi2 = piD ;
  arraych[1]=charray[3];
arraychi2[1]=chi2array[3];
  arrayd0[1]=d0array[3];
  arraydz[1]=dzarray[3];
 arraypid[1]=pidarray[3];
 arrayphi[1]=phiarray[3];
 arrayvtxdxy[1]=vtxdxyarray[3];
 arrayvtxdz[1]=vtxdzarray[3];}

//
if(piVec[1]!=0.0 && piVec[1]==piA.Pt()){ pi3 = piA ;
  arraych[2]=charray[0];
arraychi2[2]=chi2array[0];
  arrayd0[2]=d0array[0];
  arraydz[2]=dzarray[0];
 arraypid[2]=pidarray[0];
 arrayphi[2]=phiarray[0];
 arrayvtxdxy[2]=vtxdxyarray[0];
 arrayvtxdz[2]=vtxdzarray[0];}

if(piVec[1]!=0.0 && piVec[1]==piB.Pt()){ pi3 = piB ;
  arraych[2]=charray[1];
arraychi2[2]=chi2array[1];
  arrayd0[2]=d0array[1];
  arraydz[2]=dzarray[1];
 arraypid[2]=pidarray[1];
 arrayphi[2]=phiarray[1];
 arrayvtxdxy[2]=vtxdxyarray[1];
 arrayvtxdz[2]=vtxdzarray[1];}

if(piVec[1]!=0.0 && piVec[1]==piC.Pt()){ pi3 = piC ;
  arraych[2]=charray[2];
arraychi2[2]=chi2array[2];
  arrayd0[2]=d0array[2];
  arraydz[2]=dzarray[2];
 arraypid[2]=pidarray[2];
 arrayphi[2]=phiarray[2];
```

```
 arrayvtxdxy[2]=vtxdxyarray[2];
 arrayvtxdz[2]=vtxdzarray[2];}

if(piVec[1]!=0.0 && piVec[1]==piD.Pt()){ pi3 = piD ;
  arraych[2]=charray[3];
arraychi2[2]=chi2array[3];
  arrayd0[2]=d0array[3];
  arraydz[2]=dzarray[3];
 arraypid[2]=pidarray[3];
 arrayphi[2]=phiarray[3];
 arrayvtxdxy[2]=vtxdxyarray[3];
 arrayvtxdz[2]=vtxdzarray[3];}

//
if(piVec[0]!=0.0 && piVec[0]==piA.Pt()){ pi4 = piA ;
  arraych[3]=charray[0];
arraychi2[3]=chi2array[0];
  arrayd0[3]=d0array[0];
  arraydz[3]=dzarray[0];
 arraypid[3]=pidarray[0];
 arrayphi[3]=phiarray[0];
 arrayvtxdxy[3]=vtxdxyarray[0];
 arrayvtxdz[3]=vtxdzarray[0];}

if(piVec[0]!=0.0 && piVec[0]==piB.Pt()){ pi4 = piB ;
  arraych[3]=charray[1];
arraychi2[3]=chi2array[1];
  arrayd0[3]=d0array[1];
  arraydz[3]=dzarray[1];
 arraypid[3]=pidarray[1];
 arrayphi[3]=phiarray[1];
 arrayvtxdxy[3]=vtxdxyarray[1];
 arrayvtxdz[3]=vtxdzarray[1];}

if(piVec[0]!=0.0 && piVec[0]==piC.Pt()){ pi4 = piC ;
  arraych[3]=charray[2];
arraychi2[3]=chi2array[2];
  arrayd0[3]=d0array[2];
  arraydz[3]=dzarray[2];
 arraypid[3]=pidarray[2];
 arrayphi[3]=phiarray[2];
 arrayvtxdxy[3]=vtxdxyarray[2];
 arrayvtxdz[3]=vtxdzarray[2];}

if(piVec[0]!=0.0 && piVec[0]==piD.Pt()){ pi4 = piD ;
  arraych[3]=charray[3];
arraychi2[3]=chi2array[3];
  arrayd0[3]=d0array[3];
  arraydz[3]=dzarray[3];
 arraypid[3]=pidarray[3];
 arrayphi[3]=phiarray[3];
 arrayvtxdxy[3]=vtxdxyarray[3];
 arrayvtxdz[3]=vtxdzarray[3];}


//------------------------

// kaons
double eneK=TMath::Sqrt(pt*pt+pz*pz+m_k*m_k);
TLorentzVector trk_lorentzK(itTrack->px(),itTrack->py(),itTrack->pz(),eneK);
kkkkRec += trk_lorentzK;

    if(ntrk==0) kA = trk_lorentzK;
    if(ntrk==1) kB = trk_lorentzK;
    if(ntrk==2) kC = trk_lorentzK;
    if(ntrk==3) kD = trk_lorentzK;

vector<Double_t> kVec = { kA.Pt(), kB.Pt(), kC.Pt(), kD.Pt() };
//
sort(kVec.begin(), kVec.end());
 //
if(kVec[3]!=0.0 && kVec[3]==kA.Pt()){ k1 = kA ; }
```

```
      if(kVec[3]!=0.0 && kVec[3]==kB.Pt()){ k1 = kB ; }
      if(kVec[3]!=0.0 && kVec[3]==kC.Pt()){ k1 = kC ; }
      if(kVec[3]!=0.0 && kVec[3]==kD.Pt()){ k1 = kD ; }
      //
      if(kVec[2]!=0.0 && kVec[2]==kA.Pt()){ k2 = kA ; }
      if(kVec[2]!=0.0 && kVec[2]==kB.Pt()){ k2 = kB ; }
      if(kVec[2]!=0.0 && kVec[2]==kC.Pt()){ k2 = kC ; }
      if(kVec[2]!=0.0 && kVec[2]==kD.Pt()){ k2 = kD ; }
      //
      if(kVec[1]!=0.0 && kVec[1]==kA.Pt()){ k3 = kA ; }
      if(kVec[1]!=0.0 && kVec[1]==kB.Pt()){ k3 = kB ; }
      if(kVec[1]!=0.0 && kVec[1]==kC.Pt()){ k3 = kC ; }
      if(kVec[1]!=0.0 && kVec[1]==kD.Pt()){ k3 = kD ; }
      //
      if(kVec[0]!=0.0 && kVec[0]==kA.Pt()){ k4 = kA ; }
      if(kVec[0]!=0.0 && kVec[0]==kB.Pt()){ k4 = kB ; }
      if(kVec[0]!=0.0 && kVec[0]==kC.Pt()){ k4 = kC ; }
      if(kVec[0]!=0.0 && kVec[0]==kD.Pt()){ k4 = kD ; }

     //---------------------------------------
     //---------------------------------------
     // end of my 4trk definitions


     ntrk++; //pixel tracks
    } //...end of npixelhits>0

   ntrk0++; //all tracks

} //...end of tracks

histosTH1F["hntrk0"]->Fill(ntrk0);
histosTH1F["hntrk"]->Fill(ntrk);

      if(ntrk==4 && totcharge==0){
        histosTH1F["hntrk4q0"]->Fill(ntrk);
      }


      double pi1pt = pi1.Pt();
      double pi2pt = pi2.Pt();
      double pi3pt = pi3.Pt();
      double pi4pt = pi4.Pt();


          // my 4-vectors
          pi1pi2Rec = pi1 + pi2;
          pi3pi4Rec = pi3 + pi4;
          pi1pi3Rec = pi1 + pi3;
          pi2pi4Rec = pi2 + pi4;
          pi1pi4Rec = pi1 + pi4;
          pi2pi3Rec = pi2 + pi3;
          //
          k1k2Rec = k1 + k2;
          k3k4Rec = k3 + k4;
          k1k3Rec = k1 + k3;
          k2k4Rec = k2 + k4;
          k1k4Rec = k1 + k4;
          k2k3Rec = k2 + k3;

          // ...checking out
          pi1234Rec = pi1pi2Rec +  pi3pi4Rec;
          pi1324Rec = pi1pi3Rec +  pi2pi4Rec;
          pi1423Rec = pi1pi4Rec +  pi2pi3Rec;


    //...combining pions and kaons for the event selection type = 11 (one primary & one Vee)
          //
          /*
    ...first combining, then select the Q_pairs=0
    pi1pi2 pi3k4
    pi1pi3 pi2k4
```

```
pi2pi3 pi1k4

pi1pi2 k3pi4
pi1pi4 k3pi2
pi2pi4 k3pi1

pi1k2 pi3pi4
pi3k2 pi1pi4
pi4k2 pi1pi3

k1pi2 pi3pi4
k1pi3 pi2pi4
k1pi4 pi2pi3
      */

      //...commented out means already defined
      //
      //pi1pi2Rec
      pi3k4Rec  = pi3 + k4;
      //pi1pi3Rec
      pi2k4Rec  = pi2 + k4;
      pi2pi3Rec = pi2 + pi3; //...for completeness
      pi1k4Rec  = pi1 + k4;
      //
      //
      //pi1pi2Rec
      k3pi4Rec  = k3  + pi4;
      pi1pi4Rec = pi1 + pi4; //...for completeness
      k3pi2Rec  = k3  + pi2;
      //pi2pi4Rec
      k3pi1Rec  = k3  + pi1;
      //
      //
      pi1k2Rec  = pi1 + k2;
      //pi3pi4Rec
      pi3k2Rec  = pi3 + k2;
      //pi1pi4Rec
      pi4k2Rec  = pi4 + k2;
      //pi1pi3Rec
      //
      //
      k1pi2Rec  = k1  + pi2;
      //pi3pi4Rec
      k1pi3Rec  = k1  + pi3;
      //pi2pi4Rec

      k1pi4Rec  = k1  + pi4;
      //pi2pi3Rec

      /*
      pipiRec = pi1pi2Rec +
                pi3pi4Rec +
                pi1pi3Rec +
                pi2pi4Rec +
                pi1pi4Rec +
                pi2pi3Rec ;
      */

      //...reseting to the original definition with new order

    charray[0] =   arraych[0] ;// charge;
  chi2array[0] = arraychi2[0] ;// chi2;
    d0array[0] =   arrayd0[0] ;// d0;
    dzarray[0] =   arraydz[0] ;// dz;
   pidarray[0] =  arraypid[0] ;// pid2;
   phiarray[0] =  arrayphi[0] ;// phi;
vtxdxyarray[0] = arrayvtxdxy[0];//vtxdxy;
 vtxdzarray[0] =  arrayvtxdz[0];//vtxdz;
    //
    charray[1] =   arraych[1] ;// charge;
  chi2array[1] = arraychi2[1] ;// chi2;
    d0array[1] =   arrayd0[1] ;// d0;
```

```
        dzarray[1] =   arraydz[1] ;// dz;
       pidarray[1] =  arraypid[1] ;// pid2;
       phiarray[1] =  arrayphi[1] ;// phi;
     vtxdxyarray[1] = arrayvtxdxy[1];//vtxdxy;
      vtxdzarray[1] =  arrayvtxdz[1];//vtxdz;
        //
        charray[2] =   arraych[2] ;// charge;
      chi2array[2] = arraychi2[2] ;// chi2;
        d0array[2] =   arrayd0[2] ;// d0;
        dzarray[2] =   arraydz[2] ;// dz;
       pidarray[2] =  arraypid[2] ;// pid2;
       phiarray[2] =  arrayphi[2] ;// phi;
     vtxdxyarray[2] = arrayvtxdxy[2];//vtxdxy;
      vtxdzarray[2] =  arrayvtxdz[2];//vtxdz;
        //
        charray[3] =   arraych[3] ;// charge;
      chi2array[3] = arraychi2[3] ;// chi2;
        d0array[3] =   arrayd0[3] ;// d0;
        dzarray[3] =   arraydz[3] ;// dz;
       pidarray[3] =  arraypid[3] ;// pid2;
       phiarray[3] =  arrayphi[3] ;// phi;
     vtxdxyarray[3] = arrayvtxdxy[3];//vtxdxy;
      vtxdzarray[3] =  arrayvtxdz[3];//vtxdz;


      /* do we need a similar one here ?
       if(ntrk==0){
         int nclusters=  sipixelcluster_coll->size();
         int nclusters2= sistripcluster_coll->nStripClusters;

         histosTH1F["hnclusters"]->Fill(nclusters);
         histosTH1F["hnclusters2"]->Fill(nclusters2);
       }
      */

      //...finding the tracks connected to the primary vertex using impact parameter...type:11 on
ly
      bool isTrack1 = false ;
      bool isTrack2 = false ;
      bool isTrack3 = false ;
      bool isTrack4 = false ;

      /*
        vector<Double_t> vdxyVec = { TMath::Abs(vtxdxyarray[0]), TMath::Abs(vtxdxyarray[1]),
                                     TMath::Abs(vtxdxyarray[2]), TMath::Abs(vtxdxyarray[3]) };
        // ...better to use 3D impact parameter...REVIEW!
        sort(vdxyVec.begin(), vdxyVec.end());
        //
        if(vdxyVec[0]!=0.0 && vdxyVec[0]==TMath::Abs(vtxdxyarray[0])){ isTrack1 = true ; }
        if(vdxyVec[0]!=0.0 && vdxyVec[0]==TMath::Abs(vtxdxyarray[1])){ isTrack2 = true ; }
        if(vdxyVec[0]!=0.0 && vdxyVec[0]==TMath::Abs(vtxdxyarray[2])){ isTrack3 = true ; }
        if(vdxyVec[0]!=0.0 && vdxyVec[0]==TMath::Abs(vtxdxyarray[3])){ isTrack4 = true ; }
        //
        if(vdxyVec[1]!=0.0 && vdxyVec[1]==TMath::Abs(vtxdxyarray[0])){ isTrack1 = true ; }
        if(vdxyVec[1]!=0.0 && vdxyVec[1]==TMath::Abs(vtxdxyarray[1])){ isTrack2 = true ; }
        if(vdxyVec[1]!=0.0 && vdxyVec[1]==TMath::Abs(vtxdxyarray[2])){ isTrack3 = true ; }
        if(vdxyVec[1]!=0.0 && vdxyVec[1]==TMath::Abs(vtxdxyarray[3])){ isTrack4 = true ; }
      */

       // ...transverse impact parameter distribution d0 (dxy)
       d01 = d0array[0];
       d02 = d0array[1];
       d03 = d0array[2];
       d04 = d0array[3];
       // ...longitudinal impact parameter distribution dz
       dz1 = dzarray[0];
       dz2 = dzarray[1];
       dz3 = dzarray[2];
       dz4 = dzarray[3];

       if(ntrk==4 && totcharge==0){
         histosTH1F["hd01"]->Fill(d01);
```

```
        histosTH1F["hd02"]->Fill(d02);
        histosTH1F["hd03"]->Fill(d03);
        histosTH1F["hd04"]->Fill(d04);
        //
        histosTH1F["hdz1"]->Fill(dz1);
        histosTH1F["hdz2"]->Fill(dz2);
        histosTH1F["hdz3"]->Fill(dz3);
        histosTH1F["hdz4"]->Fill(dz4);
      }
      // ...transverse impact parameter distribution vtxdxy
      vtxdxy1 = vtxdxyarray[0];
      vtxdxy2 = vtxdxyarray[1];
      vtxdxy3 = vtxdxyarray[2];
      vtxdxy4 = vtxdxyarray[3];
      // ...longitudinal impact parameter distribution vtxdz
      vtxdz1 = vtxdzarray[0];
      vtxdz2 = vtxdzarray[1];
      vtxdz3 = vtxdzarray[2];
      vtxdz4 = vtxdzarray[3];

      if(ntrk==4 && totcharge==0){
        histosTH1F["hvtxdxy1"]->Fill(vtxdxy1);
        histosTH1F["hvtxdxy2"]->Fill(vtxdxy2);
        histosTH1F["hvtxdxy3"]->Fill(vtxdxy3);
        histosTH1F["hvtxdxy4"]->Fill(vtxdxy4);
        //
        histosTH1F["hvtxdz1"]->Fill(vtxdz1);
        histosTH1F["hvtxdz2"]->Fill(vtxdz2);
        histosTH1F["hvtxdz3"]->Fill(vtxdz3);
        histosTH1F["hvtxdz4"]->Fill(vtxdz4);
      }

  // ...Luiz
  //...3D impact parameter
  double rimpac1 = TMath::Sqrt( vtxdxy1*vtxdxy1 + vtxdz1*vtxdz1 );
  double rimpac2 = TMath::Sqrt( vtxdxy2*vtxdxy2 + vtxdz2*vtxdz2 );
  double rimpac3 = TMath::Sqrt( vtxdxy3*vtxdxy3 + vtxdz3*vtxdz3 );
  double rimpac4 = TMath::Sqrt( vtxdxy4*vtxdxy4 + vtxdz4*vtxdz4 );

       vector<Double_t> rimpacVec = { rimpac1, rimpac2, rimpac3, rimpac4 };

       sort(rimpacVec.begin(), rimpacVec.end());
       //
       //...type:11
       //
       //...prompt tracks have smaller impact parameters w.r.t. pv
       if(rimpacVec[0]!=0.0 && rimpacVec[0]==rimpac1){ isTrack1 = true ; }
       if(rimpacVec[0]!=0.0 && rimpacVec[0]==rimpac2){ isTrack2 = true ; }
       if(rimpacVec[0]!=0.0 && rimpacVec[0]==rimpac3){ isTrack3 = true ; }
       if(rimpacVec[0]!=0.0 && rimpacVec[0]==rimpac4){ isTrack4 = true ; }
       //
       if(rimpacVec[1]!=0.0 && rimpacVec[1]==rimpac1){ isTrack1 = true ; }
       if(rimpacVec[1]!=0.0 && rimpacVec[1]==rimpac2){ isTrack2 = true ; }
       if(rimpacVec[1]!=0.0 && rimpacVec[1]==rimpac3){ isTrack3 = true ; }
       if(rimpacVec[1]!=0.0 && rimpacVec[1]==rimpac4){ isTrack4 = true ; }

  //--------------------------------------------------------------------------
  // VERTEX

  int nvtx=0;
  int ntrkvtx = 0;
  int ntrkvtxU = 0;

  for(VertexCollection::const_iterator itVtx = vertices->begin();itVtx != vertices->end();++itVtx
) {
    int vtxisfake = itVtx->isFake();
    if(vtxisfake==0) nvtx++;
    else continue;

    //...Luiz
    ntrkvtx = itVtx->nTracks();
    ntrkvtxU = itVtx->tracksSize();
```

```
    //itVtx->Print();
    //std::cout << " ntrkvtx = " << ntrkvtx << std::endl;
  }

  histosTH1F["hnvtx"]->Fill(nvtx);

  if(nvtx==1) histosTH1F["hntrkvtx"]->Fill(ntrkvtx);
  if(nvtx==1) histosTH1F["hntrkvtxU"]->Fill(ntrkvtxU);
       //...Luiz
       if(nvtx==0) histosTH1F["hntrkvtx0"]->Fill(ntrkvtx);
       if(nvtx==2) histosTH1F["hntrkvtx2"]->Fill(ntrkvtx);
       if(nvtx==3) histosTH1F["hntrkvtx3"]->Fill(ntrkvtx);
       if(nvtx==4) histosTH1F["hntrkvtx4"]->Fill(ntrkvtx);


  // considering all of nvtx ...Luiz
  // commented out
  //if(nvtx!=1) return;

  //...Luiz
  int isfake = vertices->begin()->isFake();

  double xvtx = vertices->begin()->x();
  double yvtx = vertices->begin()->y();
  double zvtx = vertices->begin()->z();
  double chi2vtx = vertices->begin()->normalizedChi2();
  double ndofvtx = vertices->begin()->ndof();

  histosTH1F["hvtxx"]->Fill(xvtx);
  histosTH1F["hvtxy"]->Fill(yvtx);
  histosTH1F["hvtxz"]->Fill(zvtx);
  histosTH1F["hvtxchi2"]->Fill(chi2vtx);

  /*
  // ...Luiz     ...this is interesting! primary vertex position & impact parameters
  math::XYZPoint pv2(xvtx,yvtx,zvtx);
  double pv2dxy = dxy(pv2);   ??
  double pv2dz  = dz(pv2);    ??
  histosTH1F["hpv2dxy"]->Fill(pv2dxy);
  histosTH1F["hpv2dz"]->Fill(pv2dz);
  */

  //--------------------------------------------------------------------------
  // V0

  //...Kshort collection...Luiz
  bool isKshort = false;
  int nks=0;

  for(VertexCompositeCandidateCollection::const_iterator it_ks = kshorts->begin() ; it_ks != ksho
rts->end() ; ++it_ks){

         nks++;
         isKshort = nks;
         double ksvertexx = it_ks->vx();
         double ksvertexy = it_ks->vy();
         double ksvertexz = it_ks->vz();
         double kspt = it_ks->pt();
         double kseta = it_ks->eta();
         double ksphi = it_ks->phi();
         double ksmass = it_ks->mass();
         double ksradius = TMath::Sqrt((ksvertexx-xvtx)*(ksvertexx-xvtx)+(ksvertexy-yvtx)*(ksvert
exy-yvtx));
         double ks3Dradius = TMath::Sqrt((ksvertexx-xvtx)*(ksvertexx-xvtx)+(ksvertexy-yvtx)*(ksve
rtexy-yvtx)+(ksvertexz-zvtx)*(ksvertexz-zvtx));
         double ksenergy = TMath::Sqrt(kspt*kspt+0.4976*0.4976);
         double gammalorentzks = ksenergy/0.4976;
         double kslifetime = ksradius/gammalorentzks;
         double ks3Dlifetime = ks3Dradius/gammalorentzks;
         histosTH1F["hkspt"]->Fill(kspt);
         histosTH1F["hkseta"]->Fill(kseta);
         histosTH1F["hksphi"]->Fill(ksphi);
```

```
          histosTH1F["hksmass"]->Fill(ksmass);
          //
          if(nks == 1){histosTH1F["hksmassv1"]->Fill(ksmass);
                       histosTH1F["hksradiusv1"]->Fill(ksradius);
                       histosTH1F["hkslifetimev1"]->Fill(kslifetime);
                       histosTH1F["hks3Dradiusv1"]->Fill(ks3Dradius);
                       histosTH1F["hks3Dlifetimev1"]->Fill(ks3Dlifetime);
          }
          if(nks == 2){histosTH1F["hksmassv2"]->Fill(ksmass);
                       histosTH1F["hksradiusv2"]->Fill(ksradius);
                       histosTH1F["hkslifetimev2"]->Fill(kslifetime);
                       histosTH1F["hks3Dradiusv2"]->Fill(ks3Dradius);
                       histosTH1F["hks3Dlifetimev2"]->Fill(ks3Dlifetime);
          }
          //
          histosTH1F["hksvertexx"]->Fill(ksvertexx);
          histosTH1F["hksvertexy"]->Fill(ksvertexy);
          histosTH1F["hksvertexz"]->Fill(ksvertexz);
          histosTH1F["hksradius"]->Fill(ksradius);
          histosTH1F["hks3Dradius"]->Fill(ks3Dradius);
          histosTH1F["hkslifetime"]->Fill(kslifetime);
          histosTH1F["hks3Dlifetime"]->Fill(ks3Dlifetime);
          histosTH2F["h2dimksxy"]->Fill(ksvertexx,ksvertexy);
          histosTH2F["h2dimksxz"]->Fill(ksvertexx,ksvertexz);
          histosTH2F["h2dimksyz"]->Fill(ksvertexy,ksvertexz);

          /*
          std::cout << " nks = " << nks << std::endl;
          std::cout << " ksvertexx = " << ksvertexx << std::endl;
          std::cout << " ksvertexy = " << ksvertexy << std::endl;
          std::cout << " ksvertexz = " << ksvertexz << std::endl;
          std::cout << " ksmass = " << ksmass << std::endl;
          std::cout << " kspt = " << kspt << std::endl;
          std::cout << " ksradius = " << ksradius << std::endl;
          */
          //it_ks->Print();
}
        //...end of Kshort
        histosTH1F["hnks"]->Fill(nks);
        histosTH2F["hntrknks"]->Fill(ntrk,nks);
        histosTH2F["hnvtxnks"]->Fill(nvtx,nks);
        histosTH2F["hntrknvtx"]->Fill(ntrk,nvtx);
        //std::cout << " ---------------------------- " << std::endl;
        //std::cout << " nks  = " << nks << std::endl;
        //std::cout << " ntrk = " << ntrk << std::endl;
        //std::cout << " nvtx = " << nvtx << std::endl;
        //std::cout << " isKshort = " << isKshort << std::endl;
        //std::cout << " ---------------------------- " << std::endl;


    //...Lambda collection...Luiz
    ////bool isLambda = false;
    //std::cout << "isLambda ...boo! " << isLambda << std::endl;
    int nlam=0;

    for(VertexCompositeCandidateCollection::const_iterator it_lam = lambdas->begin() ; it_lam != l
ambdas->end() ; ++it_lam){

          nlam++;
          //...isLambda = nlam;  //compiler doesn't like it for some mysterious reason! but likes
the above isKshort
          bool   isLambda = nlam;
          double lamvertexx = it_lam->vx();
          double lamvertexy = it_lam->vy();
          double lamvertexz = it_lam->vz();
          double lampt = it_lam->pt();
          double lameta = it_lam->eta();
          double lamphi = it_lam->phi();
          double lammass = it_lam->mass();
          double lamradius = TMath::Sqrt((lamvertexx-xvtx)*(lamvertexx-xvtx)+(lamvertexy-yvtx)*(la
mvertexy-yvtx));
          double lam3Dradius = TMath::Sqrt((lamvertexx-xvtx)*(lamvertexx-xvtx)+(lamvertexy-yvtx)*(
```

```
lamvertexy-yvtx)+(lamvertexz-zvtx)*(lamvertexz-zvtx));
         double lamenergy = TMath::Sqrt(lampt*lampt+1.115683*1.115683);
         double gammalorentzlam = lamenergy/1.115683;
         double lamlifetime = lamradius/gammalorentzlam;
         double lam3Dlifetime = lam3Dradius/gammalorentzlam;
         histosTH1F["hlampt"]->Fill(lampt);
         histosTH1F["hlameta"]->Fill(lameta);
         histosTH1F["hlamphi"]->Fill(lamphi);
         histosTH1F["hlammass"]->Fill(lammass);
         histosTH1F["hlamvertexx"]->Fill(lamvertexx);
         histosTH1F["hlamvertexy"]->Fill(lamvertexy);
         histosTH1F["hlamvertexz"]->Fill(lamvertexz);
         histosTH1F["hlamradius"]->Fill(lamradius);
         histosTH1F["hlam3Dradius"]->Fill(lam3Dradius);
         histosTH1F["hlamlifetime"]->Fill(lamlifetime);
         histosTH1F["hlam3Dlifetime"]->Fill(lam3Dlifetime);
         histosTH2F["h2dimlamxy"]->Fill(lamvertexx,lamvertexy);
         histosTH2F["h2dimlamxz"]->Fill(lamvertexx,lamvertexz);
         histosTH2F["h2dimlamyz"]->Fill(lamvertexy,lamvertexz);
         //std::cout << " ksvertexx = " << ksvertexx << std::endl;
         //std::cout << " ksvertexy = " << ksvertexy << std::endl;
         //std::cout << " ksvertexz = " << ksvertexz << std::endl;
         //std::cout << " ksmass = " << ksmass << std::endl;
         //it_lam->Print();
       }
       //...end of Lambda
       histosTH1F["hnlam"]->Fill(nlam);
       histosTH2F["hntrknlam"]->Fill(ntrk,nlam);
       histosTH2F["hnvtxnlam"]->Fill(nvtx,nlam);


  //------------------------------------------------------------------------------
  //------------------------------------------------------------------------------
  // PPS

  // 2018 setup
  //-------------------------------------------------
  // -z                    IP                +z
  //         sec45                  sec56
  //top:  24       4           104         124
  //ver:     23  3                103 123
  //bot:  25       5           105         125
  //
  //-----------------------------------

  //  load track data;
  //  fwlite::Handle< vector<CTPPSLocalTrackLite> > RPtracks;
  //  RPtracks.getByLabel(iEvent, "ctppsLocalTrackLiteProducer");

  bool rp_valid_004 = false;
  bool rp_valid_005 = false;
  bool rp_valid_024 = false;
  bool rp_valid_025 = false;

  bool rp_valid_104 = false;
  bool rp_valid_105 = false;
  bool rp_valid_124 = false;
  bool rp_valid_125 = false;

  double xLN=100, xLF=100, yLN=100, yLF=100;
  double xRN=100, xRF=100, yRN=100, yRF=100;

  //here is a new set of alignment constants, this time extracted from the
  //physics runs:
  //alignment_corrections[24] = UnitHitData(false, -0.465, -0.689);
  //alignment_corrections[4] = UnitHitData(false, -0.210, -1.479);
  //alignment_corrections[104] = UnitHitData(false, +0.167, -0.916);
  //alignment_corrections[124] = UnitHitData(false, -0.450, +0.044);
  //alignment_corrections[25] = UnitHitData(false, -0.081, +0.009);
  //alignment_corrections[5] = UnitHitData(false, -0.112, +0.842);
  //alignment_corrections[105] = UnitHitData(false, +0.373, +1.312);
  //alignment_corrections[125] = UnitHitData(false, -0.574, +0.316);
```

```
  //The format is:
  //  UnitHitData(false, MEAN_X, MEAN_Y)

