Bugün

Numpy

TLorentzVector

Z bozonu

Hazirlayan

Ece ASILAR

Numpy

Neden ?

- python'da array'ler iyi guzel ama sinirli, 255 elemandan fazlasini alamiyor. ROOT bunu sevmiyor.
- bu boslugu doldurmak icin python'da numpy var.

```
In [2]: import numpy as n

In [3]: x = n.array([1,2,3,4,5])

In [4]: x.dtype
Out[4]: dtype('int32')

In [5]: x = n.array([1.,2.,3.,4.,5.])

In [6]: x.dtype
Out[6]: dtype('float64')

In [7]: x = n.array([1,2,3,4,5], dtype=float)

In [8]: x.dtype
Out[8]: dtype('float64')

In [9]: x
Out[9]: array([ 1.,  2.,  3.,  4.,  5.])

In [10]: n.linspace(0,5,11)
Out[10]: array([ 0. ,  0.5,  1. ,  1.5,  2. ,  2.5,  3. ,  3.5,  4. ,  4.5,  5. ])

In [11]: n.arange(0,5,.25)
Out[11]:
array([ 0.  ,  0.25,  0.5 ,  0.75,  1.  ,  1.25,  1.5 ,  1.75,  2.  ,
        2.25,  2.5 ,  2.75,  3.  ,  3.25,  3.5 ,  3.75,  4.  ,  4.25,
        4.5 ,  4.75])

In [12]: n.zero
n.zeros       n.zeros_like

In [12]: n.zeros(3, dtype=float)
Out[12]: array([ 0.,  0.,  0.])
```

Numpy

Neden ?

- python'da array'ler iyi guzel ama sinirli, 255 elemandan fazlasini alamiyor. ROOT bunu sevmiyor.
- bu boslugu doldurmak icin python'da numpy var.

Numpy'yi Gelecek ders kullanicaz !!

```
In [19]: x = n.array([1,2,3,4,5,6,7,8], dtype=float)

In [20]: x.s
x.searchsorted  x.setflags      x.size          x.squeeze       x.strides       x.swapaxes
x.setfield      x.shape         x.sort          x.std           x.sum

In [20]: x.shape
Out[20]: (8,)

In [21]: y = x.reshape(4,2)

In [22]: y
Out[22]:
array([[ 1.,  2.],
       [ 3.,  4.],
       [ 5.,  6.],
       [ 7.,  8.]])
```

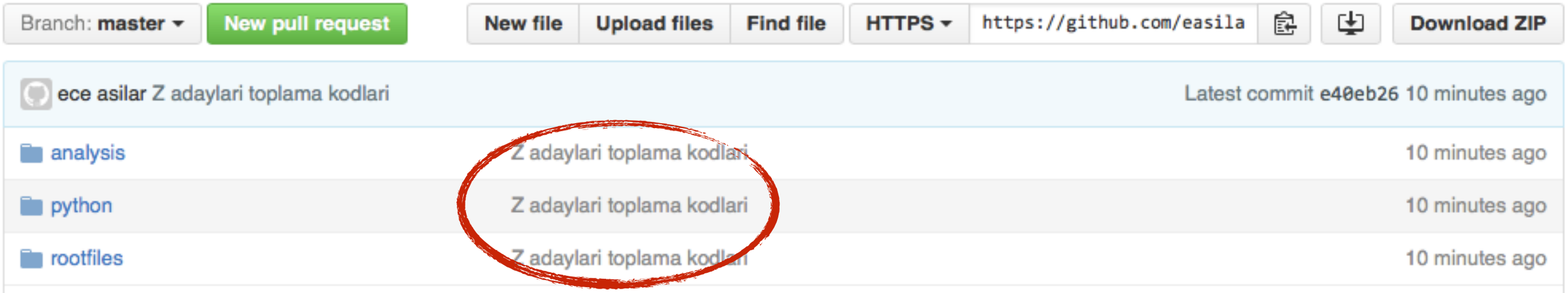
```
In [23]: y[0,:]   # everything in first row
Out[23]: array([ 1.,  2.])

In [25]: y[0:2,1] # first two elements in second column
Out[25]: array([ 2.,  4.])

In [28]: y[::2,1] # every second element in second column
Out[28]: array([ 2.,  6.])

In [29]: y[::2,1] = 0 # set every second element in second column to zero

In [30]: y
Out[30]:
array([[ 1.,  0.],
       [ 3.,  4.],
       [ 5.,  0.],
       [ 7.,  8.]])
```

