Bitácora

Pablo Jibrán Pomares Valdés

9 de diciembre de 2024

Filtro

Parámetros considerados.
Se busca que:

- Existe conservación de carga
- Se tengan 4 muones
- No haya jets con btag, rudeza media (DeepbTag $< 0{,}5847$). ref
- Todos los muones sean globales.

Código

```
void filter(){
  TFile *f_old = TFile::Open("root://eospublic.cern.ch//eos/opendata/cms/Run2016H/DoubleMuon/NANOAOD/UL2
  TTree* t_old;
  f_old->GetObject("Events", t_old);

  const int nentries = t_old->GetEntries();

  // Deactivate all branches
  t_old->SetBranchStatus("*", 0);

  // Activate desired branches
  for (auto actBranchName : {"run", "event", "Muon_charge", "Muon_dxy", "Muon_dxyErr", "Muon_isGlobal",
                             "Muon_isTracker", "Muon_pt", "Muon_phi", "Muon_eta", "nMuon", "MET_phi",
                             "MET_pt", "MET_significance", "nJet", "Jet_btagCSVV2", "Jet_btagDeepB"}){
    t_old->SetBranchStatus(actBranchName, 1);
  };

  // Entry selection
  UInt_t nMuon, nJet;
  Float_t Jet_btagDeepB[20];
  Bool_t Muon_isGlobal[10];
  Int_t Muon_charge[10];
  t_old->SetBranchAddress("nMuon", &nMuon);
  t_old->SetBranchAddress("nJet", &nJet);
  t_old->SetBranchAddress("Jet_btagDeepB", &Jet_btagDeepB);
  t_old->SetBranchAddress("Muon_isGlobal", &Muon_isGlobal);
  t_old->SetBranchAddress("Muon_charge", &Muon_charge);

  TFile newfile("prueba1.root", "recreate");
  auto t_new = t_old->CloneTree(0);
```

Código

```
for (int i=0; i<nentries; i++){
  t_old->GetEntry(i);

  int sumCharge = 0;
  for (int j=0; j<nMuon; j++){
    sumCharge += Muon_charge[j];
  };
  bool chargeViolation = sumCharge;

  bool passbTag = true;
  for (int j=0; j<nJet; j++){
    if (Jet_btagDeepB[j] > 0.5847){
      passbTag = false;
    };
  };

  bool passnMuon = false;
  if (nMuon == 4){
    passnMuon = true;
  };

  bool passAllGlobal = true;
  for (int j=0; j<nMuon; j++){
    if (Muon_isGlobal[j] == 0){
      passAllGlobal = false;
    };
  };
  if (passAllGlobal && passbTag && passnMuon && !chargeViolation) {
    t_new->Fill();
  };

};

t_new->Print();
newfile.Write();
}
```

Buscador de bosones Z

Parámetros considerados
Consideré las variables:

- Muon_pt
- Muon_charge
- Muon_eta
- Muon_phi

Masa invariante

Sabemos que la expresión para la masa invariante:

$$M^2 = 2p_{T1}p_{T2}(\cosh(\eta_1 - \eta_2) - \cos(\phi_1 - \phi_2))$$

```
Double_t inv_mass(Float_t pt1, Float_t pt2, Float_t phi1, Float_t phi2, Float_t eta1, Float_t eta2) {

  Double_t eta_diff = eta1 - eta2;
  Double_t phi_diff = phi1 - phi2;
  Double_t pt_prod = 2*pt1*pt2;

  Double_t m2 = pt_prod*(TMath::CosH(eta_diff) - TMath::Cos(phi_diff));
  Double_t m = TMath::Sqrt(m2);

  return m;
}
```

z_finder I

En general, se comparan todos los muones para buscar la generación de bosones Z. Por conservación de la carga, los muones que comparemos tiene que ser opuestos. Si esa condición se cumple, se busca que la masa invariante se encuentre en el rango $|m - m_Z| \leq 10$ GeV.
En caso de que se encuentren dos candidatos que compartan un leptón, se selecciona el que tenga un mayor ángulo entre los muones.
Por último, se regresa un tuple que contenga el número de bosones Z en el determinado evento y una masa.

z_finder II

```
// Find the number of Z bosons on a event and its mass.
// If >1 it returns (for now) a single mass. However, this is unimportant because event is discarted.
std::tuple<UInt_t, Double_t> z_finder(Float_t muon_pt[4], Float_t muon_phi[4], Float_t muon_eta[4], Int_t muon_charge[4]){
  UInt_t num_z = 0;
  Double_t masses[2];
  for (int i=0; i<4; i++){
    int local_z = 0; // if >1 check for the greatest eta diff
    int z_index[3];
    Double_t local_masses[2] = {0., 0.};

    for (int j=i+1; j<4; j++){
      Float_t pt1 = muon_pt[i];
      Float_t pt2 = muon_pt[j];
      Float_t phi1 = muon_phi[i];
      Float_t phi2 = muon_phi[j];
      Float_t eta1 = muon_eta[i];
      Float_t eta2 = muon_eta[j];
      Int_t q1 = muon_charge[i];
      Int_t q2 = muon_charge[j];

      bool same_charge = q1 + q2;
      z_index[0] = i;

      if (!same_charge) {
        Double_t m = inv_mass(pt1, pt2, phi1, phi2, eta1, eta2);
        if (m > 81.2 && m < 101.2){
          local_masses[local_z] = m;
          z_index[local_z+1] = j;
          local_z++;
          };
        }
    };
```

z_finder III

```
    if (local_z == 2){
      float phi1 = TMath::Abs(muon_phi[z_index[0]]);
      float phi2 = TMath::Abs(muon_phi[z_index[1]]);
      float phi3 = TMath::Abs(muon_phi[z_index[2]]);

      float diff1_2 = phi1 - phi2;
      float diff1_3 = phi1 - phi3;

      if (diff1_2 > diff1_3){
        masses[num_z] = local_masses[0];
      }
      else {
        masses[num_z] = local_masses[1];
      };
    };

    if (local_z) {num_z++;};
  }

  std::tuple<UInt_t, Double_t> result = {num_z, masses[0]};

  return result;
}
```

z_finder results

```
Processing wwz_finder.C...
Se tienen 110 eventos con 1 Z.
Se tienen 2 eventos con 2 Z.
Se tienen 4593 eventos con ningún Z.
```