  //T, from L to R, as on Jan's slides
  double mean_x24  = -0.465;
  double mean_x4   = -0.210;
  double mean_x104 =  0.167;
  double mean_x124 = -0.450;

  //B, from L to R
  double mean_x25  = -0.081;
  double mean_x5   = -0.112;
  double mean_x105 =  0.373;
  double mean_x125 = -0.574;

  //T, from L to R
  double mean_y24  = -0.689;
  double mean_y4   = -1.479;
  double mean_y104 = -0.916;
  double mean_y124 =  0.044;

  //B, from L to R
  double mean_y25  = 0.009;
  double mean_y5   = 0.842;
  double mean_y105 = 1.312;
  double mean_y125 = 0.316;

  // process track data
  for (const auto &tr : *RPtracks){

    CTPPSDetId rpId(tr.getRPId());
    unsigned int rpDecId = 100*rpId.arm() + 10*rpId.station() + 1*rpId.rp();

    //    std::cout<<"rpDecId= "<<rpDecId<<std::endl;
    //std::cout<<" --- RP Id --- "<<std::endl;
    //std::cout<<"rpDecId= "<<rpDecId<<std::endl;
        //std::cout<<"rpId.arm= "<<rpId.arm()<<std::endl;
        //std::cout<<"rpId.station= "<<rpId.station()<<std::endl;
        //std::cout<<"rpId.rp= "<<rpId.rp()<<std::endl;
    //std::cout<<"                          "<<std::endl;

    if(rpDecId == 4){rp_valid_004 = true; xLN = tr.getX() + mean_x4; yLN = tr.getY() + mean_y4;}
    if(rpDecId == 5){rp_valid_005 = true; xLN = tr.getX() + mean_x5; yLN = tr.getY() + mean_y5;}

    if(rpDecId == 24){rp_valid_024 = true; xLF = tr.getX() + mean_x24; yLF = tr.getY() + mean_y24
;}
    if(rpDecId == 25){rp_valid_025 = true; xLF = tr.getX() + mean_x25; yLF = tr.getY() + mean_y25
;}

    if(rpDecId == 104){rp_valid_104 = true; xRN = tr.getX() + mean_x104; yRN = tr.getY() + mean_y
104;}
    if(rpDecId == 105){rp_valid_105 = true; xRN = tr.getX() + mean_x105; yRN = tr.getY() + mean_y
105;}

    if(rpDecId == 124){rp_valid_124 = true; xRF = tr.getX() + mean_x124; yRF = tr.getY() + mean_y
124;}
    if(rpDecId == 125){rp_valid_125 = true; xRF = tr.getX() + mean_x125; yRF = tr.getY() + mean_y
125;}

    //    if(rpDecId == 4){rp_valid_004 = true; xLN = tr.getX(); yLN = tr.getY();}
    //    if(rpDecId == 5){rp_valid_005 = true; xLN = tr.getX(); yLN = tr.getY();}

    //    if(rpDecId == 24){rp_valid_024 = true; xLF = tr.getX(); yLF = tr.getY();}
    //    if(rpDecId == 25){rp_valid_025 = true; xLF = tr.getX(); yLF = tr.getY();}

    //    if(rpDecId == 104){rp_valid_104 = true; xRN = tr.getX(); yRN = tr.getY();}
    //    if(rpDecId == 105){rp_valid_105 = true; xRN = tr.getX(); yRN = tr.getY();}

    //    if(rpDecId == 124){rp_valid_124 = true; xRF = tr.getX(); yRF = tr.getY();}
    //    if(rpDecId == 125){rp_valid_125 = true; xRF = tr.getX(); yRF = tr.getY();}
  }
```

```
bool diag_top45_bot56 = rp_valid_024 && rp_valid_004 && rp_valid_105 && rp_valid_125;
bool diag_bot45_top56 = rp_valid_025 && rp_valid_005 && rp_valid_104 && rp_valid_124;

bool top45_top56      = rp_valid_024 && rp_valid_004 && rp_valid_104 && rp_valid_124;
bool bot45_bot56      = rp_valid_025 && rp_valid_005 && rp_valid_105 && rp_valid_125;

int nconf=0;
if(diag_top45_bot56) nconf++;
if(diag_bot45_top56) nconf++;
if(top45_top56) nconf++;
if(bot45_bot56) nconf++;

histosTH1F["hnconf"]->Fill(nconf);
if(nconf != 1) return;

//  bool diag=false;
//  if(diag_top45_bot56 || diag_bot45_top56) diag = true;

//Topol
//1 - TB, 2 - BT
//3 - TT, 4 - BB

int tb=-1;

if(diag_top45_bot56) tb=0;
if(diag_bot45_top56) tb=1;
if(top45_top56) tb=2;
if(bot45_bot56) tb=3;

histosTH1F["hconf"]->Fill(tb);


// ----- single-arm kinematics reconstruction -----

double ThxR, ThyR, ThxL, ThyL;//, xVtxL, xVtxR;

//  double D_x_L = - v_x_L_1_F * L_x_L_2_F + v_x_L_2_F * L_x_L_1_F;
//sign convention
double D_x_L = + v_x_L_1_F * L_x_L_2_F - v_x_L_2_F * L_x_L_1_F;
ThxL = (v_x_L_1_F * xLF - v_x_L_2_F * xLN) / D_x_L;
//  xVtxL = (- xLN * L_x_L_2_F + xLF * L_x_L_1_F) / D_x_L;

double D_x_R = + v_x_R_1_F * L_x_R_2_F - v_x_R_2_F * L_x_R_1_F;
ThxR = (v_x_R_1_F * xRF - v_x_R_2_F * xRN) / D_x_R;
//  xVtxR = (+ xRN * L_x_R_2_F - xRF * L_x_R_1_F) / D_x_R;

//  double th_y_L_1_F = - yLN / L_y_L_1_F;
//  double th_y_L_2_F = - yLF / L_y_L_2_F;
// sign convention
double th_y_L_1_F = + yLN / L_y_L_1_F;
double th_y_L_2_F = + yLF / L_y_L_2_F;
ThyL = (th_y_L_1_F + th_y_L_2_F) / 2.;

double th_y_R_1_F = + yRN / L_y_R_1_F;
double th_y_R_2_F = + yRF / L_y_R_2_F;
ThyR = (th_y_R_1_F + th_y_R_2_F) / 2.;

// ----- theta reconstruction -----
//  double th_sq = k.th_x*k.th_x + k.th_y*k.th_y;
//  k.th = sqrt(th_sq);
//  k.phi = atan2(k.th_y, k.th_x);

// t reconstruction
//  k.t_x = env.p*env.p * k.th_x * k.th_x;
//  k.t_y = env.p*env.p * k.th_y * k.th_y;
//  k.t = k.t_x + k.t_y;

// Correct residual shifts in thx (only, not needed thy)
// 2015
//    if(specialreco) ThxL=rec_proton_left->thx-5.04e-5;
// 2018
// was on during the run for Express
```

```
// Gauss fit: shift xL+xR = -1.80371e-04
// ThxR += 1.815e-04;

//my calculations from shift in dpx/6500

double a_off =  0.000002386 ; //TB
double b_off = -0.000006593 ; //BT
double c_off = -0.000007524 ; //TT
double d_off =  0.000003268 ; //BB

if(tb==0){ThxL += 0. ;           ThxR += a_off ;}//TB
if(tb==1){ThxL += (b_off-c_off); ThxR += c_off ;}//BT
if(tb==2){ThxL += 0. ;           ThxR += c_off ;}//TT
if(tb==3){ThxL += (d_off-a_off); ThxR += a_off ;}//BB


histosTH1F["hthxEla"]->Fill(ThxL+ThxR);
histosTH1F["hthyEla"]->Fill(ThyL+ThyR);


histosTH2F["hthx2DIM"]->Fill(ThxL,ThxR);
histosTH2F["hthythx2DIM"]->Fill(ThxL+ThxR,ThyL+ThyR);

//-------------------------------------------

bool isElastic = false;

// 5 sigma
if(TMath::Abs(ThyL+ThyR)< 15e-6 &&
   TMath::Abs(ThxL+ThxR)<45e-6) isElastic=true;

if(isElastic) return;


histosTH1F["hthxEla2"]->Fill(ThxL+ThxR);
histosTH1F["hthyEla2"]->Fill(ThyL+ThyR);

//-------------------------------------------
// after vtx cut, proton cut, elastic cut

if(runnr == 319104) histosTH1F["hLS104"]->Fill(LS);
if(runnr == 319124) histosTH1F["hLS124"]->Fill(LS);
if(runnr == 319125) histosTH1F["hLS125"]->Fill(LS);
if(runnr == 319159) histosTH1F["hLS159"]->Fill(LS);
if(runnr == 319174) histosTH1F["hLS174"]->Fill(LS);
if(runnr == 319175) histosTH1F["hLS175"]->Fill(LS);
if(runnr == 319176) histosTH1F["hLS176"]->Fill(LS);
if(runnr == 319177) histosTH1F["hLS177"]->Fill(LS);
if(runnr == 319190) histosTH1F["hLS190"]->Fill(LS);

if(runnr == 319222) histosTH1F["hLS222"]->Fill(LS);
if(runnr == 319223) histosTH1F["hLS223"]->Fill(LS);
if(runnr == 319254) histosTH1F["hLS254"]->Fill(LS);
if(runnr == 319255) histosTH1F["hLS255"]->Fill(LS);
if(runnr == 319256) histosTH1F["hLS256"]->Fill(LS);
if(runnr == 319262) histosTH1F["hLS262"]->Fill(LS);
if(runnr == 319263) histosTH1F["hLS263"]->Fill(LS);
if(runnr == 319264) histosTH1F["hLS264"]->Fill(LS);
if(runnr == 319265) histosTH1F["hLS265"]->Fill(LS);
if(runnr == 319266) histosTH1F["hLS266"]->Fill(LS);
if(runnr == 319267) histosTH1F["hLS267"]->Fill(LS);
if(runnr == 319268) histosTH1F["hLS268"]->Fill(LS);
if(runnr == 319270) histosTH1F["hLS270"]->Fill(LS);

if(runnr == 319300) histosTH1F["hLS300"]->Fill(LS);
if(runnr == 319311) histosTH1F["hLS311"]->Fill(LS);

//------------------------------
//  if(!jsonLocal(runnr,LS)) return;

//------------------------------
// Forward protons ...Luiz
```

```
//for (const auto &rpProton : *ProtonsMultiRP ) {
/*
for (const auto &rpProton : *ProtonsSingleRP ) {

CTPPSDetId rpId((*rpProton.contributingLocalTracks().begin())->getRPId());
  unsigned int decRPId = rpId.arm() * 100 + rpId.station() * 10 + rpId.rp();

double proton_t = rpProton.t();
double proton_xi = rpProton.xi();
double proton_p = rpProton.p();
double proton_pt = rpProton.pt();
double proton_thx = rpProton.thetaX();
double proton_thy = rpProton.thetaY();
double proton_mass = rpProton.mass();

std::cout<<" --- Forward Protons RP Id --- "<<std::endl;
std::cout<<"decRPId= "<<decRPId<<std::endl;

std::cout << " *** forward proton ***" <<std::endl;
std::cout << " t    = " << proton_t <<std::endl;
std::cout << " xi   = " << proton_xi <<std::endl;
std::cout << " p    = " << proton_p <<std::endl;
std::cout << " pt   = " << proton_pt <<std::endl;
std::cout << " thx  = " << proton_thx <<std::endl;
std::cout << " thy  = " << proton_thy <<std::endl;
std::cout << " mass = " << proton_mass <<std::endl;
  }
*/
//}

//--------------------------------------------
//CMS-TOTEM matching

double TOTEMpy= 6500.*(ThyL+ThyR);
double TOTEMpx=-6500.*(ThxL+ThxR);

//  double TOTEMpt1= TMath::Sqrt(pow(TOTEMpx1,2)+pow(TOTEMpy1,2));
//  double TOTEMpt2= TMath::Sqrt(pow(TOTEMpx2,2)+pow(TOTEMpy2,2));

double TOTEMphiL = TMath::ATan2(ThyL,ThxL);
double TOTEMphiR = TMath::ATan2(ThyR,ThxR);

double TOTEMdphi = TOTEMphiL-TOTEMphiR;
if(TOTEMdphi<0) TOTEMdphi = TOTEMdphi + 2*TMath::Pi(); // from (-2pi,2pi) to (0,2pi)
if(TOTEMdphi>TMath::Pi()) TOTEMdphi = 2*TMath::Pi() - TOTEMdphi; // from (0,2pi) to (0,pi)

//double CMSpx=pipiRec.Px();
//double CMSpy=pipiRec.Py();

//...Ferenc
// (int & topology, pair<double,double> & pL, pair<double,double> & pR)
// pL = pair<double,double>(-6500*ThxL, 6500*ThyL);
// pR = pair<double,double>(-6500*ThxR, 6500*ThyR);
double proton_left_px  = -6500*ThxL ;
double proton_left_py  =  6500*ThyL ;
double proton_right_px = -6500*ThxR ;
double proton_right_py =  6500*ThyR ;

// pz needed
//double proton_left_p  = TMath::Sqrt((-6500*ThxL)*(-6500*ThxL)+(6500*ThyL)*(6500*ThyL));
//double proton_right_p = TMath::Sqrt((-6500*ThxR)*(-6500*ThxR)+(6500*ThyR)*(6500*ThyR));

  histosTH1F["hprotonplx"]->Fill(proton_left_px);
  histosTH1F["hprotonply"]->Fill(proton_left_py);
  histosTH1F["hprotonprx"]->Fill(proton_right_px);
  histosTH1F["hprotonpry"]->Fill(proton_right_py);

  //histosTH1F["hprotonpl"]->Fill(proton_left_p);
  //histosTH1F["hprotonpr"]->Fill(proton_right_p);

//...4-momentum transfer squared
// t1 = -p² (theta*_x_R² + theta*_y_R²)
```

```
// t2 = -p² (theta*_x_L² + theta*_y_L²)
// |-t1|
double t1 = 6500*6500*(ThxR*ThxR+ThyR*ThyR);
// |-t2|
double t2 = 6500*6500*(ThxL*ThxL+ThyL*ThyL);
// |-(t1+t2)|
double Thetax = ThxL+ThxR;
double Thetay = ThyL+ThyR;
double t12 = 6500*6500*(Thetax*Thetax+Thetay*Thetay);

  histosTH1F["ht1"]->Fill(t1);
  histosTH1F["ht2"]->Fill(t2);
  histosTH1F["ht12"]->Fill(t12);

  // |-(t1+t2)|
  double t1t2 = t1;
  histosTH1F["ht1t2"]->Fill(t1t2);
  t1t2 = t2;
  histosTH1F["ht1t2"]->Fill(t1t2);


//-------------------------------------------------------------------------------
// 2 tracks

double CMSpx2=pipiRec.Px();
double CMSpy2=pipiRec.Py();

  bool CTpxcut2 = TMath::Abs(CMSpx2+TOTEMpx)<0.15;
  bool CTpycut2 = TMath::Abs(CMSpy2+TOTEMpy)<0.06;

  bool allCuts2 = CTpxcut2 && CTpycut2;

if(ntrk==2 && totcharge==0){

  //...Luiz
  //bool CTpxcut = TMath::Abs(CMSpx+TOTEMpx)<0.15;
  //bool CTpycut = TMath::Abs(CMSpy+TOTEMpy)<0.06;

  //bool allCuts = CTpxcut && CTpycut;

  histosTH1F["hdpy2trk"]->Fill(CMSpy2+TOTEMpy);
  histosTH1F["hdpx2trk"]->Fill(CMSpx2+TOTEMpx);
  if(CTpxcut2) histosTH1F["hdpy2trkB"]->Fill(CMSpy2+TOTEMpy);
  if(CTpycut2) histosTH1F["hdpx2trkB"]->Fill(CMSpx2+TOTEMpx);

  histosTH2F["h2DIMdpy2trk"]->Fill(CMSpy2,TOTEMpy);
  histosTH2F["h2DIMdpx2trk"]->Fill(CMSpx2,TOTEMpx);

  //Mass 2 tracks
  //double mrec = pipiRec.M();
  double mrec2 = pipiRec.M();

  if(allCuts2){

    //histosTH1F["hm2"]->Fill(mrec);
    histosTH1F["hm2"]->Fill(mrec2);

    histosTH1F["hpt2"]->Fill(pi4pos1.Pt());      //????
    histosTH1F["hpt2"]->Fill(pi4neg1.Pt());

    histosTH1F["heta2"]->Fill(pi4pos1.Eta());
    histosTH1F["heta2"]->Fill(pi4neg1.Eta());

    histosTH1F["hdphi2"]->Fill(TOTEMdphi);

  }

}

//-------------------------------------------------------------------------------
// 4 tracks
```

```
  double CMSpx4=pi4Rec.Px();
  double CMSpy4=pi4Rec.Py();

    bool CTpxcut4 = TMath::Abs(CMSpx4+TOTEMpx)<0.15;
    bool CTpycut4 = TMath::Abs(CMSpy4+TOTEMpy)<0.06;

    bool allCuts4 = CTpxcut4 && CTpycut4;

  if(ntrk==4 && totcharge==0){

    //...Luiz
    //bool CTpxcut4 = TMath::Abs(CMSpx4+TOTEMpx)<0.15;
    //bool CTpycut4 = TMath::Abs(CMSpy4+TOTEMpy)<0.06;

    //bool allCuts4 = CTpxcut4 && CTpycut4;

    histosTH1F["hdpy4trk"]->Fill(CMSpy4+TOTEMpy);
    histosTH1F["hdpx4trk"]->Fill(CMSpx4+TOTEMpx);
    if(CTpxcut4) histosTH1F["hdpy4trkB"]->Fill(CMSpy4+TOTEMpy);
    if(CTpycut4) histosTH1F["hdpx4trkB"]->Fill(CMSpx4+TOTEMpx);

    histosTH2F["h2DIMdpy4trk"]->Fill(CMSpy4,TOTEMpy);
    histosTH2F["h2DIMdpx4trk"]->Fill(CMSpx4,TOTEMpx);

    //------------------
    //Mass 4 tracks

    double mrec4=pi4Rec.M();

    if(allCuts4){

      histosTH1F["hm4"]->Fill(mrec4);

      histosTH1F["hpt4"]->Fill(pi4pos1.Pt());
      histosTH1F["hpt4"]->Fill(pi4neg1.Pt());
      histosTH1F["hpt4"]->Fill(pi4pos2.Pt());
      histosTH1F["hpt4"]->Fill(pi4neg2.Pt());

      histosTH1F["heta4"]->Fill(pi4pos1.Eta());
      histosTH1F["heta4"]->Fill(pi4neg1.Eta());
      histosTH1F["heta4"]->Fill(pi4pos2.Eta());
      histosTH1F["heta4"]->Fill(pi4neg2.Eta());

      histosTH1F["hdphi4"]->Fill(TOTEMdphi);

    }

  }

    //-------------------------------------------
    //...Luiz
    //...pions
    fiducialRegion4 = (ntrk==4 && TMath::Abs(pi1.Eta())<etaCut && TMath::Abs(pi2.Eta())<etaCut &&
                  TMath::Abs(pi3.Eta())<etaCut && TMath::Abs(pi4.Eta())<etaCut);
    fiducialRegionPt4 = (ntrk==4 && pi1.Pt()>ptCut && pi2.Pt()>ptCut &&
                         pi3.Pt()>ptCut && pi4.Pt()>ptCut);
    //...kaons
    fiducialRegionK4   = (ntrk==4 && TMath::Abs(k1.Eta())<etaCut && TMath::Abs(k2.Eta())<etaCut &
&
                  TMath::Abs(k3.Eta())<etaCut && TMath::Abs(k4.Eta())<etaCut);
    fiducialRegionPtK4 = (ntrk==4 && k1.Pt()>ptCut && k2.Pt()>ptCut &&
                         k3.Pt()>ptCut && k4.Pt()>ptCut);
    //-------------------------------------------

    //...Luiz
    double mrec=pipipipiRec.M();
    double mrecKKKK=kkkkRec.M();

    //...Luiz
    histosTH2F["h2dimdpyAll"]->Fill(CMSpy4,TOTEMpy);
    histosTH1F["hdpyAll"]->Fill(CMSpy4+TOTEMpy);
```

```
        if(fiducialRegion4 && fiducialRegionPt4){
        histosTH2F["h2dimdpy"]->Fill(CMSpy4,TOTEMpy);
        histosTH1F["hdpy"]->Fill(CMSpy4+TOTEMpy);
        }

   histosTH2F["h2dimdpxAll"]->Fill(CMSpx4,TOTEMpx);
   histosTH1F["hdpxAll"]->Fill(CMSpx4+TOTEMpx);

        if(fiducialRegion4 && fiducialRegionPt4){
        histosTH2F["h2dimdpx"]->Fill(CMSpx4,TOTEMpx);
        histosTH1F["hdpx"]->Fill(CMSpx4+TOTEMpx);
        }

        if(fiducialRegionK4 && fiducialRegionPtK4){
          std::cout << "Hello compiler...boo!" << std::endl;
        }


   //...Luiz
   histosTH1F["hvtx"]->Fill(isfake);
      //...Luiz
      if(ntrk==4){
        histosTH1F["hvtx2"]->Fill(isfake);
        if(fiducialRegion4 && totcharge==0) histosTH1F["hvtx3"]->Fill(isfake);
      }

      if(ntrk==4){
        if(totcharge==0){
      histosTH1F["hrimpac1"]->Fill(rimpac1);
      histosTH1F["hrimpac2"]->Fill(rimpac2);
      histosTH1F["hrimpac3"]->Fill(rimpac3);
      histosTH1F["hrimpac4"]->Fill(rimpac4);
        }
      }

  //-------------------------------------------------------------------------------

      //...number of vertices accepted

      /* Ferenc's recommendation!! commented out!

      //...very important...needed for theVees
      //......not this--> if(nvtx!=0 || nvtx!=1) continue;
      // this:
      if(nvtx!=0){
        if(nvtx!=1){
          if(nvtx!=2) continue; //...Ok! it works!
        }
      }
      */

      //if(nvtx!=1) continue;

 //-------------------------------------------------------------------------------
 // my stuff

      // M(1,2) M(3,4) M(1,3) M(2,4) M(1,4) M(2,3)
      double mrecpi1pi2=pi1pi2Rec.M();
      double mrecpi3pi4=pi3pi4Rec.M();
      double mrecpi1pi3=pi1pi3Rec.M();
      double mrecpi2pi4=pi2pi4Rec.M();
      double mrecpi1pi4=pi1pi4Rec.M();
      double mrecpi2pi3=pi2pi3Rec.M();
      //
      double mrec1234=pi1234Rec.M();
      double mrec1324=pi1324Rec.M();
      double mrec1423=pi1423Rec.M();
      //
      double ptpi1pi2=pi1pi2Rec.Pt();
      double ptpi3pi4=pi3pi4Rec.Pt();
      double ptpi1pi3=pi1pi3Rec.Pt();
      double ptpi2pi4=pi2pi4Rec.Pt();
```

```
  double ptpi1pi4=pi1pi4Rec.Pt();
  double ptpi2pi3=pi2pi3Rec.Pt();
  //
  double pzpi1pi2=pi1pi2Rec.Pz();
  double pzpi3pi4=pi3pi4Rec.Pz();
  double pzpi1pi3=pi1pi3Rec.Pz();
  double pzpi2pi4=pi2pi4Rec.Pz();
  double pzpi1pi4=pi1pi4Rec.Pz();
  double pzpi2pi3=pi2pi3Rec.Pz();
  //
  double etapi1pi2=pi1pi2Rec.Eta();
  double etapi3pi4=pi3pi4Rec.Eta();
  double etapi1pi3=pi1pi3Rec.Eta();
  double etapi2pi4=pi2pi4Rec.Eta();
  double etapi1pi4=pi1pi4Rec.Eta();
  double etapi2pi3=pi2pi3Rec.Eta();

  /*
  // fixing the mass of the pion pair
  double eneMK012 = TMath::Sqrt(ptpi1pi2*ptpi1pi2 + pzpi1pi2*pzpi1pi2 + m_k0*m_k0);
  double eneMK034 = TMath::Sqrt(ptpi3pi4*ptpi3pi4 + pzpi3pi4*pzpi3pi4 + m_k0*m_k0);

  histosTH1F["henemk012"]->Fill(eneMK012);
  histosTH1F["henemk034"]->Fill(eneMK034);
  */

//...combining pions and kaons for the event selection type = 11 (one primary & one Vee)
      /*
...first combining, then select the Q_pairs=0
pi1pi2 pi3k4
pi1pi3 pi2k4
pi2pi3 pi1k4

pi1pi2 k3pi4
pi1pi4 k3pi2
pi2pi4 k3pi1

pi1k2 pi3pi4
pi3k2 pi1pi4
pi4k2 pi1pi3

k1pi2 pi3pi4
k1pi3 pi2pi4
k1pi4 pi2pi3
      */

  double mrecpi3k4=pi3k4Rec.M();
  double mrecpi2k4=pi2k4Rec.M();
  double mrecpi1k4=pi1k4Rec.M();
  double mreck3pi4=k3pi4Rec.M();
  double mreck3pi2=k3pi2Rec.M();
  double mreck3pi1=k3pi1Rec.M();
  double mrecpi1k2=pi1k2Rec.M();
  double mrecpi3k2=pi3k2Rec.M();
  double mrecpi4k2=pi4k2Rec.M();
  double mreck1pi2=k1pi2Rec.M();
  double mreck1pi3=k1pi3Rec.M();
  double mreck1pi4=k1pi4Rec.M();
  //
  double mrecKpi = 0.0;

  // M(1,2) M(3,4) M(1,3) M(2,4) M(1,4) M(2,3) ...kaons only
  double mreck1k2=k1k2Rec.M();
  double mreck3k4=k3k4Rec.M();
  double mreck1k3=k1k3Rec.M();
  double mreck2k4=k2k4Rec.M();
  double mreck1k4=k1k4Rec.M();
  double mreck2k3=k2k3Rec.M();
  //
  double ptk1k2=k1k2Rec.Pt();
  double ptk3k4=k3k4Rec.Pt();
  double ptk1k3=k1k3Rec.Pt();
```

```
      double ptk2k4=k2k4Rec.Pt();
      double ptk1k4=k1k4Rec.Pt();
      double ptk2k3=k2k3Rec.Pt();
      //
      double etak1k2=k1k2Rec.Eta();
      double etak3k4=k3k4Rec.Eta();
      double etak1k3=k1k3Rec.Eta();
      double etak2k4=k2k4Rec.Eta();
      double etak1k4=k1k4Rec.Eta();
      double etak2k3=k2k3Rec.Eta();

//--------------------------------------------------------------------------------
// my cuts

if(fiducialRegion4 && fiducialRegionPt4 && allCuts4){

      //...ntrk vs nks
      histosTH2F["hntrknksall4"]->Fill(ntrk,nks);
      //...nvtx vs nks
      histosTH2F["hnvtxnksall4"]->Fill(nvtx,nks);


        //...K0 mass window
        double masslow = 0.49;
        double masshigh = 0.51;
        double masslow2 = 0.48;
        double masshigh2 = 0.52;
        double rholow = 0.75;
        double rhohigh = 0.79;
        double rholow2 = 0.72;
        double rhohigh2 = 0.79;

        double sigpipik0 = 0.031;  // 31 MeV = 1*sigma, 62 MeV window = 2*sigma
        double sigpipirho = 0.138; // 138 MeV window ??????????????????????