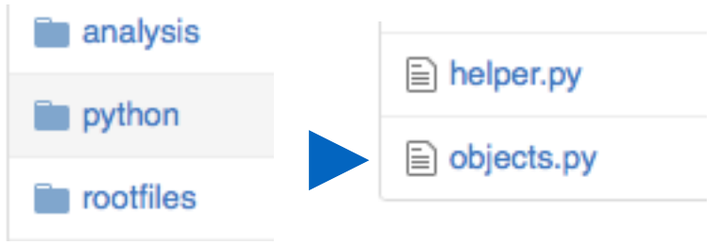
```
In [50]: x[(x > 2) & (x < 4)]
Out[50]: array([ 3.,  3.])

In [51]: x[(x > 2)]
Out[51]: array([ 3.,  4.,  3.])
```

```
In [35]: x = n.array([1,2,3,4,5,6,7,8], dtype=float)

In [36]: y = n.array([2,0,2,0,2,0,2,0], dtype=float)

In [37]: x*y
Out[37]: array([  2.,   0.,   6.,   0.,  10.,   0.,  14.,   0.])

In [38]: x + 2
Out[38]: array([  3.,   4.,   5.,   6.,   7.,   8.,   9.,  10.])

In [41]: y = n.cos( n.linspace(0, n.pi,11) )

In [42]: y
Out[42]:
array([  1.00000000e+00,   9.51056516e-01,   8.09016994e-01,
         5.87785252e-01,   3.09016994e-01,   6.12303177e-17,
        -3.09016994e-01,  -5.87785252e-01,  -8.09016994e-01,
        -9.51056516e-01,  -1.00000000e+00])
```

Sırada,

- Tree'nin icinden nesneleri nasil cekerim
- TLorentzvector'u nasil kullanirim
- ☑ Butun kod'lar burada: easilar / HUPP-codeHelper





=> analiz yaparken farkli amaclar icin tasarlanmis farkli kodlardan, ayni nesneleri cagirmam gerekebilir.

ornegin,

bir olayin icindeki electronlari iki ayri kod kullanmak isteyebilir;

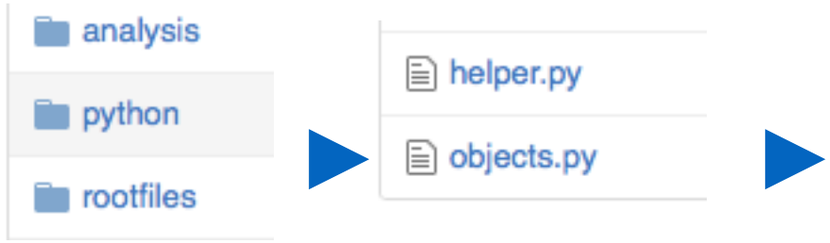
birinci kod verimlilik (Efficiency) hesaplamak icin,

ikinci kod bu electronlarin toplam degismez kutlelerini hesaplamak uzerine kullanabilir.

Bu durumda mantikli olan tum nesneleri ayri bir modul'de cagirmaktir.

suna benzer fonksiyonlarimiz olabilir: electron_Cagir , electronlari_Cagir , muon_Cagir ...

Iste objects.py tam olarak bunun icin var.



Bir olay icinde,

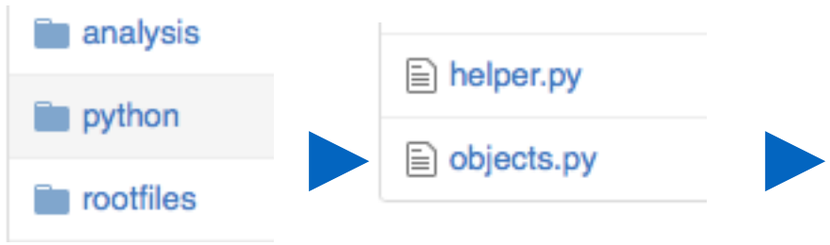
bir suru cesit parcaciktan

bir suru oluyor.