        //...cut 05       ...K0sK0s channel ...selection by mass
        if(totcharge==0){

          if(charray[0]+charray[1] == 0)
                 {
                    histosTH1F["hm4rec2OS_pi1pi2t05"]->Fill(mrecpi1pi2);
                    if(mrecpi1pi2 < masshigh && mrecpi1pi2 > masslow){
                    histosTH1F["hm4rec2OS_pi1pi2m05"]->Fill(mrecpi1pi2);
                    histosTH1F["h2OSpt1205"]->Fill(ptpi1pi2);
                    histosTH1F["h2OSeta1205"]->Fill(etapi1pi2);
                    histosTH2F["h2dim2OSpteta1205"]->Fill(ptpi1pi2,etapi1pi2);
                    }
                    histosTH1F["hm4rec2OS_pi3pi4t05"]->Fill(mrecpi3pi4);
                    if(mrecpi3pi4 < masshigh && mrecpi3pi4 > masslow){
                    histosTH1F["hm4rec2OS_pi3pi4m05"]->Fill(mrecpi3pi4);
                    histosTH1F["h2OSpt3405"]->Fill(ptpi3pi4);
                    histosTH1F["h2OSeta3405"]->Fill(etapi3pi4);
                    histosTH2F["h2dim2OSpteta3405"]->Fill(ptpi3pi4,etapi3pi4);
                    }
                    /*
                    if(pidarray[0]==pidPion && pidarray[1]==pidPion &&
                       pidarray[2]==pidPion && pidarray[3]==pidPion){
                      //
                    histosTH1F["hm4rec2OSm1234t05pid"]->Fill(mrec);
                    }
                    */
                    histosTH1F["hm4rec2OSm1234t05"]->Fill(mrec);
                    histosTH2F["h2dim2OSm12x34t05"]->Fill(mrecpi1pi2,mrecpi3pi4);
                    if(mrecpi1pi2 < masshigh && mrecpi1pi2 > masslow &&
                       mrecpi3pi4 < masshigh && mrecpi3pi4 > masslow ){
                       histosTH1F["hm4rec2OSm123405"]->Fill(mrec);
                       histosTH1F["hm4rec2OSmrec123405"]->Fill(mrec1234);
                       // testing mix-up channels
                       if(!nks){
                       histosTH1F["hm4rec2OSm123405nov"]->Fill(mrec);
                       }
```

```
                        if(nks==2){
                        histosTH1F["hm4rec2OSm123405yesv"]->Fill(mrec);
                        }
                        if(nks==1){
                        histosTH1F["hm4rec2OSm123405yes1"]->Fill(mrec);
                        }
                        //
                        if(mrec > 1.50 && mrec < 1.58){
                          histosTH1F["hm4rec2OSm123405pi1pt"]->Fill(pi1pt);
                          histosTH1F["hm4rec2OSm123405pi2pt"]->Fill(pi2pt);
                          histosTH1F["hm4rec2OSm123405pi3pt"]->Fill(pi3pt);
                          histosTH1F["hm4rec2OSm123405pi4pt"]->Fill(pi4pt);
                        }
                        histosTH2F["h2dim2OSm12x3405"]->Fill(mrecpi1pi2,mrecpi3pi4);
                      }
                      if(mrecpi1pi2 < masshigh2 && mrecpi1pi2 > masslow2 &&
                         mrecpi3pi4 < masshigh2 && mrecpi3pi4 > masslow2 ){
                         histosTH1F["hm4rec2OSm1234052"]->Fill(mrec);
                      }
                      //...| M(pi+pi-) - M(K0) | < 31 MeV = 1*sigma, mass window = 2*sigma = 62 Me
V
                      if( ( TMath::Abs(mrecpi1pi2 - m_k0) < sigpipik0 ) &&
                          ( TMath::Abs(mrecpi3pi4 - m_k0) < sigpipik0 ) ){
                         histosTH1F["hm4rec2OSm123405sig"]->Fill(mrec);
                      }
                      //...rho
                      if(mrecpi1pi2 < rhohigh && mrecpi1pi2 > rholow &&
                         mrecpi3pi4 < rhohigh && mrecpi3pi4 > rholow ){
                         histosTH1F["hm4rec2OSr123405"]->Fill(mrec);
                      }
                      if(mrecpi1pi2 < rhohigh2 && mrecpi1pi2 > rholow2 &&
                         mrecpi3pi4 < rhohigh2 && mrecpi3pi4 > rholow2 ){
                         histosTH1F["hm4rec2OSr1234052"]->Fill(mrec);
                      }
                      //...| M(pi+pi-) - M(rho) | < 138 MeV = 1*sigma, mass window = 2*sigma = 70
MeV
                      if( ( TMath::Abs(mrecpi1pi2 - m_rho) < sigpipirho ) &&
                          ( TMath::Abs(mrecpi3pi4 - m_rho) < sigpipirho ) ){
                         histosTH1F["hm4rec2OSr123405sig"]->Fill(mrec);
                      }
                   }
          if(charray[0]+charray[2] == 0)
                    {
                     histosTH1F["hm4rec2OS_pi1pi3t05"]->Fill(mrecpi1pi3);
                     if(mrecpi1pi3 < masshigh && mrecpi1pi3 > masslow){
                     histosTH1F["hm4rec2OS_pi1pi3m05"]->Fill(mrecpi1pi3);
                     histosTH1F["h2OSpt1305"]->Fill(ptpi1pi3);
                     histosTH1F["h2OSeta1305"]->Fill(etapi1pi3);
                     histosTH2F["h2dim2OSpteta1305"]->Fill(ptpi1pi3,etapi1pi3);
                     }
                     histosTH1F["hm4rec2OS_pi2pi4t05"]->Fill(mrecpi2pi4);
                     if(mrecpi2pi4 < masshigh && mrecpi2pi4 > masslow){
                     histosTH1F["hm4rec2OS_pi2pi4m05"]->Fill(mrecpi2pi4);
                     histosTH1F["h2OSpt2405"]->Fill(ptpi2pi4);
                     histosTH1F["h2OSeta2405"]->Fill(etapi2pi4);
                     histosTH2F["h2dim2OSpteta2405"]->Fill(ptpi2pi4,etapi2pi4);
                     }
                     /*
                     if(pidarray[0]==pidPion && pidarray[1]==pidPion &&
                        pidarray[2]==pidPion && pidarray[3]==pidPion){
                       //
                     histosTH1F["hm4rec2OSm1324t05pid"]->Fill(mrec);
                     }
                     */
                     histosTH1F["hm4rec2OSm1324t05"]->Fill(mrec);
                     histosTH2F["h2dim2OSm13x24t05"]->Fill(mrecpi1pi3,mrecpi2pi4);
                     if(mrecpi1pi3 < masshigh && mrecpi1pi3 > masslow &&
                        mrecpi2pi4 < masshigh && mrecpi2pi4 > masslow ){
                        histosTH1F["hm4rec2OSm132405"]->Fill(mrec);
                        histosTH1F["hm4rec2OSmrec132405"]->Fill(mrec1324);
                        // testing mix-up channels
                        if(!nks){
```

```
                        histosTH1F["hm4rec2OSm132405nov"]->Fill(mrec);
                        }
                        if(nks==2){
                        histosTH1F["hm4rec2OSm132405yesv"]->Fill(mrec);
                        }
                        if(nks==1){
                        histosTH1F["hm4rec2OSm132405yes1"]->Fill(mrec);
                        }
                        //
                        if(mrec > 1.50 && mrec < 1.58){
                          histosTH1F["hm4rec2OSm132405pi1pt"]->Fill(pi1pt);
                          histosTH1F["hm4rec2OSm132405pi2pt"]->Fill(pi2pt);
                          histosTH1F["hm4rec2OSm132405pi3pt"]->Fill(pi3pt);
                          histosTH1F["hm4rec2OSm132405pi4pt"]->Fill(pi4pt);
                        }
                     histosTH2F["h2dim2OSm13x2405"]->Fill(mrecpi1pi3,mrecpi2pi4);
                     }
                     if(mrecpi1pi3 < masshigh2 && mrecpi1pi3 > masslow2 &&
                        mrecpi2pi4 < masshigh2 && mrecpi2pi4 > masslow2 ){
                        histosTH1F["hm4rec2OSm1324052"]->Fill(mrec);
                     }
                     //...| M(pi+pi-) - M(K0) | < 31 MeV = 1*sigma, mass window = 2*sigma = 62 Me
V
                     if( ( TMath::Abs(mrecpi1pi3 - m_k0) < sigpipik0 ) &&
                         ( TMath::Abs(mrecpi2pi4 - m_k0) < sigpipik0 ) ){
                        histosTH1F["hm4rec2OSm132405sig"]->Fill(mrec);
                     }
                     //...rho
                     if(mrecpi1pi3 < rhohigh && mrecpi1pi3 > rholow &&
                        mrecpi2pi4 < rhohigh && mrecpi2pi4 > rholow ){
                        histosTH1F["hm4rec2OSr132405"]->Fill(mrec);
                     }
                     if(mrecpi1pi3 < rhohigh2 && mrecpi1pi3 > rholow2 &&
                        mrecpi2pi4 < rhohigh2 && mrecpi2pi4 > rholow2 ){
                        histosTH1F["hm4rec2OSr1324052"]->Fill(mrec);
                     }
                     //...| M(pi+pi-) - M(rho) | < 35 MeV = 1*sigma, mass window = 2*sigma = 70 M
eV
                     //...| M(pi+pi-) - M(rho) | < 138 MeV = 1*sigma, mass window = 2*sigma = 276
 MeV
                     if( ( TMath::Abs(mrecpi1pi3 - m_rho) < sigpipirho ) &&
                         ( TMath::Abs(mrecpi2pi4 - m_rho) < sigpipirho ) ){
                        histosTH1F["hm4rec2OSr132405sig"]->Fill(mrec);
                     }
                  }
               if(charray[0]+charray[3] == 0)
                  {
                     histosTH1F["hm4rec2OS_pi1pi4t05"]->Fill(mrecpi1pi4);
                     if(mrecpi1pi4 < masshigh && mrecpi1pi4 > masslow){
                     histosTH1F["hm4rec2OS_pi1pi4m05"]->Fill(mrecpi1pi4);
                     histosTH1F["h2OSpt1405"]->Fill(ptpi1pi4);
                     histosTH1F["h2OSeta1405"]->Fill(etapi1pi4);
                     histosTH2F["h2dim2OSpteta1405"]->Fill(ptpi1pi4,etapi1pi4);
                     }
                     histosTH1F["hm4rec2OS_pi2pi3t05"]->Fill(mrecpi2pi3);
                     if(mrecpi2pi3 < masshigh && mrecpi2pi3 > masslow){
                     histosTH1F["hm4rec2OS_pi2pi3m05"]->Fill(mrecpi2pi3);
                     histosTH1F["h2OSpt2305"]->Fill(ptpi2pi3);
                     histosTH1F["h2OSeta2305"]->Fill(etapi2pi3);
                     histosTH2F["h2dim2OSpteta2305"]->Fill(ptpi2pi3,etapi2pi3);
                     }
                     /*
                     if(pidarray[0]==pidPion && pidarray[1]==pidPion &&
                        pidarray[2]==pidPion && pidarray[3]==pidPion){
                       //
                     histosTH1F["hm4rec2OSm1423t05pid"]->Fill(mrec);
                     }
                     */
                     histosTH1F["hm4rec2OSm1423t05"]->Fill(mrec);
                     histosTH2F["h2dim2OSm14x23t05"]->Fill(mrecpi1pi4,mrecpi2pi3);
                     if(mrecpi1pi4 < masshigh && mrecpi1pi4 > masslow &&
                        mrecpi2pi3 < masshigh && mrecpi2pi3 > masslow ){
```

```
                        histosTH1F["hm4rec2OSm142305"]->Fill(mrec);
                        histosTH1F["hm4rec2OSmrec142305"]->Fill(mrec1423);
                        // testing mix-up channels
                        if(!nks){
                        histosTH1F["hm4rec2OSm142305nov"]->Fill(mrec);
                        }
                        if(nks==2){
                        histosTH1F["hm4rec2OSm142305yesv"]->Fill(mrec);
                        }
                        if(nks==1){
                        histosTH1F["hm4rec2OSm142305yes1"]->Fill(mrec);
                        }
                        //
                        if(mrec > 1.50 && mrec < 1.58){
                          histosTH1F["hm4rec2OSm142305pi1pt"]->Fill(pi1pt);
                          histosTH1F["hm4rec2OSm142305pi2pt"]->Fill(pi2pt);
                          histosTH1F["hm4rec2OSm142305pi3pt"]->Fill(pi3pt);
                          histosTH1F["hm4rec2OSm142305pi4pt"]->Fill(pi4pt);
                        }
                      histosTH2F["h2dim2OSm14x2305"]->Fill(mrecpi1pi4,mrecpi2pi3);
                      }
                      if(mrecpi1pi4 < masshigh2 && mrecpi1pi4 > masslow2 &&
                         mrecpi2pi3 < masshigh2 && mrecpi2pi3 > masslow2 ){
                         histosTH1F["hm4rec2OSm1423052"]->Fill(mrec);
                      }
                      //...| M(pi+pi-) - M(K0) | < 31 MeV = 1*sigma, mass window = 2*sigma = 62 Me
V
                      if( ( TMath::Abs(mrecpi1pi4 - m_k0) < sigpipik0 ) &&
                          ( TMath::Abs(mrecpi2pi3 - m_k0) < sigpipik0 ) ){
                         histosTH1F["hm4rec2OSm142305sig"]->Fill(mrec);
                      }
                      //...rho
                      if(mrecpi1pi4 < rhohigh && mrecpi1pi4 > rholow &&
                         mrecpi2pi3 < rhohigh && mrecpi2pi3 > rholow ){
                         histosTH1F["hm4rec2OSr142305"]->Fill(mrec);
                      }
                      if(mrecpi1pi4 < rhohigh2 && mrecpi1pi4 > rholow2 &&
                         mrecpi2pi3 < rhohigh2 && mrecpi2pi3 > rholow2 ){
                         histosTH1F["hm4rec2OSr1423052"]->Fill(mrec);
                      }
                      //...| M(pi+pi-) - M(rho) | < 35 MeV = 1*sigma, mass window = 2*sigma = 70 M
eV
                      if( ( TMath::Abs(mrecpi1pi4 - m_rho) < sigpipirho ) &&
                          ( TMath::Abs(mrecpi2pi3 - m_rho) < sigpipirho ) ){
                         histosTH1F["hm4rec2OSr142305sig"]->Fill(mrec);
                      }
                    }
          } //...end of cut05

          // } //...end of fiducialRegion4 && allCuts4


        //...cut 8.................theVees

/* not now 1

           //AA...using PID
           if(totcharge==0){

           //...using PID Pions & Kaons for selection=11

              //...Luiz
              //histosTH1F["hm4rec2OSvee"]->Fill(mrec);

              if(isKshort){

              //...one primary & one Vee && no Lambda //  K+pi- pi+pi-  or  K-pi+ pi+pi-
              if(nvtx==1 && nks==1 && nlam==0){

   not now 1 */
```

```
      /*
...first combining, then select the Q_pairs=0

pi1pi2 pi3k4
pi1pi3 pi2k4
pi2pi3 pi1k4

pi1pi2 k3pi4
pi1pi4 k3pi2
pi2pi4 k3pi1

pi1k2 pi3pi4
pi3k2 pi1pi4
pi4k2 pi1pi3

k1pi2 pi3pi4
k1pi3 pi2pi4
k1pi4 pi2pi3
      */
          //double mrecKpi = 0.0 ;
          //

          //...d0 is the transverse impact parameter (dxy) w.r.t. IP
          //...dz is the longitudinal impact parameter     w.r.t. IP
          //...vtxdxy is the transverse impact parameter   w.r.t. primary vertex
          //...vtxdz is the longitudinal impact parameter  w.r.t. primary vertex

/* not now 2
          //
          if(charray[0]+charray[1] == 0 && pidarray[2]==3 && pidarray[3]==2
             && isTrack3 && isTrack4 )
            {mrecKpi = mrecpi3k4 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[0]+charray[2] == 0 && pidarray[1]==3 && pidarray[3]==2
             && isTrack2 && isTrack4 )
            {mrecKpi = mrecpi2k4 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[1]+charray[2] == 0 && pidarray[0]==3 && pidarray[3]==2
             && isTrack1 && isTrack4 )
            {mrecKpi = mrecpi1k4 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          //
          if(charray[0]+charray[1] == 0 && pidarray[2]==2 && pidarray[3]==3
             && isTrack3 && isTrack4 )
            {mrecKpi = mreck3pi4 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[0]+charray[3] == 0 && pidarray[2]==2 && pidarray[1]==3
             && isTrack3 && isTrack2 )
            {mrecKpi = mreck3pi2 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[1]+charray[3] == 0 && pidarray[2]==2 && pidarray[0]==3
             && isTrack3 && isTrack1 )
            {mrecKpi = mreck3pi1 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          //
          if(charray[2]+charray[3] == 0 && pidarray[0]==3 && pidarray[1]==2
             && isTrack1 && isTrack2 )
            {mrecKpi = mrecpi1k2 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[0]+charray[3] == 0 && pidarray[2]==3 && pidarray[1]==2
             && isTrack3 && isTrack2 )
            {mrecKpi = mrecpi3k2 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[0]+charray[2] == 0 && pidarray[3]==3 && pidarray[1]==2
             && isTrack4 && isTrack2 )
            {mrecKpi = mrecpi4k2 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          //
          if(charray[2]+charray[3] == 0 && pidarray[0]==2 && pidarray[1]==3
             && isTrack1 && isTrack2 )
            {mrecKpi = mreck1pi2 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[1]+charray[3] == 0 && pidarray[0]==2 && pidarray[2]==3
             && isTrack1 && isTrack3 )
            {mrecKpi = mreck1pi3 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}
          if(charray[1]+charray[2] == 0 && pidarray[0]==2 && pidarray[3]==3
             && isTrack1 && isTrack4 )
            {mrecKpi = mreck1pi4 ; histosTH1F["hm4rec2OSvee11"]->Fill(mrecKpi);}


          //A
          if(charray[0]+charray[1] == 0 && pidarray[0]==3 && pidarray[1]==3 && isTrack1 && i
```

```
sTrack2 ) histosTH1F["hm4rec2OS_pi1pi2vee11"]->Fill(mrecpi1pi2);
              if(charray[2]+charray[3] == 0 && pidarray[2]==3 && pidarray[3]==2 && isTrack3 && i
sTrack4 ) histosTH1F["hm4rec2OS_pi3k4vee11"]->Fill(mrecpi3k4);
              ///histosTH2F["hm4dim2OS_pi1pi2_pi3k4vee11"]->Fill(mrecpi1pi2,mrecpi3k4);
              //
              if(charray[0]+charray[2] == 0 && pidarray[0]==3 && pidarray[2]==3 && isTrack1 && i
sTrack3 ) histosTH1F["hm4rec2OS_pi1pi3vee11"]->Fill(mrecpi1pi3);
              if(charray[1]+charray[3] == 0 && pidarray[1]==3 && pidarray[3]==2 && isTrack2 && i
sTrack4 ) histosTH1F["hm4rec2OS_pi2k4vee11"]->Fill(mrecpi2k4);
              ///histosTH2F["hm4dim2OS_pi1pi3_pi2k4vee11"]->Fill(mrecpi1pi3,mrecpi2k4);
              //
              if(charray[1]+charray[2] == 0 && pidarray[1]==3 && pidarray[2]==3 && isTrack2 && i
sTrack3 ) histosTH1F["hm4rec2OS_pi2pi3vee11"]->Fill(mrecpi2pi3);
              if(charray[0]+charray[3] == 0 && pidarray[0]==3 && pidarray[3]==2 && isTrack1 && i
sTrack4 ) histosTH1F["hm4rec2OS_pi1k4vee11"]->Fill(mrecpi1k4);
              ///histosTH2F["hm4dim2OS_pi2pi3_pi1k4vee11"]->Fill(mrecpi2pi3,mrecpi1k4);

              //B
              //if(charray[0]+charray[1] == 0) histosTH1F["hm4rec2OS_pi1pi2vee11"]->Fill(mrecpi1
pi2);
              if(charray[2]+charray[3] == 0 && pidarray[2]==2 && pidarray[3]==3 && isTrack3 && i
sTrack4 ) histosTH1F["hm4rec2OS_k3pi4vee11"]->Fill(mreck3pi4);
              ///histosTH2F["hm4dim2OS_pi1pi2_k3pi4vee11"]->Fill(mrecpi1pi2,mreck3pi4);
              //
              if(charray[0]+charray[3] == 0 && pidarray[0]==3 && pidarray[3]==3 && isTrack1 && i
sTrack4 ) histosTH1F["hm4rec2OS_pi1pi4vee11"]->Fill(mrecpi1pi4);
              if(charray[2]+charray[1] == 0 && pidarray[2]==2 && pidarray[1]==3 && isTrack3 && i
sTrack2 ) histosTH1F["hm4rec2OS_k3pi2vee11"]->Fill(mreck3pi2);
              ///histosTH2F["hm4dim2OS_pi1pi4_k3pi2vee11"]->Fill(mrecpi1pi4,mreck3pi2);
              //
              if(charray[1]+charray[3] == 0 && pidarray[1]==3 && pidarray[3]==3 && isTrack2 && i
sTrack4 ) histosTH1F["hm4rec2OS_pi2pi4vee11"]->Fill(mrecpi2pi4);
              if(charray[2]+charray[0] == 0 && pidarray[2]==2 && pidarray[0]==3 && isTrack3 && i
sTrack1 ) histosTH1F["hm4rec2OS_k3pi1vee11"]->Fill(mreck3pi1);
              ///histosTH2F["hm4dim2OS_pi2pi4_k3pi1vee11"]->Fill(mrecpi2pi4,mreck3pi1);

              //C
              if(charray[0]+charray[1] == 0 && pidarray[0]==3 && pidarray[1]==2 && isTrack1 && i
sTrack2 ) histosTH1F["hm4rec2OS_pi1k2vee11"]->Fill(mrecpi1k2);
              if(charray[2]+charray[3] == 0 && pidarray[2]==3 && pidarray[3]==3 && isTrack3 && i
sTrack4 ) histosTH1F["hm4rec2OS_pi3pi4vee11"]->Fill(mrecpi3pi4);
              ///histosTH2F["hm4dim2OS_pi1k2_pi3pi4vee11"]->Fill(mrecpi1k2,mrecpi3pi4);
              //
              if(charray[2]+charray[1] == 0 && pidarray[2]==3 && pidarray[1]==2 && isTrack3 && i
sTrack2 ) histosTH1F["hm4rec2OS_pi3k2vee11"]->Fill(mrecpi3k2);
              //if(charray[0]+charray[3] == 0) histosTH1F["hm4rec2OS_pi1pi4vee11"]->Fill(mrecpi1
pi4);
              ///histosTH2F["hm4dim2OS_pi3k2_pi1pi4vee11"]->Fill(mrecpi3k2,mrecpi1pi4);
              //
              if(charray[3]+charray[1] == 0 && pidarray[3]==3 && pidarray[1]==2 && isTrack4 && i
sTrack2 ) histosTH1F["hm4rec2OS_pi4k2vee11"]->Fill(mrecpi4k2);
              //if(charray[0]+charray[2] == 0) histosTH1F["hm4rec2OS_pi1pi3vee11"]->Fill(mrecpi1
pi3);
              ///histosTH2F["hm4dim2OS_pi4k2_pi1pi3vee11"]->Fill(mrecpi4k2,mrecpi1pi3);

              //D
              if(charray[0]+charray[1] == 0 && pidarray[0]==2 && pidarray[1]==3 && isTrack1 && i
sTrack2 ) histosTH1F["hm4rec2OS_k1pi2vee11"]->Fill(mreck1pi2);
              //if(charray[2]+charray[3] == 0) histosTH1F["hm4rec2OS_pi3pi4vee11"]->Fill(mrecpi3
pi4);
              ///histosTH2F["hm4dim2OS_k1pi2_pi3pi4vee11"]->Fill(mreck1pi2,mrecpi3pi4);
              //
              if(charray[0]+charray[2] == 0 && pidarray[0]==2 && pidarray[2]==3 && isTrack1 && i
sTrack3 ) histosTH1F["hm4rec2OS_k1pi3vee11"]->Fill(mreck1pi3);
              //if(charray[1]+charray[3] == 0) histosTH1F["hm4rec2OS_pi2pi4vee11"]->Fill(mrecpi2
pi4);
              ///histosTH2F["hm4dim2OS_k1pi3_pi2pi4vee11"]->Fill(mreck1pi3,mrecpi2pi4);
              //
              if(charray[0]+charray[3] == 0 && pidarray[0]==2 && pidarray[3]==3 && isTrack1 && i
sTrack4 ) histosTH1F["hm4rec2OS_k1pi4vee11"]->Fill(mreck1pi4);
              //if(charray[1]+charray[2] == 0) histosTH1F["hm4rec2OS_pi2pi3vee11"]->Fill(mrecpi2
pi3);
```

```
              ///histosTH2F["hm4dim2OS_k1pi3_pi2pi4vee11"]->Fill(mreck1pi3,mrecpi2pi4);

            } //end of nvtx=1 nks=1 nlam=0
            //} //...end of PID Pions & Kaons

          //...using PID Pions for selection=02 or 01
            if(pidarray[0]==3 && pidarray[1]==3 && pidarray[2]==3 && pidarray[3]==3)
            {
            //...no primary & two Vees
            if(nvtx==0 && nks==2){
            histosTH1F["hm4rec2OSvee02"]->Fill(mrec);
                if(charray[0]+charray[1] == 0)
                {
              histosTH1F["hm4rec2OS_pi1pi2vee02"]->Fill(mrecpi1pi2);
              histosTH1F["hm4rec2OS_pi3pi4vee02"]->Fill(mrecpi3pi4);
              histosTH2F["hm4dim2OS_pi1pi2_pi3pi4vee02"]->Fill(mrecpi1pi2,mrecpi3pi4);
                }else if(charray[0]+charray[2] == 0){
              histosTH1F["hm4rec2OS_pi1pi3vee02"]->Fill(mrecpi1pi3);
              histosTH1F["hm4rec2OS_pi2pi4vee02"]->Fill(mrecpi2pi4);
              histosTH2F["hm4dim2OS_pi1pi3_pi2pi4vee02"]->Fill(mrecpi1pi3,mrecpi2pi4);
                }
              } //end of nvtx=0 nks=2

            //...no primary & 1 Vee
            if(nvtx==0 && nks==1){
            histosTH1F["hm4rec2OSvee01"]->Fill(mrec);
                if(charray[0]+charray[1] == 0)
                {
              histosTH1F["hm4rec2OS_pi1pi2vee01"]->Fill(mrecpi1pi2);
              histosTH1F["hm4rec2OS_pi3pi4vee01"]->Fill(mrecpi3pi4);
              histosTH2F["hm4dim2OS_pi1pi2_pi3pi4vee01"]->Fill(mrecpi1pi2,mrecpi3pi4);
                }else if(charray[0]+charray[2] == 0){
              histosTH1F["hm4rec2OS_pi1pi3vee01"]->Fill(mrecpi1pi3);
              histosTH1F["hm4rec2OS_pi2pi4vee01"]->Fill(mrecpi2pi4);
              histosTH2F["hm4dim2OS_pi1pi3_pi2pi4vee01"]->Fill(mrecpi1pi3,mrecpi2pi4);
                }
              } //end of nvtx=0 nks=1

              } //...end of PID Pions
            } //...end of isKshort
           } //...end of totalcharge=0
           //AA...end of PID

not now 2 */

           //BB...no PID Pions or Kaons
           if(totcharge==0){

             //...ntrk vs nks
             histosTH2F["hntrknksq0"]->Fill(ntrk,nks);
             //...nvtx vs nks
             histosTH2F["hnvtxnksq0"]->Fill(nvtx,nks);

             if(isKshort){

               //...type:11 is double counting but selecting the best distributions can fix it
             if(nvtx==1 && nks==1){

               //double rlimit = 0.5;
               if(charray[0]+charray[1] == 0
                  && isTrack3 && isTrack4 )
                 {mrecKpi = mrecpi3k4 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
               if(charray[0]+charray[2] == 0
                  && isTrack2 && isTrack4  )
                 {mrecKpi = mrecpi2k4 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
               if(charray[1]+charray[2] == 0
                  && isTrack1 && isTrack4 )
                 {mrecKpi = mrecpi1k4 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
               //
               if(charray[0]+charray[1] == 0
                  && isTrack3 && isTrack4 )
                 {mrecKpi = mreck3pi4 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
```

```
if(charray[0]+charray[3] == 0
   && isTrack3 && isTrack2 )
  {mrecKpi = mreck3pi2 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
if(charray[1]+charray[3] == 0
   && isTrack3 && isTrack1 )
  {mrecKpi = mreck3pi1 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
//
if(charray[2]+charray[3] == 0
   && isTrack1 && isTrack2 )
  {mrecKpi = mrecpi1k2 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
if(charray[0]+charray[3] == 0
   && isTrack3 && isTrack2 )
  {mrecKpi = mrecpi3k2 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
if(charray[0]+charray[2] == 0
   && isTrack4 && isTrack2 )
  {mrecKpi = mrecpi4k2 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
//
if(charray[2]+charray[3] == 0
   && isTrack1 && isTrack2 )
  {mrecKpi = mreck1pi2 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
if(charray[1]+charray[3] == 0
   && isTrack1 && isTrack3 )
  {mrecKpi = mreck1pi3 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}
if(charray[1]+charray[2] == 0
   && isTrack1 && isTrack4 )
  {mrecKpi = mreck1pi4 ; histosTH1F["hm4rec2OSveeno11"]->Fill(mrecKpi);}

//...attention here!

//A  k4
if(charray[0]+charray[1] == 0 && isTrack3 && isTrack4 ) {
  histosTH1F["hm4rec2OS_pi1pi2k4veeno11"]->Fill(mrecpi1pi2); //a
  histosTH1F["hm4rec2OS_pi3k4veeno11"]->Fill(mrecpi3k4);}
///histosTH2F["hm4dim2OS_pi1pi2_pi3k4veeno11"]->Fill(mrecpi1pi2,mrecpi3k4);
//
if(charray[0]+charray[2] == 0 && isTrack2 && isTrack4 ) {
  histosTH1F["hm4rec2OS_pi1pi3k4veeno11"]->Fill(mrecpi1pi3); //b
  histosTH1F["hm4rec2OS_pi2k4veeno11"]->Fill(mrecpi2k4);}
///histosTH2F["hm4dim2OS_pi1pi3_pi2k4veeno11"]->Fill(mrecpi1pi3,mrecpi2k4);
//
if(charray[1]+charray[2] == 0 && isTrack1 && isTrack4 ) {
  histosTH1F["hm4rec2OS_pi2pi3k4veeno11"]->Fill(mrecpi2pi3); //c
  histosTH1F["hm4rec2OS_pi1k4veeno11"]->Fill(mrecpi1k4);}
///histosTH2F["hm4dim2OS_pi2pi3_pi1k4veeno11"]->Fill(mrecpi2pi3,mrecpi1k4);

//B  k3
if(charray[0]+charray[1] == 0 && isTrack3 && isTrack4 ) {
  histosTH1F["hm4rec2OS_pi1pi2k3veeno11"]->Fill(mrecpi1pi2); //ax2
  histosTH1F["hm4rec2OS_k3pi4veeno11"]->Fill(mreck3pi4);}
///histosTH2F["hm4dim2OS_pi1pi2_k3pi4veeno11"]->Fill(mrecpi1pi2,mreck3pi4);
//
if(charray[0]+charray[3] == 0 && isTrack3 && isTrack2 ) {
  histosTH1F["hm4rec2OS_pi1pi4k3veeno11"]->Fill(mrecpi1pi4); //d
  histosTH1F["hm4rec2OS_k3pi2veeno11"]->Fill(mreck3pi2);}
///histosTH2F["hm4dim2OS_pi1pi4_k3pi2veeno11"]->Fill(mrecpi1pi4,mreck3pi2);
//
if(charray[1]+charray[3] == 0 && isTrack3 && isTrack1 ) {
  histosTH1F["hm4rec2OS_pi2pi4k3veeno11"]->Fill(mrecpi2pi4); //e
  histosTH1F["hm4rec2OS_k3pi1veeno11"]->Fill(mreck3pi1);}
///histosTH2F["hm4dim2OS_pi2pi4_k3pi1veeno11"]->Fill(mrecpi2pi4,mreck3pi1);

//C  k2
if(charray[2]+charray[3] == 0 && isTrack1 && isTrack2 ) {
  histosTH1F["hm4rec2OS_pi1k2veeno11"]->Fill(mrecpi1k2);
  histosTH1F["hm4rec2OS_pi3pi4k2veeno11"]->Fill(mrecpi3pi4);} //f
///histosTH2F["hm4dim2OS_pi1k2_pi3pi4veeno11"]->Fill(mrecpi1k2,mrecpi3pi4);
//
if(charray[0]+charray[3] == 0 && isTrack3 && isTrack2 ) {
  histosTH1F["hm4rec2OS_pi3k2veeno11"]->Fill(mrecpi3k2);
  histosTH1F["hm4rec2OS_pi1pi4k2veeno11"]->Fill(mrecpi1pi4);} //dx2
///histosTH2F["hm4dim2OS_pi3k2_pi1pi4veeno11"]->Fill(mrecpi3k2,mrecpi1pi4);
//
```

```
            if(charray[0]+charray[2] == 0 && isTrack4 && isTrack2 ) {
              histosTH1F["hm4rec2OS_pi4k2veeno11"]->Fill(mrecpi4k2);
              histosTH1F["hm4rec2OS_pi1pi3k2veeno11"]->Fill(mrecpi1pi3);} //bx2
            ///histosTH2F["hm4dim2OS_pi4k2_pi1pi3veeno11"]->Fill(mrecpi4k2,mrecpi1pi3);

            //D  k1
            if(charray[2]+charray[3] == 0 && isTrack1 && isTrack2 ) {
              histosTH1F["hm4rec2OS_k1pi2veeno11"]->Fill(mreck1pi2);
              histosTH1F["hm4rec2OS_pi3pi4k1veeno11"]->Fill(mrecpi3pi4);} //fx2
            ///histosTH2F["hm4dim2OS_k1pi2_pi3pi4veeno11"]->Fill(mreck1pi2,mrecpi3pi4);
            //
            if(charray[1]+charray[3] == 0 && isTrack1 && isTrack3 ) {
              histosTH1F["hm4rec2OS_k1pi3veeno11"]->Fill(mreck1pi3);
              histosTH1F["hm4rec2OS_pi2pi4k1veeno11"]->Fill(mrecpi2pi4);} //ex2
            ///histosTH2F["hm4dim2OS_k1pi3_pi2pi4veeno11"]->Fill(mreck1pi3,mrecpi2pi4);
            //
            if(charray[1]+charray[2] == 0 && isTrack1 && isTrack4 ) {
              histosTH1F["hm4rec2OS_k1pi4veeno11"]->Fill(mreck1pi4);
              histosTH1F["hm4rec2OS_pi2pi3k1veeno11"]->Fill(mrecpi2pi3);} //cx2
            ///histosTH2F["hm4dim2OS_k1pi3_pi2pi4veeno11"]->Fill(mreck1pi3,mrecpi2pi4);

          } //end of nvtx=1 nks=1


          if(nvtx==0 && nks==2){

          histosTH1F["hm4rec2OSveeno02"]->Fill(mrec);
              if(charray[0]+charray[1] == 0)
              {
            histosTH1F["hm4rec2OS_pi1pi2veeno02"]->Fill(mrecpi1pi2);
            histosTH1F["hm4rec2OS_pi3pi4veeno02"]->Fill(mrecpi3pi4);
            histosTH2F["hm4dim2OS_pi1pi2_pi3pi4veeno02"]->Fill(mrecpi1pi2,mrecpi3pi4);
              }else if(charray[0]+charray[2] == 0){
            histosTH1F["hm4rec2OS_pi1pi3veeno02"]->Fill(mrecpi1pi3);
            histosTH1F["hm4rec2OS_pi2pi4veeno02"]->Fill(mrecpi2pi4);
            histosTH2F["hm4dim2OS_pi1pi3_pi2pi4veeno02"]->Fill(mrecpi1pi3,mrecpi2pi4);
              }
            } //end of nvtx=0 nks=2

          if(nvtx==0 && nks==1){

          histosTH1F["hm4rec2OSveeno01"]->Fill(mrec);
              if(charray[0]+charray[1] == 0)
              {
            histosTH1F["hm4rec2OS_pi1pi2veeno01"]->Fill(mrecpi1pi2);
            histosTH1F["hm4rec2OS_pi3pi4veeno01"]->Fill(mrecpi3pi4);
            histosTH2F["hm4dim2OS_pi1pi2_pi3pi4veeno01"]->Fill(mrecpi1pi2,mrecpi3pi4);
              }else if(charray[0]+charray[2] == 0){
            histosTH1F["hm4rec2OS_pi1pi3veeno01"]->Fill(mrecpi1pi3);
            histosTH1F["hm4rec2OS_pi2pi4veeno01"]->Fill(mrecpi2pi4);
            histosTH2F["hm4dim2OS_pi1pi3_pi2pi4veeno01"]->Fill(mrecpi1pi3,mrecpi2pi4);
              }
            } //end of nvtx=0 nks=1
          } //...end of isKshort
        } //BB...end of totalcharge=0

      //-----------end of cut 8


// >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

        //...cut k1  ...K+K-

        if(totcharge==0){
          histosTH1F["hm4rec2OSk"]->Fill(mrecKKKK);

          if(nvtx==1){

          histosTH1F["hm4rec2OS2k"]->Fill(mrecKKKK);

          //...REVIEW! LOGIC NOT OK!
```

```
          if(charray[0]+charray[1] == 0)
                  {
                     histosTH1F["hm4rec2OS_k1k2m"]->Fill(mreck1k2);
                     //
                     ////histosTH1F["h2OSpt12k"]->Fill(ptk1k2);
                     ////histosTH1F["h2OSeta12k"]->Fill(etak1k2);
                     ////histosTH2F["h2dim2OSpteta12k"]->Fill(ptk1k2,etak1k2);
                     //
                     histosTH1F["hm4rec2OS_k3k4m"]->Fill(mreck3k4);
                     //
                     ////histosTH1F["h2OSpt34k"]->Fill(ptk3k4);
                     ////histosTH1F["h2OSeta34k"]->Fill(etak3k4);
                     ////histosTH2F["h2dim2OSpteta34k"]->Fill(ptk3k4,etak3k4);
                     //
                       histosTH1F["hm4rec2OSm1234k"]->Fill(mrecKKKK);
                       histosTH2F["h2dim2OSm12x34k"]->Fill(mreck1k2,mreck3k4);
                                }
          if(charray[0]+charray[2] == 0)
                  {
                     histosTH1F["hm4rec2OS_k1k3m"]->Fill(mreck1k3);
                     //
                     ////histosTH1F["h2OSpt13k"]->Fill(ptk1k3);
                     ////histosTH1F["h2OSeta13k"]->Fill(etak1k3);
                     ////histosTH2F["h2dim2OSpteta13k"]->Fill(ptk1k3,etak1k3);
                     //
                     histosTH1F["hm4rec2OS_k2k4m"]->Fill(mreck2k4);
                     //
                     ////histosTH1F["h2OSpt24k"]->Fill(ptk2k4);
                     ////histosTH1F["h2OSeta24k"]->Fill(etak2k4);
                     ////histosTH2F["h2dim2OSpteta24k"]->Fill(ptk2k4,etak2k4);
                     //
                       histosTH1F["hm4rec2OSm1324k"]->Fill(mrecKKKK);
                       histosTH2F["h2dim2OSm13x24k"]->Fill(mreck1k3,mreck2k4);
                                }
          if(charray[0]+charray[3] == 0)
                  {
                     histosTH1F["hm4rec2OS_k1k4m"]->Fill(mreck1k4);
                     //
                     ////histosTH1F["h2OSpt14k"]->Fill(ptk1k4);
                     ////histosTH1F["h2OSeta14k"]->Fill(etak1k4);
                     ////histosTH2F["h2dim2OSpteta14k"]->Fill(ptk1k4,etak1k4);
                     //
                     histosTH1F["hm4rec2OS_k2k3m"]->Fill(mreck2k3);
                     //
                     ////histosTH1F["h2OSpt23k"]->Fill(ptk2k3);
                     ////histosTH1F["h2OSeta23k"]->Fill(etak2k3);
                     ////histosTH2F["h2dim2OSpteta23k"]->Fill(ptk2k3,etak2k3);
                     //
                       histosTH1F["hm4rec2OSm1423k"]->Fill(mrecKKKK);
                       histosTH2F["h2dim2OSm14x23k"]->Fill(mreck1k4,mreck2k3);
                                }

          } //...end of nvtx=1
        } //...end of cut k1 totalcharge


// >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>


      //...cut 1    ...nvtx==1 or 2

        if(totcharge==0){
          //...Luiz
          histosTH1F["hm4recOS"]->Fill(mrec);
          histosTH1F["hm4recOS2"]->Fill(mrec);
        }


      //...cut 2    ...Luiz

        if(totcharge==0){
```

```
            histosTH1F["hm4rec2OS"]->Fill(mrec);

            //...nvtx=1
            if(nvtx==1){
               histosTH1F["hm4rec2OS2"]->Fill(mrec);
               //...no V0    ...pure 4-pion channel
               if(!nks){
                  histosTH1F["hm4rec2OS2nov0"]->Fill(mrec);
                }
            }
            //...nvtx=1 and nks=0  : type:10
            //...does not improve the 4-pi channel, the difference is 22k events only
            if(nvtx==1 && nks==0){
               histosTH1F["hm4rec2OS2veeno10"]->Fill(mrec);
            }
         }
        // end of cut 2


  } //...end of fiducialRegion4 && allCuts4


  //--------------------------------------

#ifdef THIS_IS_AN_EVENT_EXAMPLE
   Handle<ExampleData> pIn;
   //   iEvent.getByLabel("example",pIn);
   //...Luiz
   iEvent.getByToken(exampletoken, pIn);
#endif

#ifdef THIS_IS_AN_EVENTSETUP_EXAMPLE
   ESHandle<SetupData> pSetup;
   iSetup.get<SetupRecord>().get(pSetup);
#endif

}


// ------------ method called once each job just before starting event loop  ------------
void
PromptAnalyzer::beginJob()
{
  //...Luiz
  edm::Service<TFileService> fs;

  int nbins_eta = 80;
  int nbins_pt = 100;
  int nbins_phi = 64;

  histosTH1F["hpt"] = fs->make<TH1F>("hpt","p_{T}",nbins_pt,0,5);
  histosTH1F["heta"] = fs->make<TH1F>("heta","#eta",nbins_eta,-4,4);
  histosTH1F["hphi"] = fs->make<TH1F>("hphi","#varphi",nbins_phi,-3.2,3.2);
  histosTH1F["halgo"] = fs->make<TH1F>("halgo","Algo",15,0,15.);
  histosTH1F["hnhits"] = fs->make<TH1F>("hnhits","nhits pix+strip",40,0,40.);

  histosTH1F["hlooper"] = fs->make<TH1F>("hlooper","isLooper",5,0,5);
  histosTH1F["hchi2"] = fs->make<TH1F>("hchi2","normalized #chi^{2}",1050,-50,1000.);
  histosTH1F["hd0"] = fs->make<TH1F>("hd0","d0",    2000,-10,10.);
  histosTH1F["hdz"] = fs->make<TH1F>("hdz","dz",    500,-100,100.);

  histosTH1F["hd0BS"] = fs->make<TH1F>("hd0BS","d0",2000,-10,10.);
  histosTH1F["hdzBS"] = fs->make<TH1F>("hdzBS","dz",500,-100,100.);

  histosTH1F["hntrk0"] = fs->make<TH1F>("hntrk0","Ntrk",150,0,150);
  histosTH1F["hntrk"] = fs->make<TH1F>("hntrk","Ntrk for nPixelHits>0",150,0,150);

  histosTH1F["hnvtx"] = fs->make<TH1F>("hnvtx","Nvtx",10,0,10);
  histosTH1F["hvtxx"] = fs->make<TH1F>("hvtxx","X vtx",1000,-1.,1.);
  histosTH1F["hvtxy"] = fs->make<TH1F>("hvtxy","Y vtx",1000,-1.,1.);
  histosTH1F["hvtxz"] = fs->make<TH1F>("hvtxz","Z vtx",300,-30.,30.);
```

```
  histosTH1F["hvtxchi2"] = fs->make<TH1F>("hvtxchi2","chi2 vtx",1100,-100.,1000.);

  //---------------------------
  // CTPPS

  histosTH1F["hnconf"] = fs->make<TH1F>("hnconf", "Number of configurations (TB or BT or TT or BB
)" , 5, 0., 5.);

  vector<string> labRP;
  labRP.push_back("TB"); labRP.push_back("BT"); labRP.push_back("TT"); labRP.push_back("BB");

  histosTH1F["hconf"]  = fs->make<TH1F>("hconf"," ",labRP.size(),0,labRP.size());
  for(size_t k = 0; k < labRP.size(); ++k){histosTH1F["hconf"]->GetXaxis()->SetBinLabel((k+1),lab
RP[k].c_str()); }

  //--------------------------------
  // PPS

  histosTH1F["hthyEla"] = fs->make<TH1F>("hthyEla"  ,"#theta_{Y}^{L}+#theta_{Y}^{R}", 2000 , -0.0
004 , 0.0004);
  histosTH1F["hthxEla"] = fs->make<TH1F>("hthxEla"  ,"#theta_{X}^{L}+#theta_{X}^{R}", 2000 , -0.0
004 , 0.0004);

  histosTH1F["hthyEla2"] = fs->make<TH1F>("hthyEla2"  ,"#theta_{Y}^{L}+#theta_{Y}^{R}", 2000 , -0
.0004 , 0.0004);
  histosTH1F["hthxEla2"] = fs->make<TH1F>("hthxEla2"  ,"#theta_{X}^{L}+#theta_{X}^{R}", 2000 , -0
.0004 , 0.0004);

  histosTH2F["hthx2DIM"] = fs->make<TH2F>("hthx2DIM"  , "#theta_{X}^{R} vs #theta_{X}^{L}" ,400,-
0.0004,0.0004,400,-0.0004,0.0004);
  histosTH2F["hthythx2DIM"] = fs->make<TH2F>("hthythx2DIM"  , "#theta_{Y} vs #theta_{X}" ,800,-0.
0004,0.0004,800,-0.0004,0.0004);

  // CT matching

  histosTH1F["hdpy2trk"] = fs->make<TH1F>("hdpy2trk","2trk, p^{CMS}_{Y}+p^{TOTEM}_{Y}",500,-0.5,0
.5);
  histosTH1F["hdpx2trk"] = fs->make<TH1F>("hdpx2trk","2trk, p^{CMS}_{X}+p^{TOTEM}_{X}",500,-0.5,0
.5);

  histosTH1F["hdpy2trkB"] = fs->make<TH1F>("hdpy2trkB","2trk, p^{CMS}_{Y}+p^{TOTEM}_{Y}",500,-0.5
,0.5);
  histosTH1F["hdpx2trkB"] = fs->make<TH1F>("hdpx2trkB","2trk, p^{CMS}_{X}+p^{TOTEM}_{X}",500,-0.5
,0.5);

  histosTH2F["h2DIMdpy2trk"] = fs->make<TH2F>("h2DIMdpy2trk","2trk, p^{TOTEM}_{Y} vs p^{CMS}_{Y}"
,200,-2.,2.,200,-2.,2.);
  histosTH2F["h2DIMdpx2trk"] = fs->make<TH2F>("h2DIMdpx2trk","p^{TOTEM}_{X} vs p^{CMS}_{X}",200,-
2.,2.,200,-2.,2.);

  histosTH1F["hdpy4trk"] = fs->make<TH1F>("hdpy4trk","p^{CMS}_{Y}+p^{TOTEM}_{Y}",500,-0.5,0.5);
  histosTH1F["hdpx4trk"] = fs->make<TH1F>("hdpx4trk","p^{CMS}_{X}+p^{TOTEM}_{X}",500,-0.5,0.5);

  histosTH1F["hdpy4trkB"] = fs->make<TH1F>("hdpy4trkB","p^{CMS}_{Y}+p^{TOTEM}_{Y}",500,-0.5,0.5);
  histosTH1F["hdpx4trkB"] = fs->make<TH1F>("hdpx4trkB","p^{CMS}_{X}+p^{TOTEM}_{X}",500,-0.5,0.5);

  histosTH2F["h2DIMdpy4trk"] = fs->make<TH2F>("h2DIMdpy4trk","4trk, p^{TOTEM}_{Y} vs p^{CMS}_{Y}"
,200,-2.,2.,200,-2.,2.);
  histosTH2F["h2DIMdpx4trk"] = fs->make<TH2F>("h2DIMdpx4trk","4trk, p^{TOTEM}_{X} vs p^{CMS}_{X}"
,200,-2.,2.,200,-2.,2.);

  //-------------------------------------
  // LS validation

  histosTH1F["hLS104"] = fs->make<TH1F>("hLS104","LS 319104",1850,0.,1850.);
  histosTH1F["hLS124"] = fs->make<TH1F>("hLS124","LS 319124",1850,0.,1850.);
  histosTH1F["hLS125"] = fs->make<TH1F>("hLS125","LS 319125",1850,0.,1850.);
  histosTH1F["hLS159"] = fs->make<TH1F>("hLS159","LS 319159",1850,0.,1850.);
  histosTH1F["hLS174"] = fs->make<TH1F>("hLS174","LS 319174",1850,0.,1850.);
  histosTH1F["hLS175"] = fs->make<TH1F>("hLS175","LS 319175",1850,0.,1850.);
  histosTH1F["hLS176"] = fs->make<TH1F>("hLS176","LS 319176",1850,0.,1850.);
  histosTH1F["hLS177"] = fs->make<TH1F>("hLS177","LS 319177",1850,0.,1850.);
```

```
  histosTH1F["hLS190"] = fs->make<TH1F>("hLS190","LS 319190",1850,0.,1850.);

  histosTH1F["hLS222"] = fs->make<TH1F>("hLS222","LS 319222",1850,0.,1850.);
  histosTH1F["hLS223"] = fs->make<TH1F>("hLS223","LS 319223",1850,0.,1850.);
  histosTH1F["hLS254"] = fs->make<TH1F>("hLS254","LS 319254",1850,0.,1850.);
  histosTH1F["hLS255"] = fs->make<TH1F>("hLS255","LS 319255",1850,0.,1850.);
  histosTH1F["hLS256"] = fs->make<TH1F>("hLS256","LS 319256",1850,0.,1850.);
  histosTH1F["hLS262"] = fs->make<TH1F>("hLS262","LS 319262",1850,0.,1850.);
  histosTH1F["hLS263"] = fs->make<TH1F>("hLS263","LS 319263",1850,0.,1850.);
  histosTH1F["hLS264"] = fs->make<TH1F>("hLS264","LS 319264",1850,0.,1850.);
  histosTH1F["hLS265"] = fs->make<TH1F>("hLS265","LS 319265",1850,0.,1850.);
  histosTH1F["hLS266"] = fs->make<TH1F>("hLS266","LS 319266",1850,0.,1850.);
  histosTH1F["hLS267"] = fs->make<TH1F>("hLS267","LS 319267",1850,0.,1850.);
  histosTH1F["hLS268"] = fs->make<TH1F>("hLS268","LS 319268",1850,0.,1850.);
  histosTH1F["hLS270"] = fs->make<TH1F>("hLS270","LS 319270",1850,0.,1850.);

  histosTH1F["hLS300"] = fs->make<TH1F>("hLS300","LS 319300",1850,0.,1850.);
  histosTH1F["hLS311"] = fs->make<TH1F>("hLS311","LS 319311",1850,0.,1850.);

  //-------------------------------------
  // Mass spectra

  //int massbins=1000;
  //
  //histosTH1F["hm2"]    = fs->make<TH1F>("hm2", "M_{#pi^{+}#pi^{-}} (GeV)", massbins,0.,10.);
  //histosTH1F["hm4"]    = fs->make<TH1F>("hm4", "M_{#pi^{+}#pi^{+}#pi^{-}#pi^{-}} (GeV)", massbi
ns,0.,10.);
  //...Luiz
  histosTH1F["hm2"]    = fs->make<TH1F>("hm2", "M_{#pi^{+}#pi^{-}} (GeV)",1000,0.,10.);
  histosTH1F["hm4"]    = fs->make<TH1F>("hm4", "M_{#pi^{+}#pi^{+}#pi^{-}#pi^{-}} (GeV)",1000,0.,1
0.);

  histosTH1F["hpt2"] = fs->make<TH1F>("hpt2","p_{T}",40,0.,2.);
  histosTH1F["heta2"]= fs->make<TH1F>("heta2","#eta",50,-5.,5.);

  histosTH1F["hpt4"] = fs->make<TH1F>("hpt4","p_{T}",40,0.,2.);
  histosTH1F["heta4"]= fs->make<TH1F>("heta4","#eta",50,-5.,5.);

  //-----

  histosTH1F["hdphi2"] = fs->make<TH1F>("hdphi2","#Delta#varphi_{LR}",320,0,TMath::Pi());
  histosTH1F["hdphi4"] = fs->make<TH1F>("hdphi4","#Delta#varphi_{LR}",320,0,TMath::Pi());

  std::cout<<"booked all."<<std::endl;

  //--------------------------------------------------------------------------
  //...Luiz
  //...my histograms

  //...10-micron resolution ...Luiz          4000 --> 2000 is there a limit here ?
  //**histosTH1F["hpv2dxy"] = fs->make<TH1F>("hpv2dxy","pv2dxy transverse impact parameter w.r.t.
 pv",2000,-1.,1.);
  //**histosTH1F["hpv2dz"] = fs->make<TH1F>("hpv2dz","pv2dz longitudinal impact parameter w.r.t.
pv",2000,-1.,1.);

  histosTH1F["hntrk4q0"] = fs->make<TH1F>("hntrk4q0","Ntrk for nPixelHits>0 Q=0",150,0,150);

  histosTH1F["hntrkvtx"] = fs->make<TH1F>("hntrkvtx","Ntrkvtx",150,0,150);
  histosTH1F["hntrkvtxU"] = fs->make<TH1F>("hntrkvtxU","NtrkvtxU",150,0,150);
  histosTH1F["hntrkvtx0"] = fs->make<TH1F>("hntrkvtx0","Ntrkvtx0",150,0,150);
  histosTH1F["hntrkvtx2"] = fs->make<TH1F>("hntrkvtx2","Ntrkvtx2",150,0,150);
  histosTH1F["hntrkvtx3"] = fs->make<TH1F>("hntrkvtx3","Ntrkvtx3",150,0,150);
  histosTH1F["hntrkvtx4"] = fs->make<TH1F>("hntrkvtx4","Ntrkvtx4",150,0,150);
  //histosTH1F["hntrkntrkvtx2"] = fs->make<TH1F>("hntrkntrkvtx2","Ntrk for Ntrkvtx=2",150,0,150);
  //histosTH1F["hntrk2ntrkvtx"] = fs->make<TH1F>("hntrk2ntrkvtx","Ntrkvtx for Ntrk=2",150,0,150);

  //histosTH2F["hntrkntrkvtx"] = new TH2F("hntrkntrkvtx","Ntrk vs Ntrkvtx",150,0,150,150,0,150);

  histosTH1F["hvtx"] = fs->make<TH1F>("hvtx","vtx.isFake()",2,0,2);
  //...Luiz
  histosTH1F["hvtx2"] = fs->make<TH1F>("hvtx2","vtx.isFake() 4 tracks",2,0,2);
  histosTH1F["hvtx3"] = fs->make<TH1F>("hvtx3","vtx.isFake() 4 tracks both |#eta|<2.5 and OS",2,0
```

```
,2);

  //...Kshorts
  histosTH1F["hnks"] = fs->make<TH1F>("hnks","N Kshorts",10,0,10);
  histosTH2F["hntrknks"] = fs->make<TH2F>("hntrknks","# of K0s Vees vs # of Tracks",150,0,150,10,
0,10);
  histosTH2F["hnvtxnks"] = fs->make<TH2F>("hnvtxnks","# of K0s Vees vs # of Vertices",150,0,150,1
0,0,10);
  //...all4
  histosTH2F["hntrknksall4"] = fs->make<TH2F>("hntrknksall4","# of K0s Vees vs # of Tracks all4",
150,0,150,10,0,10);
  histosTH2F["hnvtxnksall4"] = fs->make<TH2F>("hnvtxnksall4","# of K0s Vees vs # of Vertices all4
",150,0,150,10,0,10);
  //...Q=0
  histosTH2F["hntrknksq0"] = fs->make<TH2F>("hntrknksq0","# of K0s Vees vs # of Tracks Q=0",150,0
,150,10,0,10);
  histosTH2F["hnvtxnksq0"] = fs->make<TH2F>("hnvtxnksq0","# of K0s Vees vs # of Vertices Q=0",150
,0,150,10,0,10);
  //
  histosTH2F["hntrknvtx"] = fs->make<TH2F>("hntrknvtx","# of Vertices vs # of Tracks",150,0,150,1
50,0,150);
  histosTH1F["hksvertexx"] = fs->make<TH1F>("hksvertexx","K0s X vertex",1200,-30.,30.);
  histosTH1F["hksvertexy"] = fs->make<TH1F>("hksvertexy","K0s Y vertex",1200,-30.,30.);
  histosTH1F["hksvertexz"] = fs->make<TH1F>("hksvertexz","K0s Z vertex",1200,-50.,50.);
  histosTH1F["hksradius"] = fs->make<TH1F>("hksradius","K0s vertex radius",20000,0.,100.);
  histosTH1F["hksradiusv1"] = fs->make<TH1F>("hksradiusv1","K0s vertex radius v1",20000,0.,100.);
  histosTH1F["hksradiusv2"] = fs->make<TH1F>("hksradiusv2","K0s vertex radius v2",20000,0.,100.);
  histosTH1F["hks3Dradius"] = fs->make<TH1F>("hks3Dradius","K0s vertex radius 3D",20000,0.,100.);
  histosTH1F["hks3Dradiusv1"] = fs->make<TH1F>("hks3Dradiusv1","K0s vertex radius 3D v1",20000,0.
,100.);
  histosTH1F["hks3Dradiusv2"] = fs->make<TH1F>("hks3Dradiusv2","K0s vertex radius 3D v2",20000,0.
,100.);
  histosTH1F["hkslifetime"] = fs->make<TH1F>("hkslifetime","K0s lifetime",20000,0.,100.);
  histosTH1F["hkslifetimev1"] = fs->make<TH1F>("hkslifetimev1","K0s lifetime v1",20000,0.,100.);
  histosTH1F["hkslifetimev2"] = fs->make<TH1F>("hkslifetimev2","K0s lifetime v2",20000,0.,100.);
  histosTH1F["hks3Dlifetime"] = fs->make<TH1F>("hks3Dlifetime","K0s lifetime 3D",20000,0.,100.);
  histosTH1F["hks3Dlifetimev1"] = fs->make<TH1F>("hks3Dlifetimev1","K0s lifetime 3D v1",20000,0.,
100.);
  histosTH1F["hks3Dlifetimev2"] = fs->make<TH1F>("hks3Dlifetimev2","K0s lifetime 3D v2",20000,0.,
100.);
  //...2D
  histosTH2F["h2dimksxy"] = fs->make<TH2F>("h2dimksxy","K0s X vs Y vtx",300,-30.,30.,300,-30.,30.
);
  histosTH2F["h2dimksxz"] = fs->make<TH2F>("h2dimksxz","K0s X vs Z vtx",300,-30.,30.,300,-30.,30.
);
  histosTH2F["h2dimksyz"] = fs->make<TH2F>("h2dimksyz","K0s Y vs Z vtx",300,-30.,30.,300,-30.,30.
);
  //
  histosTH1F["hkspt"] = fs->make<TH1F>("hkspt","K0s pt",100,0.,5.);
  histosTH1F["hkseta"] = fs->make<TH1F>("hkseta","K0s #eta",80,-4.,4.);
  histosTH1F["hksphi"] = fs->make<TH1F>("hksphi","K0s #varphi",64,-3.2,3.2);
  histosTH1F["hksmass"] = fs->make<TH1F>("hksmass","K0s mass",250,0.,5.);
  //
  histosTH1F["hksmassv1"] = fs->make<TH1F>("hksmassv1","K0s mass 1 vertex",250,0.,5.);
  histosTH1F["hksmassv2"] = fs->make<TH1F>("hksmassv2","K0sK0s mass 2 vertices",250,0.,5.);
  //**histosTH1F["hksmassv3"] = fs->make<TH1F>("hksmassv3","K0s mass 3 vertices",250,0.,5.);
  //
  //**histosTH1F["hksrad"] = fs->make<TH1F>("hksrad","K0s radius",800,0,40.);
  //**histosTH1F["hksrad4"] = fs->make<TH1F>("hksrad4","K0s radius4",800,0,40.);
  //**histosTH1F["hksrad0"] = fs->make<TH1F>("hksrad0","K0s radius0",800,0,40.);
  //
  histosTH1F["hrimpac1"] = fs->make<TH1F>("hrimpac1","3D impact parameter 1",10000,0,10.);
  histosTH1F["hrimpac2"] = fs->make<TH1F>("hrimpac2","3D impact parameter 2",10000,0,10.);
  histosTH1F["hrimpac3"] = fs->make<TH1F>("hrimpac3","3D impact parameter 3",10000,0,10.);
  histosTH1F["hrimpac4"] = fs->make<TH1F>("hrimpac4","3D impact parameter 4",10000,0,10.);
  //
  //**histosTH1F["hntag1"] = fs->make<TH1F>("hntag1","test ntag1",10,0,10);
  //**histosTH1F["hntag2"] = fs->make<TH1F>("hntag2","test ntag2",10,0,10);
  //**histosTH1F["hntag3"] = fs->make<TH1F>("hntag3","test ntag3",10,0,10);