Burada, sadece bir electron cagiran fonksiyon yazdik.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```

Bu dosyayi ileride yeni nesnelerle dolduracagiz.



electronlari ozelliklerine gore sozluk haline getiriyor.

=>electron sozlugune istedigim yeni bir eleman ekleyebilirim.

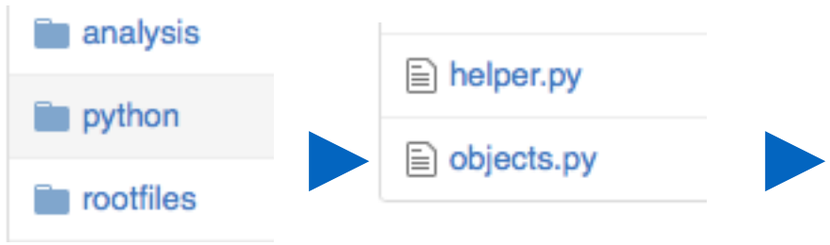
electron[“sira”]=x

burada sira electronun en yuksek pt’ye sahip olmaya gore olay icindeki sirasi olabilir.

eger electron en yuksek pt’li ise electron[“sira”]=0

ikinci ise electron[“sira”]=1 diyebilirim.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Daha en basinda elektronlarim uzerine bir takim secmeler uygulayabilirim:

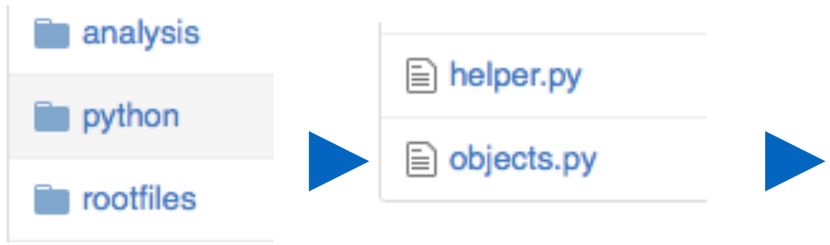
pt: bir onceki hupp dersinde gormustuk, transverse momentum.

diyorum ki bana pt'si 15 GeV'den buyuk electronlari ver.

eta < 2.5 demek asiri ileri (forward) bölge ile ilgilenmiyorum.

hadOverE: birazcik adi ustunde hadronic kalorimetre enerjisi / electromagnetik calorimetre enerjisi. Electronlarimin daha cok ECAL'da enerji birakmasini bekliyorum yani HadOverE 1'den epey kucuk olmali.