  //...Lambdas
  histosTH1F["hnlam"] = fs->make<TH1F>("hnlam","N Lambdas",10,0,10);
```

```
  histosTH2F["hntrknlam"] = fs->make<TH2F>("hntrknlam","# of #Lambda Vees vs # of Tracks",150,0,1
50,10,0,10);
  histosTH2F["hnvtxnlam"] = fs->make<TH2F>("hnvtxnlam","# of #Lambda Vees vs # of Vertices",150,0
,150,10,0,10);
  histosTH1F["hlamvertexx"] = fs->make<TH1F>("hlamvertexx","#Lambda X vertex",120,-30.,30.);
  histosTH1F["hlamvertexy"] = fs->make<TH1F>("hlamvertexy","#Lambda Y vertex",120,-30.,30.);
  histosTH1F["hlamvertexz"] = fs->make<TH1F>("hlamvertexz","#Lambda Z vertex",120,-30.,30.);
  histosTH1F["hlamradius"] = fs->make<TH1F>("hlamradius","#Lambda vertex radius",20000,0.,100.);
  histosTH1F["hlam3Dradius"] = fs->make<TH1F>("hlam3Dradius","#Lambda vertex 3D radius",20000,0.,
100.);
  histosTH1F["hlamlifetime"] = fs->make<TH1F>("hlamlifetime","#Lambda vertex lifetime",20000,0.,1
00.);
  histosTH1F["hlam3Dlifetime"] = fs->make<TH1F>("hlam3Dlifetime","#Lambda vertex 3D lifetime",200
00,0.,100.);
  //...2D
  histosTH2F["h2dimlamxy"] = fs->make<TH2F>("h2dimlamxy","#Lambda X vs Y vtx",300,-30.,30.,300,-3
0.,30.);
  histosTH2F["h2dimlamxz"] = fs->make<TH2F>("h2dimlamxz","#Lambda X vs Z vtx",300,-30.,30.,300,-3
0.,30.);
  histosTH2F["h2dimlamyz"] = fs->make<TH2F>("h2dimlamyz","#Lambda Y vs Z vtx",300,-30.,30.,300,-3
0.,30.);
  //
  histosTH1F["hlampt"] = fs->make<TH1F>("hlampt","#Lambda pt",100,0.,5.);
  histosTH1F["hlameta"] = fs->make<TH1F>("hlameta","#Lambda #eta",80,-4.,4.);
  histosTH1F["hlamphi"] = fs->make<TH1F>("hlamphi","#Lambda #varphi",64,-3.2,3.2);
  histosTH1F["hlammass"] = fs->make<TH1F>("hlammass","#Lambda mass",250,0.,5.);

  //...already defined
  //histosTH1F["hvtxx"] = fs->make<TH1F>("hvtxx","X vtx",1000,-1.,1.);
  //histosTH1F["hvtxy"] = fs->make<TH1F>("hvtxy","Y vtx",1000,-1.,1.);
  //**histosTH1F["hvtxx4"] = fs->make<TH1F>("hvtxx4","X vtx",1000,-1.,1.);
  //**histosTH1F["hvtxy4"] = fs->make<TH1F>("hvtxy4","Y vtx",1000,-1.,1.);
  //histosTH1F["hvtxz"] = fs->make<TH1F>("hvtxz","Z vtx",300,-30.,30.);
  //**histosTH1F["hvtxz4"] = fs->make<TH1F>("hvtxz4","Z vtx",300,-30.,30.);

  //...pair position
  //**histosTH1F["hxpi1pi2"] = fs->make<TH1F>("hxpi1pi2","X pi1pi2",10000,-100.,100.);
  //**histosTH1F["hypi1pi2"] = fs->make<TH1F>("hypi1pi2","Y pi1pi2",10000,-100.,100.);
  //**histosTH1F["hr12"] = fs->make<TH1F>("hr12","R pi1pi2",10000,0.,200.);

  //...Luiz
  //***histosTH2F["hvtx2dimxy"] = fs->make<TH2F>("hvtx2dimxy","X vs Y vtx",1000,-1.,1.,1000,-1.,1
.);
  //***histosTH2F["hvtx2dimxz"] = fs->make<TH2F>("hvtx2dimxz","X vs Z vtx",1000,-1.,1.,300,-3.,3.
);
  //***histosTH2F["hvtx2dimyz"] = fs->make<TH2F>("hvtx2dimyz","Y vs Z vtx",1000,-1.,1.,300,-3.,3.
);

  int massbins=250;

  //**histosTH1F["hm"] = fs->make<TH1F>("hm","M_{4#pi} ",massbins,0,5.);
  //...Luiz
  //**histosTH1F["hmxicut"] = fs->make<TH1F>("hmxicut","M_{4#pi} ",massbins,0,5.);

  //**histosTH1F["hm4rec"] = fs->make<TH1F>("hm4rec","M_{4#pi} ",massbins,0,5.);
  //**histosTH1F["hm4recbis"] = fs->make<TH1F>("hm4recbis","M_{4#pi}",2*massbins,0,5.);

  //...Luiz
  //**histosTH1F["hm4recPPPP"] = fs->make<TH1F>("hm4recPPPP","M_{4#pi} ",massbins,0,5.);
  //**histosTH1F["hm4recKKKK"] = fs->make<TH1F>("hm4recKKKK","M_{4K} ",massbins,0,5.);

  //...transverse impact parameter histograms d0 (dxy)
  histosTH1F["hd01"] = fs->make<TH1F>("hd01","d01 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hd02"] = fs->make<TH1F>("hd02","d02 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hd03"] = fs->make<TH1F>("hd03","d03 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hd04"] = fs->make<TH1F>("hd04","d04 ntrk=4 OS",5000,-5.,5.);
  //...longitudinal impact parameter histograms dz
  histosTH1F["hdz1"] = fs->make<TH1F>("hdz1","dz1 ntrk=4 OS",4000,-20.,20.);
  histosTH1F["hdz2"] = fs->make<TH1F>("hdz2","dz2 ntrk=4 OS",4000,-20.,20.);
  histosTH1F["hdz3"] = fs->make<TH1F>("hdz3","dz3 ntrk=4 OS",4000,-20.,20.);
  histosTH1F["hdz4"] = fs->make<TH1F>("hdz4","dz4 ntrk=4 OS",4000,-20.,20.);
  //..transverse impact parameter histograms vtxdxy
```

```
  histosTH1F["hvtxdxy1"] = fs->make<TH1F>("hvtxdxy1","vtxdxy1 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hvtxdxy2"] = fs->make<TH1F>("hvtxdxy2","vtxdxy2 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hvtxdxy3"] = fs->make<TH1F>("hvtxdxy3","vtxdxy3 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hvtxdxy4"] = fs->make<TH1F>("hvtxdxy4","vtxdxy4 ntrk=4 OS",5000,-5.,5.);
  //...longitudinal impact parameter histograms vtxdz
  histosTH1F["hvtxdz1"] = fs->make<TH1F>("hvtxdz1","vtxdz1 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hvtxdz2"] = fs->make<TH1F>("hvtxdz2","vtxdz2 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hvtxdz3"] = fs->make<TH1F>("hvtxdz3","vtxdz3 ntrk=4 OS",5000,-5.,5.);
  histosTH1F["hvtxdz4"] = fs->make<TH1F>("hvtxdz4","vtxdz4 ntrk=4 OS",5000,-5.,5.);

  //...OS-SS
  histosTH1F["hm4recOS"] = fs->make<TH1F>("hm4recOS","M_{4#pi} OS",massbins,0,5.);
  histosTH1F["hm4recOS2"] = fs->make<TH1F>("hm4recOS2","M_{4#pi} OS",2.0*massbins,0,10.);
  //
  //**histosTH1F["hm4recSS"] = fs->make<TH1F>("hm4recSS","M_{4#pi} SS",massbins,0,5.);
  //
  //**histosTH1F["hm4recOS_diag"] = fs->make<TH1F>("hm4recOS_diag","M_{4#pi} TB/BT OS",massbins,0
,5.);
  //**histosTH1F["hm4recSS_diag"] = fs->make<TH1F>("hm4recSS_diag","M_{4#pi} TB/BT SS",massbins,0
,5.);
  //**histosTH1F["hm4recOS_ttbb"] = fs->make<TH1F>("hm4recOS_ttbb","M_{4#pi} TT/BB OS",massbins,0
,5.);
  //**histosTH1F["hm4recSS_ttbb"] = fs->make<TH1F>("hm4recSS_ttbb","M_{4#pi} TT/BB SS",massbins,0
,5.);

  //...2OS-2SS
  histosTH1F["hm4rec2OS"] = fs->make<TH1F>("hm4rec2OS","M_{4#pi} OS",massbins,0,5.);
  histosTH1F["hm4rec2OSk"] = fs->make<TH1F>("hm4rec2OSk","M_{4K} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OSrejKp"] = fs->make<TH1F>("hm4rec2OSrejKp","M_{4#pi} OS rejecting at le
ast one K or p",massbins,0,5.);
  //**histosTH1F["hm4rec2OS2rejKp"] = fs->make<TH1F>("hm4rec2OS2rejKp","M_{4#pi} OS nvtx=1 reject
ing at least one K or p",massbins,0,5.);
  //**histosTH1F["hm4rec2OSrejKpu"] = fs->make<TH1F>("hm4rec2OSrejKpu","M_{4#pi} OS rejecting at
least one K or p or unknown",massbins,0,5.);
  //**histosTH1F["hm4rec2OS2rejKpu"] = fs->make<TH1F>("hm4rec2OS2rejKpu","M_{4#pi} OS nvtx=1 reje
cting at least one K or p or unknown",massbins,0,5.);
  //
  //...mass selection
  //
  //**histosTH1F["hm4rec2OSm1234"] = fs->make<TH1F>("hm4rec2OSm1234","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSm1324"] = fs->make<TH1F>("hm4rec2OSm1324","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSm1423"] = fs->make<TH1F>("hm4rec2OSm1423","M_{4#pi} OS",massbins,0,5.)
;
  //
  //**histosTH1F["hm4rec2OSm123400"] = fs->make<TH1F>("hm4rec2OSm123400","M_{4#pi} OS",massbins,0
,5.);
  //**histosTH1F["hm4rec2OSm132400"] = fs->make<TH1F>("hm4rec2OSm132400","M_{4#pi} OS",massbins,0
,5.);
  //**histosTH1F["hm4rec2OSm142300"] = fs->make<TH1F>("hm4rec2OSm142300","M_{4#pi} OS",massbins,0
,5.);
  //
  //**histosTH1F["hm4rec2OSm1234000"] = fs->make<TH1F>("hm4rec2OSm1234000","M_{4#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSm1324000"] = fs->make<TH1F>("hm4rec2OSm1324000","M_{4#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSm1423000"] = fs->make<TH1F>("hm4rec2OSm1423000","M_{4#pi} OS",massbins
,0,5.);
  //
  //**histosTH1F["hm4rec2OSm123404"] = fs->make<TH1F>("hm4rec2OSm123404","M_{4#pi} OS",massbins,0
,5.);
  //**histosTH1F["hm4rec2OSm132404"] = fs->make<TH1F>("hm4rec2OSm132404","M_{4#pi} OS",massbins,0
,5.);
  //**histosTH1F["hm4rec2OSm142304"] = fs->make<TH1F>("hm4rec2OSm142304","M_{4#pi} OS",massbins,0
,5.);
  //
  histosTH1F["hm4rec2OSm123405"] = fs->make<TH1F>("hm4rec2OSm123405","M_{4#pi} OS",massbins,0,5.)
;
  histosTH1F["hm4rec2OSm132405"] = fs->make<TH1F>("hm4rec2OSm132405","M_{4#pi} OS",massbins,0,5.)
;
  histosTH1F["hm4rec2OSm142305"] = fs->make<TH1F>("hm4rec2OSm142305","M_{4#pi} OS",massbins,0,5.)
```

```
;
  //
  histosTH1F["hm4rec2OSm1234052"] = fs->make<TH1F>("hm4rec2OSm1234052","M_{4#pi} OS range",massbi
ns,0,5.);
  histosTH1F["hm4rec2OSm1324052"] = fs->make<TH1F>("hm4rec2OSm1324052","M_{4#pi} OS range",massbi
ns,0,5.);
  histosTH1F["hm4rec2OSm1423052"] = fs->make<TH1F>("hm4rec2OSm1423052","M_{4#pi} OS range",massbi
ns,0,5.);
  //
  histosTH1F["hm4rec2OSm123405sig"] = fs->make<TH1F>("hm4rec2OSm123405sig","M_{4#pi} OS sigma",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OSm132405sig"] = fs->make<TH1F>("hm4rec2OSm132405sig","M_{4#pi} OS sigma",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OSm142305sig"] = fs->make<TH1F>("hm4rec2OSm142305sig","M_{4#pi} OS sigma",ma
ssbins,0,5.);
   //
  histosTH1F["hm4rec2OSr123405"] = fs->make<TH1F>("hm4rec2OSr123405","M_{4#pi} OS #rho#rho",massb
ins,0,5.);
  histosTH1F["hm4rec2OSr132405"] = fs->make<TH1F>("hm4rec2OSr132405","M_{4#pi} OS #rho#rho",massb
ins,0,5.);
  histosTH1F["hm4rec2OSr142305"] = fs->make<TH1F>("hm4rec2OSr142305","M_{4#pi} OS #rho#rho",massb
ins,0,5.);
   //
  histosTH1F["hm4rec2OSr1234052"] = fs->make<TH1F>("hm4rec2OSr1234052","M_{4#pi} OS #rho#rho rang
e",massbins,0,5.);
  histosTH1F["hm4rec2OSr1324052"] = fs->make<TH1F>("hm4rec2OSr1324052","M_{4#pi} OS #rho#rho rang
e",massbins,0,5.);
  histosTH1F["hm4rec2OSr1423052"] = fs->make<TH1F>("hm4rec2OSr1423052","M_{4#pi} OS #rho#rho rang
e",massbins,0,5.);
   //
  histosTH1F["hm4rec2OSr123405sig"] = fs->make<TH1F>("hm4rec2OSr123405sig","M_{4#pi} OS #rho#rho
sigma",massbins,0,5.);
  histosTH1F["hm4rec2OSr132405sig"] = fs->make<TH1F>("hm4rec2OSr132405sig","M_{4#pi} OS #rho#rho
sigma",massbins,0,5.);
  histosTH1F["hm4rec2OSr142305sig"] = fs->make<TH1F>("hm4rec2OSr142305sig","M_{4#pi} OS #rho#rho
sigma",massbins,0,5.);
  //
  histosTH1F["hm4rec2OSmrec123405"] = fs->make<TH1F>("hm4rec2OSmrec123405","M_{4#pi} OS mrec1234"
,massbins,0,5.);
  histosTH1F["hm4rec2OSmrec132405"] = fs->make<TH1F>("hm4rec2OSmrec132405","M_{4#pi} OS mrec1324"
,massbins,0,5.);
  histosTH1F["hm4rec2OSmrec142305"] = fs->make<TH1F>("hm4rec2OSmrec142305","M_{4#pi} OS mrec1423"
,massbins,0,5.);
  //
  // testing mix-up channels ...no 2V0
  histosTH1F["hm4rec2OSm123405nov"] = fs->make<TH1F>("hm4rec2OSm123405nov","M_{4#pi} OS no 2V0",m
assbins,0,5.);
  histosTH1F["hm4rec2OSm132405nov"] = fs->make<TH1F>("hm4rec2OSm132405nov","M_{4#pi} OS no 2V0",m
assbins,0,5.);
  histosTH1F["hm4rec2OSm142305nov"] = fs->make<TH1F>("hm4rec2OSm142305nov","M_{4#pi} OS no 2V0",m
assbins,0,5.);
  // testing mix-up channels ...yes 2V0
  histosTH1F["hm4rec2OSm123405yesv"] = fs->make<TH1F>("hm4rec2OSm123405yesv","M_{4#pi} OS yes 2V0
",massbins,0,5.);
  histosTH1F["hm4rec2OSm132405yesv"] = fs->make<TH1F>("hm4rec2OSm132405yesv","M_{4#pi} OS yes 2V0
",massbins,0,5.);
  histosTH1F["hm4rec2OSm142305yesv"] = fs->make<TH1F>("hm4rec2OSm142305yesv","M_{4#pi} OS yes 2V0
",massbins,0,5.);
  // testing mix-up channels ...yes 1V0
  histosTH1F["hm4rec2OSm123405yes1"] = fs->make<TH1F>("hm4rec2OSm123405yes1","M_{4#pi} OS yes 2V0
",massbins,0,5.);
  histosTH1F["hm4rec2OSm132405yes1"] = fs->make<TH1F>("hm4rec2OSm132405yes1","M_{4#pi} OS yes 2V0
",massbins,0,5.);
  histosTH1F["hm4rec2OSm142305yes1"] = fs->make<TH1F>("hm4rec2OSm142305yes1","M_{4#pi} OS yes 2V0
",massbins,0,5.);
  // fixed
  histosTH1F["hm4rec2OSm123405fix12"] = fs->make<TH1F>("hm4rec2OSm123405fix12","M_{4#pi} OS",mass
bins,0,5.);
  histosTH1F["hm4rec2OSm123405fix34"] = fs->make<TH1F>("hm4rec2OSm123405fix34","M_{4#pi} OS",mass
bins,0,5.);
  // fixed
  histosTH1F["hm4rec2OSm132405fix13"] = fs->make<TH1F>("hm4rec2OSm132405fix13","M_{4#pi} OS",mass
```

```
bins,0,5.);
  histosTH1F["hm4rec2OSm132405fix24"] = fs->make<TH1F>("hm4rec2OSm132405fix24","M_{4#pi} OS",mass
bins,0,5.);
  // fixed
  histosTH1F["hm4rec2OSm142305fix14"] = fs->make<TH1F>("hm4rec2OSm142305fix14","M_{4#pi} OS",mass
bins,0,5.);
  histosTH1F["hm4rec2OSm142305fix23"] = fs->make<TH1F>("hm4rec2OSm142305fix23","M_{4#pi} OS",mass
bins,0,5.);
  //
  histosTH1F["hm4rec2OSm1234t05"] = fs->make<TH1F>("hm4rec2OSm1234t05","M_{4#pi} OS",massbins,0,5
.);
  histosTH1F["hm4rec2OSm1324t05"] = fs->make<TH1F>("hm4rec2OSm1324t05","M_{4#pi} OS",massbins,0,5
.);
  histosTH1F["hm4rec2OSm1423t05"] = fs->make<TH1F>("hm4rec2OSm1423t05","M_{4#pi} OS",massbins,0,5
.);
  //
  //**histosTH1F["hm4rec2OSm1234t05pid"] = fs->make<TH1F>("hm4rec2OSm1234t05pid","M_{4#pi} OS",ma
ssbins,0,5.);
  //**histosTH1F["hm4rec2OSm1324t05pid"] = fs->make<TH1F>("hm4rec2OSm1324t05pid","M_{4#pi} OS",ma
ssbins,0,5.);
  //**histosTH1F["hm4rec2OSm1423t05pid"] = fs->make<TH1F>("hm4rec2OSm1423t05pid","M_{4#pi} OS",ma
ssbins,0,5.);
  //
  //...pT tracks
  histosTH1F["hm4rec2OSm123405pi1pt"] = fs->make<TH1F>("hm4rec2OSm123405pi1pt","M_{4#pi} OS pi1 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm123405pi2pt"] = fs->make<TH1F>("hm4rec2OSm123405pi2pt","M_{4#pi} OS pi2 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm123405pi3pt"] = fs->make<TH1F>("hm4rec2OSm123405pi3pt","M_{4#pi} OS pi3 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm123405pi4pt"] = fs->make<TH1F>("hm4rec2OSm123405pi4pt","M_{4#pi} OS pi4 p
T",massbins,0,5.);
  //
  histosTH1F["hm4rec2OSm132405pi1pt"] = fs->make<TH1F>("hm4rec2OSm132405pi1pt","M_{4#pi} OS pi1 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm132405pi2pt"] = fs->make<TH1F>("hm4rec2OSm132405pi2pt","M_{4#pi} OS pi2 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm132405pi3pt"] = fs->make<TH1F>("hm4rec2OSm132405pi3pt","M_{4#pi} OS pi3 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm132405pi4pt"] = fs->make<TH1F>("hm4rec2OSm132405pi4pt","M_{4#pi} OS pi4 p
T",massbins,0,5.);
  //
  histosTH1F["hm4rec2OSm142305pi1pt"] = fs->make<TH1F>("hm4rec2OSm142305pi1pt","M_{4#pi} OS pi1 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm142305pi2pt"] = fs->make<TH1F>("hm4rec2OSm142305pi2pt","M_{4#pi} OS pi2 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm142305pi3pt"] = fs->make<TH1F>("hm4rec2OSm142305pi3pt","M_{4#pi} OS pi3 p
T",massbins,0,5.);
  histosTH1F["hm4rec2OSm142305pi4pt"] = fs->make<TH1F>("hm4rec2OSm142305pi4pt","M_{4#pi} OS pi4 p
T",massbins,0,5.);
  //...end of pT tracks
  //
  //**histosTH1F["hm4rec2OSm123406"] = fs->make<TH1F>("hm4rec2OSm123406","M_{4#pi} OS",massbins,0
,5.);
  //**histosTH1F["hm4rec2OSm132406"] = fs->make<TH1F>("hm4rec2OSm132406","M_{4#pi} OS",massbins,0
,5.);
  //**histosTH1F["hm4rec2OSm142306"] = fs->make<TH1F>("hm4rec2OSm142306","M_{4#pi} OS",massbins,0
,5.);
  //
  //...cut k1
  histosTH1F["hm4rec2OSm1234k"] = fs->make<TH1F>("hm4rec2OSm1234k","M_{4K} OS",massbins,0,5.);
  histosTH1F["hm4rec2OSm1324k"] = fs->make<TH1F>("hm4rec2OSm1324k","M_{4K} OS",massbins,0,5.);
  histosTH1F["hm4rec2OSm1423k"] = fs->make<TH1F>("hm4rec2OSm1423k","M_{4K} OS",massbins,0,5.);
  //
  //...mass selection ...pions ...cut 0
  //***histosTH2F["h2dim2OSm12x34"]  = fs->make<TH2F>("h2dim2OSm12x34","M_{#pi_{1}#pi_{2}} vs M_{
#pi_{3}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSm12x34t"]  = fs->make<TH2F>("h2dim2OSm12x34t","M_{#pi_{1}#pi_{2}} vs M
_{#pi_{3}#pi_{4}} K0s",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta12"] = fs->make<TH2F>("h2dim2OSpteta12","pt_{#pi_{1}#pi_{2}} vs #
eta_{#pi_{1}#pi_{2}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta34"] = fs->make<TH2F>("h2dim2OSpteta34","pt_{#pi_{3}#pi_{4}} vs #
```

```
eta_{#pi_{3}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt12"]  = fs->make<TH1F>("h2OSpt12","pt_{#pi_{1}#pi_{2}} K0s mass windows",
100,0,5.);
  //**histosTH1F["h2OSpt34"]  = fs->make<TH1F>("h2OSpt34","pt_{#pi_{3}#pi_{4}} K0s mass windows",
100,0,5.);
  //**histosTH1F["h2OSeta12"] = fs->make<TH1F>("h2OSeta12","#eta_{#pi_{1}#pi_{2}} K0s mass window
s",100,-5.,5.);
  //**histosTH1F["h2OSeta34"] = fs->make<TH1F>("h2OSeta34","#eta_{#pi_{3}#pi_{4}} K0s mass window
s",100,-5.,5.);
  //
  //***histosTH2F["h2dim2OSm13x24"]  = fs->make<TH2F>("h2dim2OSm13x24","M_{#pi_{1}#pi_{3}} vs M_{
#pi_{2}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSm13x24t"]  = fs->make<TH2F>("h2dim2OSm13x24t","M_{#pi_{1}#pi_{3}} vs M
_{#pi_{2}#pi_{4}} K0s",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta13"] = fs->make<TH2F>("h2dim2OSpteta13","pt_{#pi_{1}#pi_{3}} vs #
eta_{#pi_{1}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta24"] = fs->make<TH2F>("h2dim2OSpteta24","pt_{#pi_{2}#pi_{4}} vs #
eta_{#pi_{2}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt13"]  = fs->make<TH1F>("h2OSpt13","pt_{#pi_{1}#pi_{3}} K0s mass windows",
100,0,5.);
  //**histosTH1F["h2OSpt24"]  = fs->make<TH1F>("h2OSpt24","pt_{#pi_{2}#pi_{4}} K0s mass windows",
100,0,5.);
  //**histosTH1F["h2OSeta13"] = fs->make<TH1F>("h2OSeta13","#eta_{#pi_{1}#pi_{3}} K0s mass window
s",100,-5.,5.);
  //**histosTH1F["h2OSeta24"] = fs->make<TH1F>("h2OSeta24","#eta_{#pi_{2}#pi_{4}} K0s mass window
s",100,-5.,5.);
  //
  //***histosTH2F["h2dim2OSm14x23"]  = fs->make<TH2F>("h2dim2OSm14x23","M_{#pi_{1}#pi_{4}} vs M_{
#pi_{2}#pi_{3}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSm14x23t"]  = fs->make<TH2F>("h2dim2OSm14x23t","M_{#pi_{1}#pi_{4}} vs M
_{#pi_{2}#pi_{3}} K0s",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta14"] = fs->make<TH2F>("h2dim2OSpteta14","pt_{#pi_{1}#pi_{4}} vs #
eta_{#pi_{1}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta23"] = fs->make<TH2F>("h2dim2OSpteta23","pt_{#pi_{2}#pi_{3}} vs #
eta_{#pi_{2}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt14"]  = fs->make<TH1F>("h2OSpt14","pt_{#pi_{1}#pi_{4}} K0s mass windows",
100,0,5.);
  //**histosTH1F["h2OSpt23"]  = fs->make<TH1F>("h2OSpt23","pt_{#pi_{2}#pi_{3}} K0s mass windows",
100,0,5.);
  //**histosTH1F["h2OSeta14"] = fs->make<TH1F>("h2OSeta14","#eta_{#pi_{1}#pi_{4}} K0s mass window
s",100,-5.,5.);
  //**histosTH1F["h2OSeta23"] = fs->make<TH1F>("h2OSeta23","#eta_{#pi_{2}#pi_{3}} K0s mass window
s",100,-5.,5.);
  //
  //...mass selection ...pions ...cut 00
  //...mass selection ...pions ...cut 000
  //
  //...mass selection ...pions ...cut 04
  //***histosTH2F["h2dim2OSm12x3404"]  = fs->make<TH2F>("h2dim2OSm12x3404","M_{#pi_{1}#pi_{2}} vs
 M_{#pi_{3}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSm12x34t04"]  = fs->make<TH2F>("h2dim2OSm12x34t04","M_{#pi_{1}#pi_{2}}
vs M_{#pi_{3}#pi_{4}} K0s",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta1204"] = fs->make<TH2F>("h2dim2OSpteta1204","pt_{#pi_{1}#pi_{2}}
vs #eta_{#pi_{1}#pi_{2}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta3404"] = fs->make<TH2F>("h2dim2OSpteta3404","pt_{#pi_{3}#pi_{4}}
vs #eta_{#pi_{3}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt1204"]  = fs->make<TH1F>("h2OSpt1204","pt_{#pi_{1}#pi_{2}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSpt3404"]  = fs->make<TH1F>("h2OSpt3404","pt_{#pi_{3}#pi_{4}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSeta1204"] = fs->make<TH1F>("h2OSeta1204","#eta_{#pi_{1}#pi_{2}} K0s mass wi
ndows",100,-5.,5.);
  //**histosTH1F["h2OSeta3404"] = fs->make<TH1F>("h2OSeta3404","#eta_{#pi_{3}#pi_{4}} K0s mass wi
ndows",100,-5.,5.);
  //
  //***histosTH2F["h2dim2OSm13x2404"]  = fs->make<TH2F>("h2dim2OSm13x2404","M_{#pi_{1}#pi_{3}} vs
 M_{#pi_{2}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSm13x24t04"]  = fs->make<TH2F>("h2dim2OSm13x24t04","M_{#pi_{1}#pi_{3}}
vs M_{#pi_{2}#pi_{4}} K0s",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta1304"] = fs->make<TH2F>("h2dim2OSpteta1304","pt_{#pi_{1}#pi_{3}}
vs #eta_{#pi_{1}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta2404"] = fs->make<TH2F>("h2dim2OSpteta2404","pt_{#pi_{2}#pi_{4}}
```