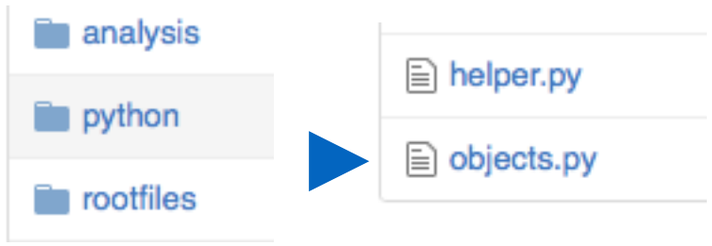
```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Bir electronu nasil alacagimi ogrendim.

Simdi sira olayin tum electronlarina ulasmakta.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```

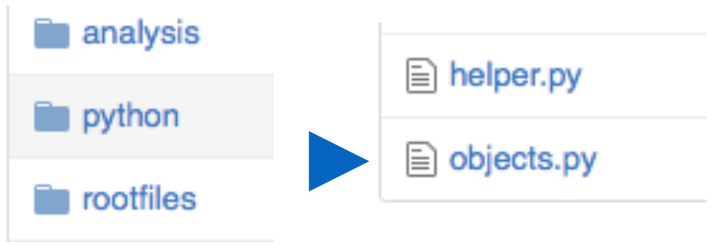


Bir electronu nasil alacagimi ogrendim.

Simdi sira olayin tum electronlarina ulasmakta.

1) olay icindeki toplam electron sayisini al

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```

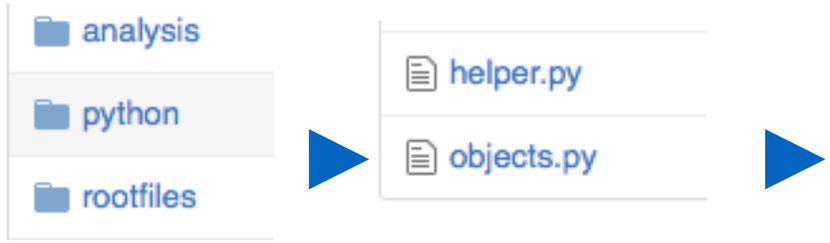


Bir electronu nasil alacagimi ogrendim.

Simdi sira olayin tum electronlarina ulasmakta.

1) olay icindeki toplam electron sayisini al
2) bir for dongusu olustur.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```

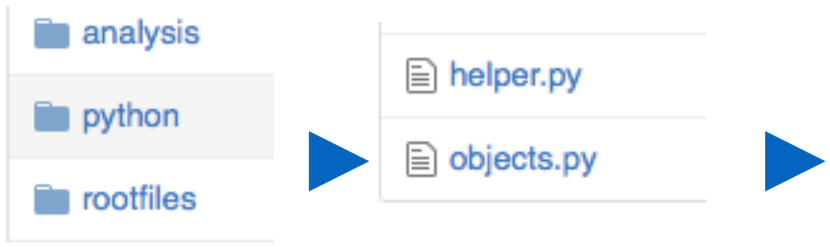


Bir electronu nasil alacagimi ogrendim.

Simdi sira olayin tum electronlarina ulasmakta.

1) olay icindeki toplam electron sayisini al
2) bir for dongusu olustur.
   - tek electron alma fonksiyonunu cagir.

```
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.Pt').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Bir electronu nasil alacagimi ogrendim.

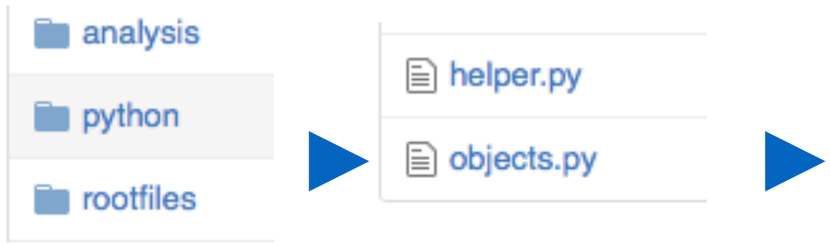
Simdi sira olayin tum electronlarina ulasmakta.

1) olay icindeki toplam electron sayisini al
2) bir for dongusu olustur.
   - tek electron alma fonksiyonunu cagir.

3) doing icinde electronlari electronlar listesine ekliyoruz.

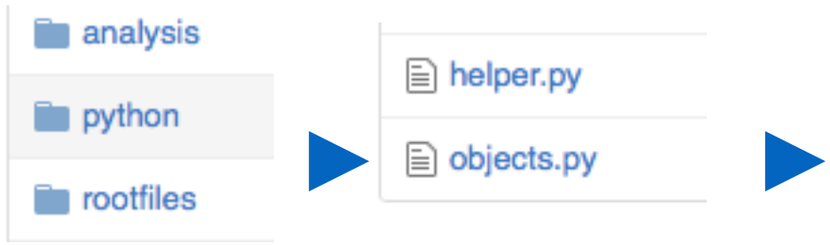
=> bu fonksiyon bize electron kutuphanelerinden olusan electronlar listesini donduruyor.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



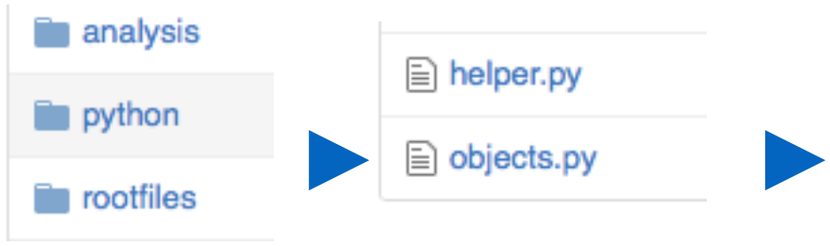
Olay icinde ki Electronlari almayi ogrendim.
Bu fonksiyon electronlari kullanarak Z'ye benzer electron ciftine bozunan ve notr olan parcaciklari bulmamiza yardim edicek.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Olay icinde ki Electronlari almayi ogrendim.
Bu fonksiyon electronlari kullanarak Z'ye benzer electron ciftine bozunan ve notr olan parcaciklari bulmamiza yardim edicek.
1) Lorentz vectorlerimi tanimliyorum

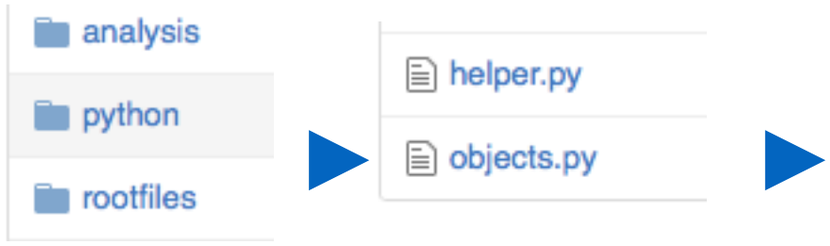
```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Olay icinde ki Electronlari almayi ogrendim.
Bu fonksiyon electronlari kullanarak Z'ye benzer electron ciftine bozunan ve notr olan parcaciklari bulmamiza yardim edicek.

1) Lorentz vectorlerimi tanimliyorum
2) 2 electronlu olaylari seciyorum

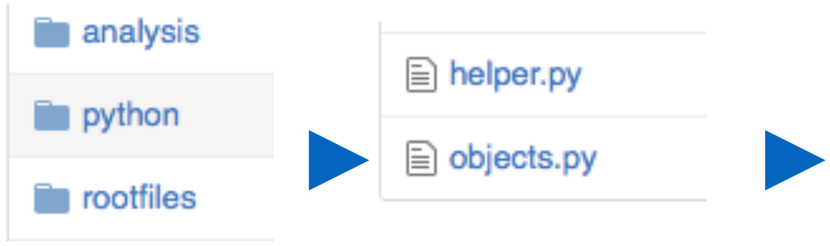
```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Olay icinde ki Electronlari almayi ogrendim.
Bu fonksiyon electronlari kullanarak Z'ye benzer electron ciftine bozunan ve notr olan parcaciklari bulmamiza yardim edicek.

1) Lorentz vectorlerimi tanimliyorum
2) 2 electronlu olaylari seciyorum
3) Bu satir electron ciftlerinden bir for dongusu ceviriyor. (reversed: listeyi tersine cevirecek.)

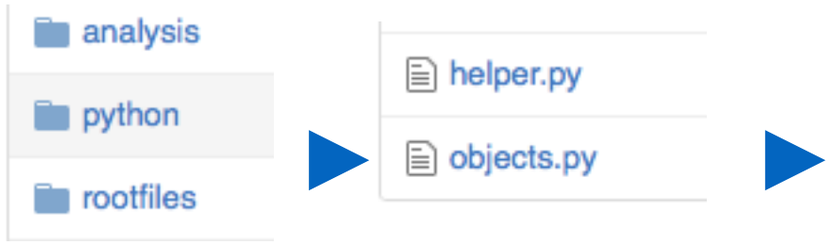
```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```



Olay icinde ki Electronlari almayi ogrendim.
Bu fonksiyon electronlari kullanarak Z'ye benzer electron ciftine bozunan ve notr olan parcaciklari bulmamiza yardim edicek.

1) Lorentz vectorlerimi tanimliyorum
2) 2 electronlu olaylari seciyorum
3) Bu satir electron ciftlerinden bir for dongusu ceviriyor. (reversed: listeyi tersine cevirecek.)
4) electron ciftimin yuksuz olmasi sartini burada olusturuyorum.

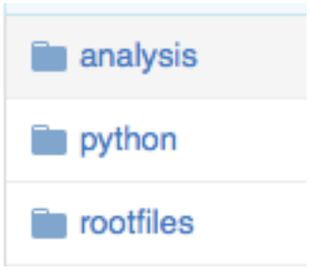
```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```

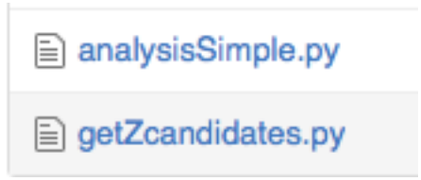


Olay icinde ki Electronlari almayi ogrendim.
Bu fonksiyon electronlari kullanarak Z'ye benzer electron ciftine bozunan ve notr olan parcaciklari bulmamiza yardim edicek.