```
vs #eta_{#pi_{2}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt1304"]  = fs->make<TH1F>("h2OSpt1304","pt_{#pi_{1}#pi_{3}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSpt2404"]  = fs->make<TH1F>("h2OSpt2404","pt_{#pi_{2}#pi_{4}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSeta1304"] = fs->make<TH1F>("h2OSeta1304","#eta_{#pi_{1}#pi_{3}} K0s mass wi
ndows",100,-5.,5.);
  //**histosTH1F["h2OSeta2404"] = fs->make<TH1F>("h2OSeta2404","#eta_{#pi_{2}#pi_{4}} K0s mass wi
ndows",100,-5.,5.);
  //
  //***histosTH2F["h2dim2OSm14x2304"]  = fs->make<TH2F>("h2dim2OSm14x2304","M_{#pi_{1}#pi_{4}} vs
 M_{#pi_{2}#pi_{3}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSm14x23t04"]  = fs->make<TH2F>("h2dim2OSm14x23t04","M_{#pi_{1}#pi_{4}}
vs M_{#pi_{2}#pi_{3}} K0s",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta1404"] = fs->make<TH2F>("h2dim2OSpteta1404","pt_{#pi_{1}#pi_{4}}
vs #eta_{#pi_{1}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta2304"] = fs->make<TH2F>("h2dim2OSpteta2304","pt_{#pi_{2}#pi_{3}}
vs #eta_{#pi_{2}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt1404"]  = fs->make<TH1F>("h2OSpt1404","pt_{#pi_{1}#pi_{4}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSpt2304"]  = fs->make<TH1F>("h2OSpt2304","pt_{#pi_{2}#pi_{3}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSeta1404"] = fs->make<TH1F>("h2OSeta1404","#eta_{#pi_{1}#pi_{4}} K0s mass wi
ndows",100,-5.,5.);
  //**histosTH1F["h2OSeta2304"] = fs->make<TH1F>("h2OSeta2304","#eta_{#pi_{2}#pi_{3}} K0s mass wi
ndows",100,-5.,5.);
  //
  //...mass selection ...pions ...cut 05
  histosTH2F["h2dim2OSm12x3405"]  = fs->make<TH2F>("h2dim2OSm12x3405","M_{#pi_{1}#pi_{2}} vs M_{#
pi_{3}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  histosTH2F["h2dim2OSm12x34t05"]  = fs->make<TH2F>("h2dim2OSm12x34t05","M_{#pi_{1}#pi_{2}} vs M_
{#pi_{3}#pi_{4}} K0s",massbins,0,5.,massbins,0,5.);
  histosTH2F["h2dim2OSpteta1205"] = fs->make<TH2F>("h2dim2OSpteta1205","pt_{#pi_{1}#pi_{2}} vs #e
ta_{#pi_{1}#pi_{2}} K0s mass windows",100,0,5.,100,-5.,5.);
  histosTH2F["h2dim2OSpteta3405"] = fs->make<TH2F>("h2dim2OSpteta3405","pt_{#pi_{3}#pi_{4}} vs #e
ta_{#pi_{3}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  histosTH1F["h2OSpt1205"]  = fs->make<TH1F>("h2OSpt1205","pt_{#pi_{1}#pi_{2}} K0s mass windows",
100,0,5.);
  histosTH1F["h2OSpt3405"]  = fs->make<TH1F>("h2OSpt3405","pt_{#pi_{3}#pi_{4}} K0s mass windows",
100,0,5.);
  histosTH1F["h2OSeta1205"] = fs->make<TH1F>("h2OSeta1205","#eta_{#pi_{1}#pi_{2}} K0s mass window
s",100,-5.,5.);
  histosTH1F["h2OSeta3405"] = fs->make<TH1F>("h2OSeta3405","#eta_{#pi_{3}#pi_{4}} K0s mass window
s",100,-5.,5.);
  //
  histosTH2F["h2dim2OSm13x2405"]  = fs->make<TH2F>("h2dim2OSm13x2405","M_{#pi_{1}#pi_{3}} vs M_{#
pi_{2}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  histosTH2F["h2dim2OSm13x24t05"]  = fs->make<TH2F>("h2dim2OSm13x24t05","M_{#pi_{1}#pi_{3}} vs M_
{#pi_{2}#pi_{4}} K0s",massbins,0,5.,massbins,0,5.);
  histosTH2F["h2dim2OSpteta1305"] = fs->make<TH2F>("h2dim2OSpteta1305","pt_{#pi_{1}#pi_{3}} vs #e
ta_{#pi_{1}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  histosTH2F["h2dim2OSpteta2405"] = fs->make<TH2F>("h2dim2OSpteta2405","pt_{#pi_{2}#pi_{4}} vs #e
ta_{#pi_{2}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  histosTH1F["h2OSpt1305"]  = fs->make<TH1F>("h2OSpt1305","pt_{#pi_{1}#pi_{3}} K0s mass windows",
100,0,5.);
  histosTH1F["h2OSpt2405"]  = fs->make<TH1F>("h2OSpt2405","pt_{#pi_{2}#pi_{4}} K0s mass windows",
100,0,5.);
  histosTH1F["h2OSeta1305"] = fs->make<TH1F>("h2OSeta1305","#eta_{#pi_{1}#pi_{3}} K0s mass window
s",100,-5.,5.);
  histosTH1F["h2OSeta2405"] = fs->make<TH1F>("h2OSeta2405","#eta_{#pi_{2}#pi_{4}} K0s mass window
s",100,-5.,5.);
  //
  histosTH2F["h2dim2OSm14x2305"]  = fs->make<TH2F>("h2dim2OSm14x2305","M_{#pi_{1}#pi_{4}} vs M_{#
pi_{2}#pi_{3}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  histosTH2F["h2dim2OSm14x23t05"]  = fs->make<TH2F>("h2dim2OSm14x23t05","M_{#pi_{1}#pi_{4}} vs M_
{#pi_{2}#pi_{3}} K0s",massbins,0,5.,massbins,0,5.);
  histosTH2F["h2dim2OSpteta1405"] = fs->make<TH2F>("h2dim2OSpteta1405","pt_{#pi_{1}#pi_{4}} vs #e
ta_{#pi_{1}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  histosTH2F["h2dim2OSpteta2305"] = fs->make<TH2F>("h2dim2OSpteta2305","pt_{#pi_{2}#pi_{3}} vs #e
ta_{#pi_{2}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  histosTH1F["h2OSpt1405"]  = fs->make<TH1F>("h2OSpt1405","pt_{#pi_{1}#pi_{4}} K0s mass windows",
100,0,5.);
```

```
  histosTH1F["h2OSpt2305"]  = fs->make<TH1F>("h2OSpt2305","pt_{#pi_{2}#pi_{3}} K0s mass windows",
100,0,5.);
  histosTH1F["h2OSeta1405"] = fs->make<TH1F>("h2OSeta1405","#eta_{#pi_{1}#pi_{4}} K0s mass window
s",100,-5.,5.);
  histosTH1F["h2OSeta2305"] = fs->make<TH1F>("h2OSeta2305","#eta_{#pi_{2}#pi_{3}} K0s mass window
s",100,-5.,5.);
  //
  //...mass selection ...pions ...cut 06
  //***histosTH2F["h2dim2OSm12x3406"]  = fs->make<TH2F>("h2dim2OSm12x3406","M_{#pi_{1}#pi_{2}} vs
 M_{#pi_{3}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta1206"] = fs->make<TH2F>("h2dim2OSpteta1206","pt_{#pi_{1}#pi_{2}}
vs #eta_{#pi_{1}#pi_{2}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta3406"] = fs->make<TH2F>("h2dim2OSpteta3406","pt_{#pi_{3}#pi_{4}}
vs #eta_{#pi_{3}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt1206"]  = fs->make<TH1F>("h2OSpt1206","pt_{#pi_{1}#pi_{2}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSpt3406"]  = fs->make<TH1F>("h2OSpt3406","pt_{#pi_{3}#pi_{4}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSeta1206"] = fs->make<TH1F>("h2OSeta1206","#eta_{#pi_{1}#pi_{2}} K0s mass wi
ndows",100,-5.,5.);
  //**histosTH1F["h2OSeta3406"] = fs->make<TH1F>("h2OSeta3406","#eta_{#pi_{3}#pi_{4}} K0s mass wi
ndows",100,-5.,5.);
  //
  //***histosTH2F["h2dim2OSm13x2406"]  = fs->make<TH2F>("h2dim2OSm13x2406","M_{#pi_{1}#pi_{3}} vs
 M_{#pi_{2}#pi_{4}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta1306"] = fs->make<TH2F>("h2dim2OSpteta1306","pt_{#pi_{1}#pi_{3}}
vs #eta_{#pi_{1}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta2406"] = fs->make<TH2F>("h2dim2OSpteta2406","pt_{#pi_{2}#pi_{4}}
vs #eta_{#pi_{2}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt1306"]  = fs->make<TH1F>("h2OSpt1306","pt_{#pi_{1}#pi_{3}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSpt2406"]  = fs->make<TH1F>("h2OSpt2406","pt_{#pi_{2}#pi_{4}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSeta1306"] = fs->make<TH1F>("h2OSeta1306","#eta_{#pi_{1}#pi_{3}} K0s mass wi
ndows",100,-5.,5.);
  //**histosTH1F["h2OSeta2406"] = fs->make<TH1F>("h2OSeta2406","#eta_{#pi_{2}#pi_{4}} K0s mass wi
ndows",100,-5.,5.);
  //
  //***histosTH2F["h2dim2OSm14x2306"]  = fs->make<TH2F>("h2dim2OSm14x2306","M_{#pi_{1}#pi_{4}} vs
 M_{#pi_{2}#pi_{3}} K0s mass windows",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta1406"] = fs->make<TH2F>("h2dim2OSpteta1406","pt_{#pi_{1}#pi_{4}}
vs #eta_{#pi_{1}#pi_{4}} K0s mass windows",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta2306"] = fs->make<TH2F>("h2dim2OSpteta2306","pt_{#pi_{2}#pi_{3}}
vs #eta_{#pi_{2}#pi_{3}} K0s mass windows",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt1406"]  = fs->make<TH1F>("h2OSpt1406","pt_{#pi_{1}#pi_{4}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSpt2306"]  = fs->make<TH1F>("h2OSpt2306","pt_{#pi_{2}#pi_{3}} K0s mass windo
ws",100,0,5.);
  //**histosTH1F["h2OSeta1406"] = fs->make<TH1F>("h2OSeta1406","#eta_{#pi_{1}#pi_{4}} K0s mass wi
ndows",100,-5.,5.);
  //**histosTH1F["h2OSeta2306"] = fs->make<TH1F>("h2OSeta2306","#eta_{#pi_{2}#pi_{3}} K0s mass wi
ndows",100,-5.,5.);
  //
  //...cut k1
  histosTH2F["h2dim2OSm12x34k"]  = fs->make<TH2F>("h2dim2OSm12x34k","M_{K_{1}K_{2}} vs M_{K_{3}K_
{4}} K^{+}K^{-}",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta12k"] = fs->make<TH2F>("h2dim2OSpteta12k","pt_{K_{1}K_{2}} vs #et
a_{K_{1}K_{2}} K^{+}K^{-}",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta34k"] = fs->make<TH2F>("h2dim2OSpteta34k","pt_{K_{3}K_{4}} vs #et
a_{K_{3}K_{4}} K^{+}K^{-}",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt12k"]  = fs->make<TH1F>("h2OSpt12k","pt_{K_{1}K_{2}} K^{+}K^{-}",100,0,5.
);
  //**histosTH1F["h2OSpt34k"]  = fs->make<TH1F>("h2OSpt34k","pt_{K_{3}K_{4}} K^{+}K^{-}",100,0,5.
);
  //**histosTH1F["h2OSeta12k"] = fs->make<TH1F>("h2OSeta12k","#eta_{K_{1}K_{2}} K^{+}K^{-}",100,-
5.,5.);
  //**histosTH1F["h2OSeta34k"] = fs->make<TH1F>("h2OSeta34k","#eta_{K_{3}K_{4}} K^{+}K^{-}",100,-
5.,5.);
  //
  histosTH2F["h2dim2OSm13x24k"]  = fs->make<TH2F>("h2dim2OSm13x24k","M_{K_{1}K_{3}} vs M_{K_{2}K_
{4}} K^{+}K^{-}",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta13k"] = fs->make<TH2F>("h2dim2OSpteta13k","pt_{K_{1}K_{3}} vs #et
```

```
a_{K_{1}K_{3}} K^{+}K^{-}",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta24k"] = fs->make<TH2F>("h2dim2OSpteta24k","pt_{K_{2}K_{4}} vs #et
a_{K_{2}K_{4}} K^{+}K^{-}",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt13k"]  = fs->make<TH1F>("h2OSpt13k","pt_{K_{1}K_{3}} K^{+}K^{-}",100,0,5.
);
  //**histosTH1F["h2OSpt24k"]  = fs->make<TH1F>("h2OSpt24k","pt_{K_{2}K_{4}} K^{+}K^{-}",100,0,5.
);
  //**histosTH1F["h2OSeta13k"] = fs->make<TH1F>("h2OSeta13k","#eta_{K_{1}K_{3}} K^{+}K^{-}",100,-
5.,5.);
  //**histosTH1F["h2OSeta24k"] = fs->make<TH1F>("h2OSeta24k","#eta_{K_{2}K_{4}} K^{+}K^{-}",100,-
5.,5.);
  //
  histosTH2F["h2dim2OSm14x23k"]  = fs->make<TH2F>("h2dim2OSm14x23k","M_{K_{1}K_{4}} vs M_{K_{2}K_
{3}} K^{+}K^{-}",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["h2dim2OSpteta14k"] = fs->make<TH2F>("h2dim2OSpteta14k","pt_{K_{1}K_{4}} vs #et
a_{K_{1}K_{4}} K^{+}K^{-}",100,0,5.,100,-5.,5.);
  //***histosTH2F["h2dim2OSpteta23k"] = fs->make<TH2F>("h2dim2OSpteta23k","pt_{K_{2}K_{3}} vs #et
a_{K_{2}K_{3}} K^{+}K^{-}",100,0,5.,100,-5.,5.);
  //**histosTH1F["h2OSpt14k"]  = fs->make<TH1F>("h2OSpt14k","pt_{K_{1}K_{4}} K^{+}K^{-}",100,0,5.
);
  //**histosTH1F["h2OSpt23k"]  = fs->make<TH1F>("h2OSpt23k","pt_{K_{2}K_{3}} K^{+}K^{-}",100,0,5.
);
  //**histosTH1F["h2OSeta14k"] = fs->make<TH1F>("h2OSeta14k","#eta_{K_{1}K_{4}} K^{+}K^{-}",100,-
5.,5.);
  //**histosTH1F["h2OSeta23k"] = fs->make<TH1F>("h2OSeta23k","#eta_{K_{2}K_{3}} K^{+}K^{-}",100,-
5.,5.);

  //**histosTH1F["hm4rec2OSvee"] = fs->make<TH1F>("hm4rec2OSvee","M_{4#pi} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OSvee11"] = fs->make<TH1F>("hm4rec2OSvee11","M_{K#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvee02"] = fs->make<TH1F>("hm4rec2OSvee02","M_{4#pi} OS",2*massbins,0,1
0.);
  //**histosTH1F["hm4rec2OSvee01"] = fs->make<TH1F>("hm4rec2OSvee01","M_{2#pi} OS",massbins,0,5.)
;
  //
  histosTH1F["hm4rec2OSveeno11"] = fs->make<TH1F>("hm4rec2OSveeno11","M_{K#pi} OS",massbins,0,5.)
;
  histosTH1F["hm4rec2OSveeno02"] = fs->make<TH1F>("hm4rec2OSveeno02","M_{4#pi} OS",massbins,0,5.)
;
  histosTH1F["hm4rec2OSveeno01"] = fs->make<TH1F>("hm4rec2OSveeno01","M_{4#pi} OS",massbins,0,5.)
;

  //**histosTH1F["hm4rec2OSvee11a"] = fs->make<TH1F>("hm4rec2OSvee11a","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11b"] = fs->make<TH1F>("hm4rec2OSvee11b","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11c"] = fs->make<TH1F>("hm4rec2OSvee11c","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11d"] = fs->make<TH1F>("hm4rec2OSvee11d","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11e"] = fs->make<TH1F>("hm4rec2OSvee11e","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11f"] = fs->make<TH1F>("hm4rec2OSvee11f","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11g"] = fs->make<TH1F>("hm4rec2OSvee11g","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11h"] = fs->make<TH1F>("hm4rec2OSvee11h","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11i"] = fs->make<TH1F>("hm4rec2OSvee11i","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11j"] = fs->make<TH1F>("hm4rec2OSvee11j","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11k"] = fs->make<TH1F>("hm4rec2OSvee11k","M_{K#pi} OS",massbins,0,5
.);
  //**histosTH1F["hm4rec2OSvee11m"] = fs->make<TH1F>("hm4rec2OSvee11m","M_{K#pi} OS",massbins,0,5
.);
  //
  //**histosTH1F["hm4rec2OSveeno11a"] = fs->make<TH1F>("hm4rec2OSveeno11a","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11b"] = fs->make<TH1F>("hm4rec2OSveeno11b","M_{K#pi} OS",massbins
,0,5.);
```

```
  //**histosTH1F["hm4rec2OSveeno11c"] = fs->make<TH1F>("hm4rec2OSveeno11c","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11d"] = fs->make<TH1F>("hm4rec2OSveeno11d","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11e"] = fs->make<TH1F>("hm4rec2OSveeno11e","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11f"] = fs->make<TH1F>("hm4rec2OSveeno11f","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11g"] = fs->make<TH1F>("hm4rec2OSveeno11g","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11h"] = fs->make<TH1F>("hm4rec2OSveeno11h","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11i"] = fs->make<TH1F>("hm4rec2OSveeno11i","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11j"] = fs->make<TH1F>("hm4rec2OSveeno11j","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11k"] = fs->make<TH1F>("hm4rec2OSveeno11k","M_{K#pi} OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OSveeno11m"] = fs->make<TH1F>("hm4rec2OSveeno11m","M_{K#pi} OS",massbins
,0,5.);

  //**histosTH1F["hm4rec2OSvee9"] = fs->make<TH1F>("hm4rec2OSvee9","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OSvee90"] = fs->make<TH1F>("hm4rec2OSvee90","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvee91"] = fs->make<TH1F>("hm4rec2OSvee91","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvee92"] = fs->make<TH1F>("hm4rec2OSvee92","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvtx0"] = fs->make<TH1F>("hm4rec2OSvtx0","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OSvtx01"] = fs->make<TH1F>("hm4rec2OSvtx01","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvtx02"] = fs->make<TH1F>("hm4rec2OSvtx02","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvtx11"] = fs->make<TH1F>("hm4rec2OSvtx11","M_{4#pi} OS",massbins,0,5.)
;
  //**histosTH1F["hm4rec2OSvtx1"] = fs->make<TH1F>("hm4rec2OSvtx1","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OSvtx2"] = fs->make<TH1F>("hm4rec2OSvtx2","M_{4#pi} OS",massbins,0,5.);

  histosTH1F["hm4rec2OS2"] = fs->make<TH1F>("hm4rec2OS2","M_{4#pi} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS2nov0"] = fs->make<TH1F>("hm4rec2OS2nov0","M_{4#pi} OS no V0",massbins,0,5
.);
  histosTH1F["hm4rec2OS2veeno10"] = fs->make<TH1F>("hm4rec2OS2veeno10","M_{4#pi} OS type:10",mass
bins,0,5.);
  histosTH1F["hm4rec2OS2k"] = fs->make<TH1F>("hm4rec2OS2k","M_{4K} OS",2.0*massbins,0,10.);
  // 12 34 13 24  ...for now
  //**histosTH1F["hm4rec2OS_pipi"] = fs->make<TH1F>("hm4rec2OS_pipi","M_{#pi#pi} OS",massbins,0,5
.);

  //**histosTH1F["hm4rec2OS_pi1pi2"] = fs->make<TH1F>("hm4rec2OS_pi1pi2","M_{#pi_{1}#pi_{2}} OS",
massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4"] = fs->make<TH1F>("hm4rec2OS_pi3pi4","M_{#pi_{3}#pi_{4}} OS",
massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3"] = fs->make<TH1F>("hm4rec2OS_pi1pi3","M_{#pi_{1}#pi_{3}} OS",
massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4"] = fs->make<TH1F>("hm4rec2OS_pi2pi4","M_{#pi_{2}#pi_{4}} OS",
massbins,0,5.);
  //

  //...mass selection
  //**histosTH1F["hm4rec2OS_pi1pi2m"] = fs->make<TH1F>("hm4rec2OS_pi1pi2m","M_{#pi_{1}#pi_{2}} K0
s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4m"] = fs->make<TH1F>("hm4rec2OS_pi3pi4m","M_{#pi_{3}#pi_{4}} K0
s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3m"] = fs->make<TH1F>("hm4rec2OS_pi1pi3m","M_{#pi_{1}#pi_{3}} K0
s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4m"] = fs->make<TH1F>("hm4rec2OS_pi2pi4m","M_{#pi_{2}#pi_{4}} K0
s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4m"] = fs->make<TH1F>("hm4rec2OS_pi1pi4m","M_{#pi_{1}#pi_{4}} K0
s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3m"] = fs->make<TH1F>("hm4rec2OS_pi2pi3m","M_{#pi_{2}#pi_{3}} K0
s mass selection",massbins,0,5.);
  //
```

```
  //**histosTH1F["hm4rec2OS_pi1pi2t"] = fs->make<TH1F>("hm4rec2OS_pi1pi2t","M_{#pi_{1}#pi_{2}} K0
s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4t"] = fs->make<TH1F>("hm4rec2OS_pi3pi4t","M_{#pi_{3}#pi_{4}} K0
s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3t"] = fs->make<TH1F>("hm4rec2OS_pi1pi3t","M_{#pi_{1}#pi_{3}} K0
s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4t"] = fs->make<TH1F>("hm4rec2OS_pi2pi4t","M_{#pi_{2}#pi_{4}} K0
s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4t"] = fs->make<TH1F>("hm4rec2OS_pi1pi4t","M_{#pi_{1}#pi_{4}} K0
s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3t"] = fs->make<TH1F>("hm4rec2OS_pi2pi3t","M_{#pi_{2}#pi_{3}} K0
s",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2m00"] = fs->make<TH1F>("hm4rec2OS_pi1pi2m00","M_{#pi_{1}#pi_{2}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4m00"] = fs->make<TH1F>("hm4rec2OS_pi3pi4m00","M_{#pi_{3}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3m00"] = fs->make<TH1F>("hm4rec2OS_pi1pi3m00","M_{#pi_{1}#pi_{3}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4m00"] = fs->make<TH1F>("hm4rec2OS_pi2pi4m00","M_{#pi_{2}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4m00"] = fs->make<TH1F>("hm4rec2OS_pi1pi4m00","M_{#pi_{1}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3m00"] = fs->make<TH1F>("hm4rec2OS_pi2pi3m00","M_{#pi_{2}#pi_{3}
} K0s mass selection",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2m000"] = fs->make<TH1F>("hm4rec2OS_pi1pi2m000","M_{#pi_{1}#pi_{
2}} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4m000"] = fs->make<TH1F>("hm4rec2OS_pi3pi4m000","M_{#pi_{3}#pi_{
4}} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3m000"] = fs->make<TH1F>("hm4rec2OS_pi1pi3m000","M_{#pi_{1}#pi_{
3}} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4m000"] = fs->make<TH1F>("hm4rec2OS_pi2pi4m000","M_{#pi_{2}#pi_{
4}} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4m000"] = fs->make<TH1F>("hm4rec2OS_pi1pi4m000","M_{#pi_{1}#pi_{
4}} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3m000"] = fs->make<TH1F>("hm4rec2OS_pi2pi3m000","M_{#pi_{2}#pi_{
3}} K0s mass selection",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2m04"] = fs->make<TH1F>("hm4rec2OS_pi1pi2m04","M_{#pi_{1}#pi_{2}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4m04"] = fs->make<TH1F>("hm4rec2OS_pi3pi4m04","M_{#pi_{3}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3m04"] = fs->make<TH1F>("hm4rec2OS_pi1pi3m04","M_{#pi_{1}#pi_{3}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4m04"] = fs->make<TH1F>("hm4rec2OS_pi2pi4m04","M_{#pi_{2}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4m04"] = fs->make<TH1F>("hm4rec2OS_pi1pi4m04","M_{#pi_{1}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3m04"] = fs->make<TH1F>("hm4rec2OS_pi2pi3m04","M_{#pi_{2}#pi_{3}
} K0s mass selection",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2t04"] = fs->make<TH1F>("hm4rec2OS_pi1pi2t04","M_{#pi_{1}#pi_{2}
} K0s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4t04"] = fs->make<TH1F>("hm4rec2OS_pi3pi4t04","M_{#pi_{3}#pi_{4}
} K0s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3t04"] = fs->make<TH1F>("hm4rec2OS_pi1pi3t04","M_{#pi_{1}#pi_{3}
} K0s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4t04"] = fs->make<TH1F>("hm4rec2OS_pi2pi4t04","M_{#pi_{2}#pi_{4}
} K0s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4t04"] = fs->make<TH1F>("hm4rec2OS_pi1pi4t04","M_{#pi_{1}#pi_{4}
} K0s",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3t04"] = fs->make<TH1F>("hm4rec2OS_pi2pi3t04","M_{#pi_{2}#pi_{3}
} K0s",massbins,0,5.);
  //
  histosTH1F["hm4rec2OS_pi1pi2m05"] = fs->make<TH1F>("hm4rec2OS_pi1pi2m05","M_{#pi_{1}#pi_{2}} K0
s mass selection",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi3pi4m05"] = fs->make<TH1F>("hm4rec2OS_pi3pi4m05","M_{#pi_{3}#pi_{4}} K0
s mass selection",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi1pi3m05"] = fs->make<TH1F>("hm4rec2OS_pi1pi3m05","M_{#pi_{1}#pi_{3}} K0
s mass selection",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi2pi4m05"] = fs->make<TH1F>("hm4rec2OS_pi2pi4m05","M_{#pi_{2}#pi_{4}} K0
```

```
s mass selection",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi1pi4m05"] = fs->make<TH1F>("hm4rec2OS_pi1pi4m05","M_{#pi_{1}#pi_{4}} K0
s mass selection",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi2pi3m05"] = fs->make<TH1F>("hm4rec2OS_pi2pi3m05","M_{#pi_{2}#pi_{3}} K0
s mass selection",massbins,0,5.);
  //
  histosTH1F["hm4rec2OS_pi1pi2t05"] = fs->make<TH1F>("hm4rec2OS_pi1pi2t05","M_{#pi_{1}#pi_{2}} K0
s",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi3pi4t05"] = fs->make<TH1F>("hm4rec2OS_pi3pi4t05","M_{#pi_{3}#pi_{4}} K0
s",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi1pi3t05"] = fs->make<TH1F>("hm4rec2OS_pi1pi3t05","M_{#pi_{1}#pi_{3}} K0
s",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi2pi4t05"] = fs->make<TH1F>("hm4rec2OS_pi2pi4t05","M_{#pi_{2}#pi_{4}} K0
s",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi1pi4t05"] = fs->make<TH1F>("hm4rec2OS_pi1pi4t05","M_{#pi_{1}#pi_{4}} K0
s",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi2pi3t05"] = fs->make<TH1F>("hm4rec2OS_pi2pi3t05","M_{#pi_{2}#pi_{3}} K0
s",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2m06"] = fs->make<TH1F>("hm4rec2OS_pi1pi2m06","M_{#pi_{1}#pi_{2}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4m06"] = fs->make<TH1F>("hm4rec2OS_pi3pi4m06","M_{#pi_{3}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3m06"] = fs->make<TH1F>("hm4rec2OS_pi1pi3m06","M_{#pi_{1}#pi_{3}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4m06"] = fs->make<TH1F>("hm4rec2OS_pi2pi4m06","M_{#pi_{2}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi4m06"] = fs->make<TH1F>("hm4rec2OS_pi1pi4m06","M_{#pi_{1}#pi_{4}
} K0s mass selection",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi3m06"] = fs->make<TH1F>("hm4rec2OS_pi2pi3m06","M_{#pi_{2}#pi_{3}
} K0s mass selection",massbins,0,5.);
  //
  //..cut k1
  histosTH1F["hm4rec2OS_k1k2m"] = fs->make<TH1F>("hm4rec2OS_k1k2m","M_{K_{1}K_{2}} K^{+}K^{-}",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OS_k3k4m"] = fs->make<TH1F>("hm4rec2OS_k3k4m","M_{K_{3}K_{4}} K^{+}K^{-}",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OS_k1k3m"] = fs->make<TH1F>("hm4rec2OS_k1k3m","M_{K_{1}K_{3}} K^{+}K^{-}",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OS_k2k4m"] = fs->make<TH1F>("hm4rec2OS_k2k4m","M_{K_{2}K_{4}} K^{+}K^{-}",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OS_k1k4m"] = fs->make<TH1F>("hm4rec2OS_k1k4m","M_{K_{1}K_{4}} K^{+}K^{-}",ma
ssbins,0,5.);
  histosTH1F["hm4rec2OS_k2k3m"] = fs->make<TH1F>("hm4rec2OS_k2k3m","M_{K_{2}K_{3}} K^{+}K^{-}",ma
ssbins,0,5.);

  //...v2
  //**histosTH1F["hm4rec2OS_pi1pi2v2"] = fs->make<TH1F>("hm4rec2OS_pi1pi2v2","M_{#pi_{1}#pi_{2}}
OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4v2"] = fs->make<TH1F>("hm4rec2OS_pi3pi4v2","M_{#pi_{3}#pi_{4}}
OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3v2"] = fs->make<TH1F>("hm4rec2OS_pi1pi3v2","M_{#pi_{1}#pi_{3}}
OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4v2"] = fs->make<TH1F>("hm4rec2OS_pi2pi4v2","M_{#pi_{2}#pi_{4}}
OS",massbins,0,5.);
  //...2dim
  //***histosTH2F["hm4dim2OS_pi1pi2_pi3pi4"] = fs->make<TH2F>("hm4dim2OS_pi1pi2_pi3pi4","M_{#pi_{
1}#pi_{2}} vs M_{#pi_{3}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["hm4dim2OS_pi1pi3_pi2pi4"] = fs->make<TH2F>("hm4dim2OS_pi1pi3_pi2pi4","M_{#pi_{
1}#pi_{3}} vs M_{#pi_{2}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //...v2
  //***histosTH2F["hm4dim2OS_pi1pi2_pi3pi4v2"] = fs->make<TH2F>("hm4dim2OS_pi1pi2_pi3pi4v2","M_{#
pi_{1}#pi_{2}} vs M_{#pi_{3}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["hm4dim2OS_pi1pi3_pi2pi4v2"] = fs->make<TH2F>("hm4dim2OS_pi1pi3_pi2pi4v2","M_{#
pi_{1}#pi_{3}} vs M_{#pi_{2}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);

  //...testing vee9  ........cut 9
  //**histosTH1F["hm4rec2OS_pi1pi2vee9"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee9","M_{#pi_{1}#pi_{
2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vee9"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee9","M_{#pi_{3}#pi_{
4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vee9"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee9","M_{#pi_{1}#pi_{
```

```
3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vee9"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee9","M_{#pi_{2}#pi_{
4}} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2vee90"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee90","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vee90"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee90","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vee90"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee90","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vee90"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee90","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2vee91"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee91","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vee91"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee91","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vee91"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee91","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vee91"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee91","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2vee92"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee92","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vee92"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee92","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vee92"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee92","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vee92"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee92","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2vtx0"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vtx0","M_{#pi_{1}#pi_{
2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vtx0"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vtx0","M_{#pi_{3}#pi_{
4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vtx0"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vtx0","M_{#pi_{1}#pi_{
3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vtx0"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vtx0","M_{#pi_{2}#pi_{
4}} OS",massbins,0,5.);
   //
  //**histosTH1F["hm4rec2OS_pi1pi2vtx01"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vtx01","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vtx01"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vtx01","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vtx01"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vtx01","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vtx01"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vtx01","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
   //
  //**histosTH1F["hm4rec2OS_pi1pi2vtx02"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vtx02","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vtx02"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vtx02","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vtx02"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vtx02","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vtx02"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vtx02","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
   //
  //**histosTH1F["hm4rec2OS_pi1pi2vtx11"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vtx11","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vtx11"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vtx11","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vtx11"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vtx11","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vtx11"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vtx11","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2vtx1"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vtx1","M_{#pi_{1}#pi_{
2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vtx1"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vtx1","M_{#pi_{3}#pi_{
4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vtx1"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vtx1","M_{#pi_{1}#pi_{
```

```
3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vtx1"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vtx1","M_{#pi_{2}#pi_{
4}} OS",massbins,0,5.);
  //
  //**histosTH1F["hm4rec2OS_pi1pi2vtx2"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vtx2","M_{#pi_{1}#pi_{
2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vtx2"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vtx2","M_{#pi_{3}#pi_{
4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vtx2"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vtx2","M_{#pi_{1}#pi_{
3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vtx2"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vtx2","M_{#pi_{2}#pi_{
4}} OS",massbins,0,5.);
  //

   //...A
   //**histosTH1F["hm4rec2OS_pi1pi2vee11"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee11","M_{#pi_{1}#p
i_{2}} OS",massbins,0,5.);
   //**histosTH1F["hm4rec2OS_pi3k4vee11"] = fs->make<TH1F>("hm4rec2OS_pi3k4vee11","M_{#pi_{3}K_{4
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_pi1pi3vee11"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee11","M_{#pi_{1}#p
i_{3}} OS",massbins,0,5.);
   //**histosTH1F["hm4rec2OS_pi2k4vee11"] = fs->make<TH1F>("hm4rec2OS_pi2k4vee11","M_{#pi_{2}K_{4
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_pi2pi3vee11"] = fs->make<TH1F>("hm4rec2OS_pi2pi3vee11","M_{#pi_{2}#p
i_{3}} OS",massbins,0,5.);
   //**histosTH1F["hm4rec2OS_pi1k4vee11"] = fs->make<TH1F>("hm4rec2OS_pi1k4vee11","M_{#pi_{1}K_{4
}} OS",massbins,0,5.);
   //...B
   //**histosTH1F["hm4rec2OS_k3pi4vee11"] = fs->make<TH1F>("hm4rec2OS_k3pi4vee11","M_{K_{3}#pi_{4
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_pi1pi4vee11"] = fs->make<TH1F>("hm4rec2OS_pi1pi4vee11","M_{#pi_{1}#p
i_{4}} OS",massbins,0,5.);
   //**histosTH1F["hm4rec2OS_k3pi2vee11"] = fs->make<TH1F>("hm4rec2OS_k3pi2vee11","M_{K_{3}#pi_{2
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_pi2pi4vee11"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee11","M_{#pi_{2}#p
i_{4}} OS",massbins,0,5.);
   //**histosTH1F["hm4rec2OS_k3pi1vee11"] = fs->make<TH1F>("hm4rec2OS_k3pi1vee11","M_{K_{3}#pi_{1
}} OS",massbins,0,5.);
   //...C
   //**histosTH1F["hm4rec2OS_pi1k2vee11"] = fs->make<TH1F>("hm4rec2OS_pi1k2vee11","M_{#pi_{1}K_{2
}} OS",massbins,0,5.);
   //**histosTH1F["hm4rec2OS_pi3pi4vee11"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee11","M_{#pi_{3}#p
i_{4}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_pi3k2vee11"] = fs->make<TH1F>("hm4rec2OS_pi3k2vee11","M_{#pi_{3}K_{2
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_pi4k2vee11"] = fs->make<TH1F>("hm4rec2OS_pi4k2vee11","M_{#pi_{4}K_{2
}} OS",massbins,0,5.);
   //...D
   //**histosTH1F["hm4rec2OS_k1pi2vee11"] = fs->make<TH1F>("hm4rec2OS_k1pi2vee11","M_{K_{1}#pi_{2
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_k1pi3vee11"] = fs->make<TH1F>("hm4rec2OS_k1pi3vee11","M_{K_{1}#pi_{3
}} OS",massbins,0,5.);
   //
   //**histosTH1F["hm4rec2OS_k1pi4vee11"] = fs->make<TH1F>("hm4rec2OS_k1pi4vee11","M_{K_{1}#pi_{4
}} OS",massbins,0,5.);
   //

   //...A...no PID  k4
   histosTH1F["hm4rec2OS_pi1pi2k4veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1pi2k4veeno11","M_{#pi_{
1}#pi_{2}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi3k4veeno11"] = fs->make<TH1F>("hm4rec2OS_pi3k4veeno11","M_{#pi_{3}K_{4
}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_pi1pi3k4veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1pi3k4veeno11","M_{#pi_{
1}#pi_{3}} OS",massbins,0,5.);
```

```
   histosTH1F["hm4rec2OS_pi2k4veeno11"] = fs->make<TH1F>("hm4rec2OS_pi2k4veeno11","M_{#pi_{2}K_{4
}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_pi2pi3k4veeno11"] = fs->make<TH1F>("hm4rec2OS_pi2pi3k4veeno11","M_{#pi_{
2}#pi_{3}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi1k4veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1k4veeno11","M_{#pi_{1}K_{4
}} OS",massbins,0,5.);
   //
   //...B...no PID  k3
   histosTH1F["hm4rec2OS_k3pi4veeno11"] = fs->make<TH1F>("hm4rec2OS_k3pi4veeno11","M_{K_{3}#pi_{4
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi1pi2k3veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1pi2k3veeno11","M_{#pi_{
1}#pi_{2}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_pi1pi4k3veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1pi4k3veeno11","M_{#pi_{
1}#pi_{4}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_k3pi2veeno11"] = fs->make<TH1F>("hm4rec2OS_k3pi2veeno11","M_{K_{3}#pi_{2
}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_pi2pi4k3veeno11"] = fs->make<TH1F>("hm4rec2OS_pi2pi4k3veeno11","M_{#pi_{
2}#pi_{4}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_k3pi1veeno11"] = fs->make<TH1F>("hm4rec2OS_k3pi1veeno11","M_{K_{3}#pi_{1
}} OS",massbins,0,5.);
   //
   //...C...no PID  k2
   histosTH1F["hm4rec2OS_pi1k2veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1k2veeno11","M_{#pi_{1}K_{2
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi3pi4k2veeno11"] = fs->make<TH1F>("hm4rec2OS_pi3pi4k2veeno11","M_{#pi_{
3}#pi_{4}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_pi3k2veeno11"] = fs->make<TH1F>("hm4rec2OS_pi3k2veeno11","M_{#pi_{3}K_{2
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi1pi4k2veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1pi4k2veeno11","M_{#pi_{
1}#pi_{4}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_pi4k2veeno11"] = fs->make<TH1F>("hm4rec2OS_pi4k2veeno11","M_{#pi_{4}K_{2
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi1pi3k2veeno11"] = fs->make<TH1F>("hm4rec2OS_pi1pi3k2veeno11","M_{#pi_{
1}#pi_{3}} OS",massbins,0,5.);
   //
   //...D...no PID  k1
   histosTH1F["hm4rec2OS_k1pi2veeno11"] = fs->make<TH1F>("hm4rec2OS_k1pi2veeno11","M_{K_{1}#pi_{2
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi3pi4k1veeno11"] = fs->make<TH1F>("hm4rec2OS_pi3pi4k1veeno11","M_{#pi_{
3}#pi_{4}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_k1pi3veeno11"] = fs->make<TH1F>("hm4rec2OS_k1pi3veeno11","M_{K_{1}#pi_{3
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi2pi4k1veeno11"] = fs->make<TH1F>("hm4rec2OS_pi2pi4k1veeno11","M_{#pi_{
2}#pi_{4}} OS",massbins,0,5.);
   //
   histosTH1F["hm4rec2OS_k1pi4veeno11"] = fs->make<TH1F>("hm4rec2OS_k1pi4veeno11","M_{K_{1}#pi_{4
}} OS",massbins,0,5.);
   histosTH1F["hm4rec2OS_pi2pi3k1veeno11"] = fs->make<TH1F>("hm4rec2OS_pi2pi3k1veeno11","M_{#pi_{
2}#pi_{3}} OS",massbins,0,5.);

  //...vee02
  //**histosTH1F["hm4rec2OS_pi1pi2vee02"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee02","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vee02"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee02","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vee02"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee02","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi2pi4vee02"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee02","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
   //...vee01
  //**histosTH1F["hm4rec2OS_pi1pi2vee01"] = fs->make<TH1F>("hm4rec2OS_pi1pi2vee01","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi3pi4vee01"] = fs->make<TH1F>("hm4rec2OS_pi3pi4vee01","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_pi1pi3vee01"] = fs->make<TH1F>("hm4rec2OS_pi1pi3vee01","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
```

```
  //**histosTH1F["hm4rec2OS_pi2pi4vee01"] = fs->make<TH1F>("hm4rec2OS_pi2pi4vee01","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);

  //...2dim vee11

  //...2dim vee02
  //***histosTH2F["hm4dim2OS_pi1pi2_pi3pi4vee02"] = fs->make<TH2F>("hm4dim2OS_pi1pi2_pi3pi4vee02"
,"M_{#pi_{1}#pi_{2}} vs M_{#pi_{3}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["hm4dim2OS_pi1pi3_pi2pi4vee02"] = fs->make<TH2F>("hm4dim2OS_pi1pi3_pi2pi4vee02"
,"M_{#pi_{1}#pi_{3}} vs M_{#pi_{2}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //...2dim vee01
  //***histosTH2F["hm4dim2OS_pi1pi2_pi3pi4vee01"] = fs->make<TH2F>("hm4dim2OS_pi1pi2_pi3pi4vee01"
,"M_{#pi_{1}#pi_{2}} vs M_{#pi_{3}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //***histosTH2F["hm4dim2OS_pi1pi3_pi2pi4vee01"] = fs->make<TH2F>("hm4dim2OS_pi1pi3_pi2pi4vee01"
,"M_{#pi_{1}#pi_{3}} vs M_{#pi_{2}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);

  //...veeno02
  histosTH1F["hm4rec2OS_pi1pi2veeno02"] = fs->make<TH1F>("hm4rec2OS_pi1pi2veeno02","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi3pi4veeno02"] = fs->make<TH1F>("hm4rec2OS_pi3pi4veeno02","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi1pi3veeno02"] = fs->make<TH1F>("hm4rec2OS_pi1pi3veeno02","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi2pi4veeno02"] = fs->make<TH1F>("hm4rec2OS_pi2pi4veeno02","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);
  //...veeno01
  histosTH1F["hm4rec2OS_pi1pi2veeno01"] = fs->make<TH1F>("hm4rec2OS_pi1pi2veeno01","M_{#pi_{1}#pi
_{2}} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi3pi4veeno01"] = fs->make<TH1F>("hm4rec2OS_pi3pi4veeno01","M_{#pi_{3}#pi
_{4}} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi1pi3veeno01"] = fs->make<TH1F>("hm4rec2OS_pi1pi3veeno01","M_{#pi_{1}#pi
_{3}} OS",massbins,0,5.);
  histosTH1F["hm4rec2OS_pi2pi4veeno01"] = fs->make<TH1F>("hm4rec2OS_pi2pi4veeno01","M_{#pi_{2}#pi
_{4}} OS",massbins,0,5.);

  //...2dim veeno11

  //...2dim veeno02
  histosTH2F["hm4dim2OS_pi1pi2_pi3pi4veeno02"] = fs->make<TH2F>("hm4dim2OS_pi1pi2_pi3pi4veeno02",
"M_{#pi_{1}#pi_{2}} vs M_{#pi_{3}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  histosTH2F["hm4dim2OS_pi1pi3_pi2pi4veeno02"] = fs->make<TH2F>("hm4dim2OS_pi1pi3_pi2pi4veeno02",
"M_{#pi_{1}#pi_{3}} vs M_{#pi_{2}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  //...2dim veeno01
  histosTH2F["hm4dim2OS_pi1pi2_pi3pi4veeno01"] = fs->make<TH2F>("hm4dim2OS_pi1pi2_pi3pi4veeno01",
"M_{#pi_{1}#pi_{2}} vs M_{#pi_{3}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);
  histosTH2F["hm4dim2OS_pi1pi3_pi2pi4veeno01"] = fs->make<TH2F>("hm4dim2OS_pi1pi3_pi2pi4veeno01",
"M_{#pi_{1}#pi_{3}} vs M_{#pi_{2}#pi_{4}} OS",massbins,0,5.,massbins,0,5.);

  //...........
  //
  //...Kaons
  //**histosTH1F["hm4rec2OS_k1k2"] = fs->make<TH1F>("hm4rec2OS_k1k2","M_{k_{1}k_{2}} OS",2*massbi
ns,0,5.);
  //**histosTH1F["hm4rec2OS_k3k4"] = fs->make<TH1F>("hm4rec2OS_k3k4","M_{k_{3}k_{4}} OS",2*massbi
ns,0,5.);
  //**histosTH1F["hm4rec2OS_k1k3"] = fs->make<TH1F>("hm4rec2OS_k1k3","M_{k_{1}k_{3}} OS",2*massbi
ns,0,5.);
  //**histosTH1F["hm4rec2OS_k2k4"] = fs->make<TH1F>("hm4rec2OS_k2k4","M_{k_{2}k_{4}} OS",2*massbi
ns,0,5.);
  //...v2
  //**histosTH1F["hm4rec2OS_k1k2v2"] = fs->make<TH1F>("hm4rec2OS_k1k2v2","M_{k_{1}k_{2}} OS",2*ma
ssbins,0,5.);
  //**histosTH1F["hm4rec2OS_k3k4v2"] = fs->make<TH1F>("hm4rec2OS_k3k4v2","M_{k_{3}k_{4}} OS",2*ma
ssbins,0,5.);
  //**histosTH1F["hm4rec2OS_k1k3v2"] = fs->make<TH1F>("hm4rec2OS_k1k3v2","M_{k_{1}k_{3}} OS",2*ma
ssbins,0,5.);
  //**histosTH1F["hm4rec2OS_k2k4v2"] = fs->make<TH1F>("hm4rec2OS_k2k4v2","M_{k_{2}k_{4}} OS",2*ma
ssbins,0,5.);
  //...2dim
  //***histosTH2F["hm4dim2OS_k1k2_k3k4"] = fs->make<TH2F>("hm4dim2OS_k1k2_k3k4","M_{k_{1}k_{2}} v
s M_{k_{3}k_{4}} OS",2*massbins,0,5.,2*massbins,0,5.);
  //***histosTH2F["hm4dim2OS_k1k3_k2k4"] = fs->make<TH2F>("hm4dim2OS_k1k3_k2k4","M_{k_{1}k_{3}} v
```

```
s M_{k_{2}k_{4}} OS",2*massbins,0,5.,2*massbins,0,5.);
  //...v2
  //***histosTH2F["hm4dim2OS_k1k2_k3k4v2"] = fs->make<TH2F>("hm4dim2OS_k1k2_k3k4v2","M_{k_{1}k_{2
}} vs M_{k_{3}k_{4}} OS",2*massbins,0,5.,2*massbins,0,5.);
  //***histosTH2F["hm4dim2OS_k1k3_k2k4v2"] = fs->make<TH2F>("hm4dim2OS_k1k3_k2k4v2","M_{k_{1}k_{3
}} vs M_{k_{2}k_{4}} OS",2*massbins,0,5.,2*massbins,0,5.);
  //...Kaons
  //
  //**histosTH1F["hm4rec2SS"] = fs->make<TH1F>("hm4rec2SS","M_{4#pi} SS",massbins,0,5.);
  //...2OSdiag
  //**histosTH1F["hm4rec2OS_diag"] = fs->make<TH1F>("hm4rec2OS_diag","M_{4#pi} TB/BT OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OS_diag2"] = fs->make<TH1F>("hm4rec2OS_diag2","M_{4#pi} TB/BT OS",1.60*m
assbins,0.0,8.0);
  //**histosTH1F["hm4rec2OS_diag3"] = fs->make<TH1F>("hm4rec2OS_diag3","M_{4#pi} TB/BT OS",0.50*m
assbins,0.0,2.5);
  //**histosTH1F["hm4rec2OS_diag4"] = fs->make<TH1F>("hm4rec2OS_diag4","M_{4#pi} TB/BT OS",0.24*m
assbins,2.5,4.0);
  //**histosTH1F["hm4rec2OS_diag5"] = fs->make<TH1F>("hm4rec2OS_diag5","M_{4#pi} TB/BT OS",0.32*m
assbins,4.0,8.0);

 //0-2.5 (125bins), 2.5-4(60bins), 4-8(80bins)
 double xmin1 = 0.;
 double xmax1 = 2.5;
 const int nbins1 = 125;

 double xmin2 = 2.5;
 double xmax2 = 4.;
 const int nbins2 = 60;

 double xmin3 = 4.;
 double xmax3 = 8.;
 const int nbins3 = 80;

 //**double bwidth1 = (xmax1 - xmin1)/nbins1;
 //**double bwidth2 = (xmax2 - xmin2)/nbins2;
 //**double bwidth3 = (xmax3 - xmin3)/nbins3;

 const int nbinstot = nbins1 + nbins2 + nbins3;
 //...Luiz
 //**double edges[nbinstot+1] ;

 //nbinstot++;

 int nbins=0;

 //**for( int i=0; i<nbins1; i++){ edges[nbins] = xmin1 + bwidth1 * i; nbins++;}
 //**for( int i=0; i<nbins2; i++){ edges[nbins] = xmin2 + bwidth2 * i; nbins++;}
 //...Luiz
 //**for( int i=0; i<=nbins3; i++){ edges[nbins] = xmin3 + bwidth3 * i; nbins++;}

 //**histosTH1F["hm4rec2OS_ttbb2varbin"] = fs->make<TH1F>("hm4rec2OS_ttbb2varbin","TTBB variable
bins",nbinstot,edges);
 //**histosTH1F["hm4rec2OS_diag2varbin"] = fs->make<TH1F>("hm4rec2OS_diag2varbin","DIAG variable
bins",nbinstot,edges);

  //...Pions
  //**histosTH1F["hm4rec2OS_diag_pi1pi2"] = fs->make<TH1F>("hm4rec2OS_diag_pi1pi2","M_{#pi_{1}#pi
_{2}} OS",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_diag_pi3pi4"] = fs->make<TH1F>("hm4rec2OS_diag_pi3pi4","M_{#pi_{3}#pi
_{4}} OS",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_diag_pi1pi3"] = fs->make<TH1F>("hm4rec2OS_diag_pi1pi3","M_{#pi_{1}#pi
_{3}} OS",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_diag_pi2pi4"] = fs->make<TH1F>("hm4rec2OS_diag_pi2pi4","M_{#pi_{2}#pi
_{4}} OS",2.0*massbins,0,10.);
  //...Kaons
  //**histosTH1F["hm4rec2OS_diag_k1k2"] = fs->make<TH1F>("hm4rec2OS_diag_k1k2","M_{k_{1}k_{2}} OS
",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_diag_k3k4"] = fs->make<TH1F>("hm4rec2OS_diag_k3k4","M_{k_{3}k_{4}} OS
",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_diag_k1k3"] = fs->make<TH1F>("hm4rec2OS_diag_k1k3","M_{k_{1}k_{3}} OS
",2.0*massbins,0,10.);
```

```
  //**histosTH1F["hm4rec2OS_diag_k2k4"] = fs->make<TH1F>("hm4rec2OS_diag_k2k4","M_{k_{2}k_{4}} OS
",2.0*massbins,0,10.);

  //...2SSdiag
  //**histosTH1F["hm4rec2SS_diag"] = fs->make<TH1F>("hm4rec2SS_diag","M_{4#pi} TB/BT SS",massbins
,0,5.);
  //
  //...2OSttbb
  //**histosTH1F["hm4rec2OS_ttbb"] = fs->make<TH1F>("hm4rec2OS_ttbb","M_{4#pi} TT/BB OS",massbins
,0,5.);
  //**histosTH1F["hm4rec2OS_ttbb2"] = fs->make<TH1F>("hm4rec2OS_ttbb2","M_{4#pi} TT/BB OS",1.60*m
assbins,0.0,8.0);
  //**histosTH1F["hm4rec2OS_ttbb3"] = fs->make<TH1F>("hm4rec2OS_ttbb3","M_{4#pi} TT/BB OS",0.50*m
assbins,0.0,2.5);
  //**histosTH1F["hm4rec2OS_ttbb4"] = fs->make<TH1F>("hm4rec2OS_ttbb4","M_{4#pi} TT/BB OS",0.24*m
assbins,2.5,4.0);
  //**histosTH1F["hm4rec2OS_ttbb5"] = fs->make<TH1F>("hm4rec2OS_ttbb5","M_{4#pi} TT/BB OS",0.32*m
assbins,4.0,8.0);

  //...Pions
  //**histosTH1F["hm4rec2OS_ttbb_pi1pi2"] = fs->make<TH1F>("hm4rec2OS_ttbb_pi1pi2","M_{#pi_{1}#pi
_{2}} OS",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_ttbb_pi3pi4"] = fs->make<TH1F>("hm4rec2OS_ttbb_pi3pi4","M_{#pi_{3}#pi
_{4}} OS",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_ttbb_pi1pi3"] = fs->make<TH1F>("hm4rec2OS_ttbb_pi1pi3","M_{#pi_{1}#pi
_{3}} OS",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_ttbb_pi2pi4"] = fs->make<TH1F>("hm4rec2OS_ttbb_pi2pi4","M_{#pi_{2}#pi
_{4}} OS",2.0*massbins,0,10.);
  //...Kaons
  //**histosTH1F["hm4rec2OS_ttbb_k1k2"] = fs->make<TH1F>("hm4rec2OS_ttbb_k1k2","M_{k_{1}k_{2}} OS
",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_ttbb_k3k4"] = fs->make<TH1F>("hm4rec2OS_ttbb_k3k4","M_{k_{3}k_{4}} OS
",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_ttbb_k1k3"] = fs->make<TH1F>("hm4rec2OS_ttbb_k1k3","M_{k_{1}k_{3}} OS
",2.0*massbins,0,10.);
  //**histosTH1F["hm4rec2OS_ttbb_k2k4"] = fs->make<TH1F>("hm4rec2OS_ttbb_k2k4","M_{k_{2}k_{4}} OS
",2.0*massbins,0,10.);
  //...2SSttbb
  //**histosTH1F["hm4rec2SS_ttbb"] = fs->make<TH1F>("hm4rec2SS_ttbb","M_{4#pi} TT/BB SS",massbins
,0,5.);
  //

  //...Luiz
  //**histosTH1F["hm4rec2OS_diag_trkP"] = fs->make<TH1F>("hm4rec2OS_diag_trkP","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_diag_trkM"] = fs->make<TH1F>("hm4rec2OS_diag_trkM","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_ttbb_trkP"] = fs->make<TH1F>("hm4rec2OS_ttbb_trkP","M_{4#pi} TT/BB OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_ttbb_trkM"] = fs->make<TH1F>("hm4rec2OS_ttbb_trkM","M_{4#pi} TT/BB OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);

  //**histosTH1F["hm4rec2OS_diag_pypxP"] = fs->make<TH1F>("hm4rec2OS_diag_pypxP","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_diag_pypxM"] = fs->make<TH1F>("hm4rec2OS_diag_pypxM","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} < 1",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_ttbb_pypxP"] = fs->make<TH1F>("hm4rec2OS_ttbb_pypxP","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec2OS_ttbb_pypxM"] = fs->make<TH1F>("hm4rec2OS_ttbb_pypxM","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} < 1",massbins,0,5.);

  //**histosTH1F["hm4rec3OS"] = fs->make<TH1F>("hm4rec3OS","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec3SS"] = fs->make<TH1F>("hm4rec3SS","M_{4#pi} SS",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_diag"] = fs->make<TH1F>("hm4rec3OS_diag","M_{4#pi} TB/BT OS",massbins
,0,5.);
  //**histosTH1F["hm4rec3SS_diag"] = fs->make<TH1F>("hm4rec3SS_diag","M_{4#pi} TB/BT SS",massbins
,0,5.);
  //**histosTH1F["hm4rec3OS_ttbb"] = fs->make<TH1F>("hm4rec3OS_ttbb","M_{4#pi} TT/BB OS",massbins
,0,5.);
  //**histosTH1F["hm4rec3SS_ttbb"] = fs->make<TH1F>("hm4rec3SS_ttbb","M_{4#pi} TT/BB SS",massbins
,0,5.);
```

```
  //...Luiz
  //**histosTH1F["hm4rec3OS_diag_trkP"] = fs->make<TH1F>("hm4rec3OS_diag_trkP","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_diag_trkM"] = fs->make<TH1F>("hm4rec3OS_diag_trkM","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_ttbb_trkP"] = fs->make<TH1F>("hm4rec3OS_ttbb_trkP","M_{4#pi} TT/BB OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_ttbb_trkM"] = fs->make<TH1F>("hm4rec3OS_ttbb_trkM","M_{4#pi} TT/BB OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);

  //**histosTH1F["hm4rec3OS_diag_pypxP"] = fs->make<TH1F>("hm4rec3OS_diag_pypxP","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_diag_pypxM"] = fs->make<TH1F>("hm4rec3OS_diag_pypxM","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} < 1",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_ttbb_pypxP"] = fs->make<TH1F>("hm4rec3OS_ttbb_pypxP","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec3OS_ttbb_pypxM"] = fs->make<TH1F>("hm4rec3OS_ttbb_pypxM","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} < 1",massbins,0,5.);

  //**histosTH1F["hm4rec4OS"] = fs->make<TH1F>("hm4rec4OS","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec4SS"] = fs->make<TH1F>("hm4rec4SS","M_{4#pi} SS",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_diag"] = fs->make<TH1F>("hm4rec4OS_diag","M_{4#pi} TB/BT OS",massbins
,0,5.);
  //**histosTH1F["hm4rec4SS_diag"] = fs->make<TH1F>("hm4rec4SS_diag","M_{4#pi} TB/BT SS",massbins
,0,5.);
  //**histosTH1F["hm4rec4OS_ttbb"] = fs->make<TH1F>("hm4rec4OS_ttbb","M_{4#pi} TT/BB OS",massbins
,0,5.);
  //**histosTH1F["hm4rec4SS_ttbb"] = fs->make<TH1F>("hm4rec4SS_ttbb","M_{4#pi} TT/BB SS",massbins
,0,5.);

  //...Luiz
  //**histosTH1F["hm4rec4OS_diag_trkP"] = fs->make<TH1F>("hm4rec4OS_diag_trkP","M_{4#pi}TB/BT OS,
 py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_diag_trkM"] = fs->make<TH1F>("hm4rec4OS_diag_trkM","M_{4#pi}TB/BT OS,
 py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_ttbb_trkP"] = fs->make<TH1F>("hm4rec4OS_ttbb_trkP","M_{4#pi}TT/BB OS,
 py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_ttbb_trkM"] = fs->make<TH1F>("hm4rec4OS_ttbb_trkM","M_{4#pi}TT/BB OS,
 py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);

  //**histosTH1F["hm4rec4OS_diag_pypxP"] = fs->make<TH1F>("hm4rec4OS_diag_pypxP","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_diag_pypxM"] = fs->make<TH1F>("hm4rec4OS_diag_pypxM","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} < 1",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_ttbb_pypxP"] = fs->make<TH1F>("hm4rec4OS_ttbb_pypxP","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec4OS_ttbb_pypxM"] = fs->make<TH1F>("hm4rec4OS_ttbb_pypxM","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} < 1",massbins,0,5.);

  //**histosTH1F["hm4rec5OS"] = fs->make<TH1F>("hm4rec5OS","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec5SS"] = fs->make<TH1F>("hm4rec5SS","M_{4#pi} SS",massbins,0,5.);
  //**histosTH1F["hm4rec5OS_diag"] = fs->make<TH1F>("hm4rec5OS_diag","M_{4#pi} TB/BT OS",massbins
,0,5.);
  //**histosTH1F["hm4rec5SS_diag"] = fs->make<TH1F>("hm4rec5SS_diag","M_{4#pi} TB/BT SS",massbins
,0,5.);
  //**histosTH1F["hm4rec5OS_ttbb"] = fs->make<TH1F>("hm4rec5OS_ttbb","M_{4#pi} TT/BB OS",massbins
,0,5.);
  //**histosTH1F["hm4rec5SS_ttbb"] = fs->make<TH1F>("hm4rec5SS_ttbb","M_{4#pi} TT/BB SS",massbins
,0,5.);

  //...Luiz
  //**histosTH1F["hm4rec5OS_diag_trkP"] = fs->make<TH1F>("hm4rec5OS_diag_trkP","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec5OS_diag_trkM"] = fs->make<TH1F>("hm4rec5OS_diag_trkM","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);
  //**histosTH1F["hm4rec5OS_ttbb_trkP"] = fs->make<TH1F>("hm4rec5OS_ttbb_trkP","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec5OS_ttbb_trkM"] = fs->make<TH1F>("hm4rec5OS_ttbb_trkM","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);

  //**histosTH1F["hm4rec5OS_diag_pypxP"] = fs->make<TH1F>("hm4rec5OS_diag_pypxP","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
```

```
  //**histosTH1F["hm4rec5OS_diag_pypxM"] = fs->make<TH1F>("hm4rec5OS_diag_pypxM","M_{4#pi} TB/BT
OS, |py/px|_{4#pi} < 1",massbins,0,5.);
  //**histosTH1F["hm4rec5OS_ttbb_pypxP"] = fs->make<TH1F>("hm4rec5OS_ttbb_pypxP","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} > 1",massbins,0,5.);
  //**histosTH1F["hm4rec5OS_ttbb_pypxM"] = fs->make<TH1F>("hm4rec5OS_ttbb_pypxM","M_{4#pi} TT/BB
OS, |py/px|_{4#pi} < 1",massbins,0,5.);

  //**histosTH1F["hm4rec6OS"] = fs->make<TH1F>("hm4rec6OS","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec6SS"] = fs->make<TH1F>("hm4rec6SS","M_{4#pi} SS",massbins,0,5.);
  //**histosTH1F["hm4rec6OS_diag"] = fs->make<TH1F>("hm4rec6OS_diag","M_{4#pi} TB/BT OS",massbins
,0,5.);
  //**histosTH1F["hm4rec6SS_diag"] = fs->make<TH1F>("hm4rec6SS_diag","M_{4#pi} TB/BT SS",massbins
,0,5.);
  //**histosTH1F["hm4rec6OS_ttbb"] = fs->make<TH1F>("hm4rec6OS_ttbb","M_{4#pi} TT/BB OS",massbins
,0,5.);
  //**histosTH1F["hm4rec6SS_ttbb"] = fs->make<TH1F>("hm4rec6SS_ttbb","M_{4#pi} TT/BB SS",massbins
,0,5.);

  //...Luiz
  //**histosTH1F["hm4rec6OS_diag_trkP"] = fs->make<TH1F>("hm4rec6OS_diag_trkP","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec6OS_diag_trkM"] = fs->make<TH1F>("hm4rec6OS_diag_trkM","M_{4#pi} TB/BT OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);
  //**histosTH1F["hm4rec6OS_ttbb_trkP"] = fs->make<TH1F>("hm4rec6OS_ttbb_trkP","M_{4#pi} TT/BB OS
, py_{#pi_{1}}py_{#pi_{2}}>0",massbins,0,5.);
  //**histosTH1F["hm4rec6OS_ttbb_trkM"] = fs->make<TH1F>("hm4rec6OS_ttbb_trkM","M_{4#pi} TT/BB OS
, py_{#pi_{1}}py_{#pi_{2}}<0",massbins,0,5.);

  //**histosTH1F["hm4recHFvetoOS"] = fs->make<TH1F>("hm4recHFvetoOS","M_{4#pi} HFv OS",massbins,0
,5.);
  //**histosTH1F["hm4recHFvetoSS"] = fs->make<TH1F>("hm4recHFvetoSS","M_{4#pi} HFv SS",massbins,0
,5.);

  //**histosTH1F["hm4rec45OS"] = fs->make<TH1F>("hm4rec45OS","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec45SS"] = fs->make<TH1F>("hm4rec45SS","M_{4#pi} SS",massbins,0,5.);
  //**histosTH1F["hm4rec4515OS"] = fs->make<TH1F>("hm4rec4515OS","M_{4#pi} OS",massbins,0,5.);
  //**histosTH1F["hm4rec4515SS"] = fs->make<TH1F>("hm4rec4515SS","M_{4#pi} SS",massbins,0,5.);

  //**histosTH1F["hm4rec9919"] = fs->make<TH1F>("hm4rec9919","M_{4#pi} 9919",massbins,0,5.);
  //**histosTH1F["hm4rec9922"] = fs->make<TH1F>("hm4rec9922","M_{4#pi} 9919,9922",massbins,0,5.);
  //**histosTH1F["hm4rec9971"] = fs->make<TH1F>("hm4rec9971","M_{4#pi} 9971",massbins,0,5.);
  //**histosTH1F["hm4rec9978"] = fs->make<TH1F>("hm4rec9978","M_{4#pi} 9978",massbins,0,5.);

  //**histosTH1F["hnclusters"] = fs->make<TH1F>("hnclusters","nPixelClusters",500,0,500.);
  //**histosTH1F["hnclusters2"] = fs->make<TH1F>("hnclusters2","nStripClusters",500,0,500.);
  //**histosTH1F["hnclustersOSdiag"] = fs->make<TH1F>("hnclustersOSdiag","nPixelClusters",500,0,5
00.);
  //**histosTH1F["hnclusters2OSdiag"] = fs->make<TH1F>("hnclusters2OSdiag","nStripClusters",500,0
,500.);

  //histosTH1F["halgo"] = fs->make<TH1F>("halgo","Algo",15,0,15.);
  //histosTH1F["hnhits"] = fs->make<TH1F>("hnhits","nhits pix+strip",40,0,40.);
  //histosTH1F["hchi2"] = fs->make<TH1F>("hchi2","normalized #chi^{2}",1050,-50,1000.);

  //...Luiz
  //histosTH1F["hdz"] = fs->make<TH1F>("hdz","dz",2000,-100,100.);
  //histosTH1F["hd0"] = fs->make<TH1F>("hd0","d0",2000,-100,100.);

  //**histosTH1F["halgov"] = fs->make<TH1F>("halgov","Algo",15,0,15.);
  //**histosTH1F["hnhitsv"] = fs->make<TH1F>("hnhitsv","nhits pixel",40,0,40.);
  //**histosTH1F["hchi2v"] = fs->make<TH1F>("hchi2v","normalized #chi^{2} vtx-fitted",550,-50,500
.);

  //...Luiz
  //**histosTH1F["hdzv"] = fs->make<TH1F>("hdzv","dz vtx-fitted",1000,-100,100.);
  //**histosTH1F["hd0v"] = fs->make<TH1F>("hd0v","d0 vtx-fitted",2000,-20,20.);

  //**histosTH1F["hchi2fin"] = fs->make<TH1F>("hchi2fin","normalized #chi^{2} vtx-fitted",550,-50
,500.);

  //...Luiz
  //**histosTH1F["hdzfin"] = fs->make<TH1F>("hdzfin","dz vtx-fitted",1000,-100,100.);
```

```
  //**histosTH1F["hd0fin"] = fs->make<TH1F>("hd0fin","d0 vtx-fitted",2000,-20,20.);

  //...Luiz
  //**histosTH1F["hdeltaR"] = fs->make<TH1F>("hdeltaR","#DeltaR trk-trk",200,0,10.);
  //**histosTH1F["hdeltaR2"] = fs->make<TH1F>("hdeltaR2","#DeltaR trk-trk",200,0,10.);

  //-----------------
  histosTH2F["h2dimdpyAll"] = fs->make<TH2F>("h2dimdpyAll", "p_{y}^{TOTEM} vs p_{y}^{CMS}",200,-2
.,2.,200,-2., 2.);
  histosTH2F["h2dimdpy"] = fs->make<TH2F>("h2dimdpy","p_{y}^{TOTEM} vs p_{y}^{CMS}",200,-2.,2.,20
0,-2.,2.);
  //***histosTH2F["h2dimdpy_diag"] = fs->make<TH2F>("h2dimdpy_diag","p_{y}^{TOTEM} vs p_{y}^{CMS}
 diag",100,-2.,2.,100,-2.,2.);
  //***histosTH2F["h2dimdpy_ttbb"] = fs->make<TH2F>("h2dimdpy_ttbb","p_{y}^{TOTEM} vs p_{y}^{CMS}
 TT/BB",100,-2.,2.,100,-2.,2.);

  //...Luiz
  histosTH1F["hdpyAll"] = fs->make<TH1F>("hdpyAll"  ,"#Deltap_{Y} CMS-TOTEM",500,-0.5,0.5);
  histosTH1F["hdpy"] = fs->make<TH1F>("hdpy"     ,"#Deltap_{Y} CMS-TOTEM",500,-0.5,0.5);
  //**histosTH1F["hdpy_diag"] = fs->make<TH1F>("hdpy_diag","#Deltap_{Y} CMS-TOTEM TB/BT",500,-0.5
,0.5);
  //**histosTH1F["hdpy_ttbb"] = fs->make<TH1F>("hdpy_ttbb","#Deltap_{Y} CMS-TOTEM TT/BB",500,-0.5
,0.5);

  histosTH2F["h2dimdpxAll"] = fs->make<TH2F>("h2dimdpxAll", "p_{x}^{TOTEM} vs p_{x}^{CMS}",200,-2
.,2.,200,-2., 2.);
  histosTH2F["h2dimdpx"] = fs->make<TH2F>("h2dimdpx","p_{x}^{TOTEM} vs p_{x}^{CMS}",200,-2.,2.,20
0,-2.,2.);
  //***histosTH2F["h2dimdpx_diag"] = fs->make<TH2F>("h2dimdpx_diag","p_{x}^{TOTEM} vs p_{x}^{CMS}
 diag",100,-2.,2.,100,-2.,2.);
  //***histosTH2F["h2dimdpx_ttbb"] = fs->make<TH2F>("h2dimdpx_ttbb","p_{x}^{TOTEM} vs p_{x}^{CMS}
 TT/BB",100,-2.,2.,100,-2.,2.);

  //...Luiz
  histosTH1F["hdpxAll"] = fs->make<TH1F>("hdpxAll", "#Deltap_{X} CMS-TOTEM",500,-0.5,0.5);
  histosTH1F["hdpx"] = fs->make<TH1F>("hdpx", "#Deltap_{X} CMS-TOTEM",500,-0.5,0.5);
  //**histosTH1F["hdpx_diag"] = fs->make<TH1F>("hdpx_diag", "#Deltap_{X} CMS-TOTEM TB/BT",500,-0.
5,0.5);
  //**histosTH1F["hdpx_ttbb"] = fs->make<TH1F>("hdpx_ttbb", "#Deltap_{X} CMS-TOTEM TT/BB",500,-0.
5,0.5);

  //...checking!
  //**histosTH1F["hcmspx"] = fs->make<TH1F>("hcmspx","CMSpx",500,-0.5,0.5);
  //**histosTH1F["hcmspy"] = fs->make<TH1F>("hcmspy","CMSpy",500,-0.5,0.5);
  //**histosTH1F["hcmspxk"] = fs->make<TH1F>("hcmspxk","CMSpxK",500,-0.5,0.5);
  //**histosTH1F["hcmspyk"] = fs->make<TH1F>("hcmspyk","CMSpyK",500,-0.5,0.5);

  //-----------------
  //***histosTH2F["h2dimxVtxRL"] = fs->make<TH2F>("h2dimxVtxRL","xVtxL vs xVtxR (m)",1000,-0.004,
0.001,1000,-0.004,0.001);
  //***histosTH2F["h2dimxVtxcmsR"] = fs->make<TH2F>("h2dimxVtxcmsR","xVtxCMS vs xVtxR (cm)",300,-
0.3,0.3,400,-0.3,0.5);
  //***histosTH2F["h2dimxVtxcmsL"] = fs->make<TH2F>("h2dimxVtxcmsL","xVtxCMS vs xVtxL (cm)",300,-
0.3,0.3,400,-0.3,0.5);
  //***histosTH2F["h2dimxVtxcmsRL"] = fs->make<TH2F>("h2dimxVtxcmsRL","xVtxCMS vs xVtxRL (cm)",30
0,-0.3,0.3,400,-0.3,0.5);

  //***histosTH2F["h2dimxVtxcmsR2"] = fs->make<TH2F>("h2dimxVtxcmsR2","xVtxCMS vs xVtxR (cm) (|xV
txL-xVtxR|<3e-5)",300,-0.3,0.3,400,-0.3,0.5);
  //***histosTH2F["h2dimxVtxcmsL2"] = fs->make<TH2F>("h2dimxVtxcmsL2","xVtxCMS vs xVtxL (cm) (|xV
txL-xVtxR|<3e-5)",300,-0.3,0.3,400,-0.3,0.5);
  //***histosTH2F["h2dimxVtxcmsRL2"] = fs->make<TH2F>("h2dimxVtxcmsRL2","xVtxCMS vs xVtxRL (cm)",
300,-0.3,0.3,400,-0.3,0.5);

  //***histosTH2F["h2dimxVtx_zVtx_CT"] = fs->make<TH2F>("h2dimxVtx_zVtx_CT","xVtxCMS-xVtxTOTEM vs
 zVtx (cm)",300,-20.,20.,400,-0.3,0.5);
  //***histosTH2F["h2dimxVtx_zVtx_C"] = fs->make<TH2F>("h2dimxVtx_zVtx_C","xVtxCMS vs zVtx (cm)",
300,-20.,20.,400,-0.3,0.5);
  //***histosTH2F["h2dimxVtx_zVtx_T"] = fs->make<TH2F>("h2dimxVtx_zVtx_T","xVtxTOTEM vs zVtx (cm)
",300,-20.,20.,400,-0.3,0.5);

  //**histosTH1F["hxVtxRL"] = fs->make<TH1F>("hxVtxRL","xVtxR-xVtxL (m)",300,-0.0003,0.0003);
```

```
  //...Luiz
  //**histosTH1F["hxVtxcmsR"] = fs->make<TH1F>("hxVtxcmsR","xVtxCMS-xVtxR (cm)",500,-0.5,0.5);
  //**histosTH1F["hxVtxcmsL"] = fs->make<TH1F>("hxVtxcmsL","xVtxCMS-xVtxL (cm)",500,-0.5,0.5);
  //**histosTH1F["hxVtxcmsRL"] = fs->make<TH1F>("hxVtxcmsRL","xVtxCMS-xVtxTOTEM (cm)",500,-0.5,0.
5);

  //**histosTH1F["hxVtxRL_diag"] = fs->make<TH1F>("hxVtxRL_diag","xVtxR-xVtxL (m)",300,-0.0003,0.
0003);

  //...Luiz
  //**histosTH1F["hxVtxcmsR_diag"] = fs->make<TH1F>("hxVtxcmsR_diag","xVtxCMS-xVtxR (cm)",500,-0.
5,0.5);
  //**histosTH1F["hxVtxcmsL_diag"] = fs->make<TH1F>("hxVtxcmsL_diag","xVtxCMS-xVtxL (cm)",500,-0.
5,0.5);
  //**histosTH1F["hxVtxcmsRL_diag"] = fs->make<TH1F>("hxVtxcmsRL_diag","xVtxCMS-xVtxTOTEM (cm)",5
00,-0.5,0.5);

  //**histosTH1F["hxVtxRL_ttbb"] = fs->make<TH1F>("hxVtxRL_ttbb","xVtxR-xVtxL (m)",300,-0.0003,0.
0003);

  //...Luiz
  //**histosTH1F["hxVtxcmsR_ttbb"] = fs->make<TH1F>("hxVtxcmsR_ttbb","xVtxCMS-xVtxR (cm)",500,-0.
5,0.5);
  //**histosTH1F["hxVtxcmsL_ttbb"] = fs->make<TH1F>("hxVtxcmsL_ttbb","xVtxCMS-xVtxL (cm)",500,-0.
5,0.5);
  //**histosTH1F["hxVtxcmsRL_ttbb"] = fs->make<TH1F>("hxVtxcmsRL_ttbb","xVtxCMS-xVtxTOTEM (cm)",5
00,-0.5,0.5);

  //...Luiz

  //***histosTH2F["hdedx"] = fs->make<TH2F>("hdedx","dE/dx vs p", 1000, 0.,20.,1000, 0.,200.);
  //***histosTH2F["hdedxvee11"] = fs->make<TH2F>("hdedxvee11","dE/dx vs p type:11", 1000, 0.,20.,
1000, 0.,200.);
  //***histosTH2F["hdedxvee02"] = fs->make<TH2F>("hdedxvee02","dE/dx vs p type:02", 1000, 0.,20.,
1000, 0.,200.);
  //***histosTH2F["hdedxvee01"] = fs->make<TH2F>("hdedxvee01","dE/dx vs p type:01", 1000, 0.,20.,
1000, 0.,200.);
  //***histosTH2F["hdedxrejKp"] = fs->make<TH2F>("hdedxrejKp","dE/dx vs p rejecting K or p", 1000
, 0.,20.,1000, 0.,200.);
  //***histosTH2F["hdedx2rejKp"] = fs->make<TH2F>("hdedx2rejKp","dE/dx vs p nvtx=1 rejecting K or
 p", 1000, 0.,20.,1000, 0.,200.);
  //***histosTH2F["hdedxrejKpu"] = fs->make<TH2F>("hdedxrejKpu","dE/dx vs p rejecting K or p or u
nknown", 1000, 0.,20.,1000, 0.,200.);
  //***histosTH2F["hdedx2rejKpu"] = fs->make<TH2F>("hdedx2rejKpu","dE/dx vs p nvtx=1 rejecting K
or p or unknown", 1000, 0.,20.,1000, 0.,200.);

  //...Luiz
  //***histosTH2F["hlndedx"]  = fs->make<TH2F>("hlndedx","ln dE/dx vs p", 500, 0.,5.,1000, 0.,5.)
;
  //***histosTH2F["hl10dedx"] = fs->make<TH2F>("hl10dedx","log10 dE/dx vs p", 500, 0.,5.,1000, 0.
,5.);

  /*
  // eneMK0
  histosTH1F["henemk012"] = fs->make<TH1F>("henemk012","eneMK0 #pi1#pi2", 5000, 0.,50.);
  histosTH1F["henemk034"] = fs->make<TH1F>("henemk034","eneMK0 #pi3#pi4", 5000, 0.,50.);
  */

  //...proton momentum
  histosTH1F["hprotonpl"] = fs->make<TH1F>("hprotonpl","proton left p",1000,0.,20.);
  histosTH1F["hprotonpr"] = fs->make<TH1F>("hprotonpr","proton right p",1000,0.,20.);
  //
  histosTH1F["hprotonplx"] = fs->make<TH1F>("hprotonplx","proton left px",2000,-20.,20.);
  histosTH1F["hprotonply"] = fs->make<TH1F>("hprotonply","proton left py",2000,-20.,20.);
  histosTH1F["hprotonprx"] = fs->make<TH1F>("hprotonprx","proton right px",2000,-20.,20.);
  histosTH1F["hprotonpry"] = fs->make<TH1F>("hprotonpry","proton right py",2000,-20.,20.);

  //...4-momentum transfer squared
  histosTH1F["ht1"] = fs->make<TH1F>("ht1","|-t1|",1000,0.,5.);
  histosTH1F["ht2"] = fs->make<TH1F>("ht2","|-t2|",1000,0.,5.);
  histosTH1F["ht1t2"] = fs->make<TH1F>("ht1t2","|-(t1+t2)|",1000,0.,5.);
```

```
  histosTH1F["ht12"] = fs->make<TH1F>("ht12","|-(t1+t2)|",1000,0.,5.);

  // xi 200,-0.5,0.5
// <<<<<<<<<<

  std::cout<<"booked all of Luiz' histograms."<<std::endl;

  //---------------end of my histograms

}

// ------------ method called once each job just after ending the event loop  ------------
void
PromptAnalyzer::endJob()
{
  // this does not work ...Luiz
  // Output file
  // TFile* output = new TFile(outputFileName.c_str(),"RECREATE");
  ////TFile* output = new TFile("output.root","RECREATE");
  ////output->cd();

  //// don't include it with TFileService ...Write() and Close() are done automatically!
  ////for(map<string,TH1F*>::iterator it_histo = histosTH1F.begin();it_histo != histosTH1F.end();
 ++it_histo)(*it_histo).second->Write();
  ////for(map<string,TH2F*>::iterator it_histo = histosTH2F.begin();it_histo != histosTH2F.end();
 ++it_histo)(*it_histo).second->Write();

  //  for(map<string,TProfile*>::iterator it_histo = histosTProf.begin();it_histo != histosTProf.
end(); ++it_histo)(*it_histo).second->Write();
  //  for(map<string,TH3F*>::iterator it_histo = histosTH3F.begin();it_histo != histosTH3F.end();
 ++it_histo)(*it_histo).second->Write();

  ////output->Close();
  std::cout<<"ciao ciao..."<<std::endl;
}
//--------------------------------------------------
bool PromptAnalyzer::jsonLocal(int runnr, int ls){

  int accept = false;

  if(runnr == 319104 && ls >= 22 && ls <=176) accept = true;
  if(runnr == 319124){
    if(ls >= 151 && ls <= 186) accept = true;
    if(ls >= 192 && ls <= 277) accept = true; //changed from 149 276
  }
  if(runnr == 319125 && ls >= 25 && ls <= 191) accept = true;
  if(runnr == 319159 && ls >= 202 && ls <= 617) accept = true;
  if(runnr == 319174 && ls >= 24 && ls <= 70) accept = true;
  if(runnr == 319175 && ls >= 1 && ls <= 139) accept = true;
  if(runnr == 319176 && ls >= 1 && ls <= 1799) accept = true;
  if(runnr == 319177){
    if(ls >= 11 && ls <= 190) accept = true;
    if(ls >= 215 && ls <= 223) accept = true;
  }
  if(runnr == 319190){
    if(ls >= 39 && ls <= 125) accept = true;
    if(ls >= 147 && ls <= 309) accept = true;
  }

  if(runnr == 319222){
    if(ls >= 192 && ls <= 230) accept = true;
    if(ls >= 233 && ls <= 294) accept = true;
  }
  if(runnr == 319223 && ls >= 5 && ls <= 131) accept = true;
  if(runnr == 319254 && ls >= 199 && ls <= 262) accept = true;
  if(runnr == 319255 && ls >= 1 && ls <= 164) accept = true;
  if(runnr == 319256){
    if(ls >= 1 && ls <= 38) accept = true;
    if(ls >= 41 && ls <= 726) accept = true;
  }
  if(runnr == 319262){
    if(ls == 10) accept = true;
```

```
    if(ls >= 15 && ls <= 16) accept = true;
    if(ls >= 20 && ls <= 23) accept = true;
    if(ls >= 29 && ls <= 34) accept = true;
    if(ls >= 39 && ls <= 40) accept = true;
    if(ls >= 46 && ls <= 58) accept = true;
    if(ls >= 61 && ls <= 78) accept = true;
    if(ls >= 82 && ls <= 88) accept = true;
    if(ls >= 90 && ls <= 123) accept = true;
    if(ls >= 129 && ls <= 358) accept = true;
  }
  if(runnr == 319263 && ls >= 1 && ls <= 364) accept = true;
  if(runnr == 319264 && ls >= 1 && ls <= 57) accept = true;
  if(runnr == 319265 && ls >= 1 && ls <= 396) accept = true;
  if(runnr == 319266){
    if(ls >= 1 && ls <= 18) accept = true;
    if(ls >= 20 && ls <= 26) accept = true;
  }
  if(runnr == 319267 && ls >= 1 && ls <= 204) accept = true;
  if(runnr == 319268){
    if(ls >= 1 && ls <= 185) accept = true;
    if(ls >= 187 && ls <= 462) accept = true;
    }
  if(runnr == 319270 && ls >= 1 && ls <= 205) accept = true;

  if(runnr == 319300){
    if(ls >= 57 && ls <= 194) accept = true;
    if(ls >= 203 && ls <= 604) accept = true;
    if(ls >= 606 && ls <= 871) accept = true;
    if(ls >= 874 && ls <= 987) accept = true;
    if(ls >= 990 && ls <= 1127) accept = true;
  }
  if(runnr == 319311){
    if(ls >= 60 && ls <= 76) accept = true;
    if(ls >= 78 && ls <= 275) accept = true;
    if(ls >= 282 && ls <= 300) accept = true;
    if(ls >= 302 && ls <= 526) accept = true;
    if(ls >= 530 && ls <= 829) accept = true;
    if(ls >= 839 && ls <= 1236) accept = true;
    if(ls >= 1238 && ls <= 1489) accept = true;
    if(ls >= 1491 && ls <= 1713) accept = true;
  }

    return accept;

}

// ------------ method called when starting to processes a run  ------------
void
PromptAnalyzer::beginRun(edm::Run const& run, edm::EventSetup const& es)
{

  bool changed(true);
  if (hltConfig_.init(run, es, "HLT",changed)) {
    hltConfig_.dump("Triggers");
    hltConfig_.dump("PrescaleTable");

  }

}

// ------------ method called when ending the processing of a run  ------------
void
PromptAnalyzer::endRun(edm::Run const&, edm::EventSetup const&)
{
}

// ------------ method fills 'descriptions' with the allowed parameters for the module  ---------
---
void
PromptAnalyzer::fillDescriptions(edm::ConfigurationDescriptions& descriptions) {
  //The following says we do not know what parameters are allowed so do no validation
  // Please change this to state exactly what you do use, even if it is no parameters
```

```
  edm::ParameterSetDescription desc;
  desc.setUnknown();
  descriptions.addDefault(desc);

  //Specify that only 'tracks' is allowed
  //To use, remove the default given above and uncomment below
  //ParameterSetDescription desc;
  //desc.addUntracked<edm::InputTag>("tracks","ctfWithMaterialTracks");
  //descriptions.addDefault(desc);
}

//define this as a plug-in
DEFINE_FWK_MODULE(PromptAnalyzer);
```