1) Lorentz vectorlerimi tanimliyorum
2) 2 electronlu olaylari seciyorum
3) Bu satir electron ciftlerinden bir for dongusu ceviriyor. (reversed: listeyi tersine cevirecek.)
4) electron ciftimin yuksuz olmasi sartini burada olusturuyorum.
5) Lorentz vectorlerin toplami da vectorel toplamdir. ve bu satir bana Z tipi parcaciklarin Lorentz vectorunu donduruyor.

```python
import ROOT

#get one electron from an event
def getElectron(tree,j):
  pt = tree.GetLeaf('Electron.PT').GetValue(j)
  eta = tree.GetLeaf('Electron.Eta').GetValue(j)
  phi= tree.GetLeaf('Electron.Phi').GetValue(j)
  charge = tree.GetLeaf('Electron.Charge').GetValue(j)
  Ntrk = tree.GetLeaf('Electron.Ntrk').GetValue(j)
  hadOverE = tree.GetLeaf('Electron.EhadOverEem').GetValue(j)
  cand={'pt':pt, 'eta':eta, 'phi':phi,'charge':charge,'Ntrk':Ntrk,'hadOverE':hadOverE}
  if pt > 15 and eta<2.5 and hadOverE < 0.2:
    return cand

#get all electrons insede an event
def getElectrons(tree):
  nelectrons = tree.GetLeaf('Electron_size').GetValue()
  electrons=[]
  for e in range(int(nelectrons)):
    electron = getElectron(tree,e)
    if electron:
      electrons.append(electron)
  return electrons

#get 2 electron lorentz vector
def getZ(eles):
  e1 = ROOT.TLorentzVector()
  e2 = ROOT.TLorentzVector()
  Z = ROOT.TLorentzVector()
  if len(eles)==2:
    for perm in [eles, reversed(eles)]:
      elep,elen = perm
      if elep['charge'] == -elen['charge'] and elep['charge'] >0:
        e1.SetPtEtaPhiM(elen['pt'],elen['eta'],elen['phi'],0)
        e2.SetPtEtaPhiM(elep['pt'],elep['eta'],elep['phi'],0)
        Z = e1+e2
    return Z
```





Bu kisa kod bize bir olay dongusu yardimiyla iki electronlu olaylari kullanarak bu electron ciftinin degismez kutle dagilimini verecek.

```
import os,sys
import ROOT
sys.path.append('../python')
from helper import DrawNicePlot, DrawAdvancedPlot
from objects import getElectron, getElectrons, getZ
from math import log,sqrt, cos, sin, atan2, sinh

file = ROOT.TFile('../rootfiles/pgs_events_Q1.root')
tree = file.Get('LHCO')

h_Zeemass = ROOT.TH1F('h_Zeemass','Z mass',2000,0,2000)
h_Zeemass.Sumw2()

fileName='eeInvMass'
path = '../plots/'
if not os.path.exists(path):
  os.makedirs(path)

n_events = tree.GetEntries()
#print n_events
for i in range(n_events):
  tree.GetEntry(i)
  eles = getElectrons(tree)
  #print eles
  Z1 = getZ(eles)
  if Z1: h_Zeemass.Fill(Z1.M())  #Z is a Lorentz vector

DrawAdvancedPlot(h_Zeemass,'ee M_{inv.}',path,fileName)
```

Yeni root dosyasi da github'da var !

Bir TH1F histogrami tanimladim.

Dosya adi ve nereye kaydetmek istedigimi soyledim. ustune bu klasorun var olup olmadigini kontrol ettim.

tree icindeki olay sayisini aldim.

bu olay sayisi kadar dongu kurdum.

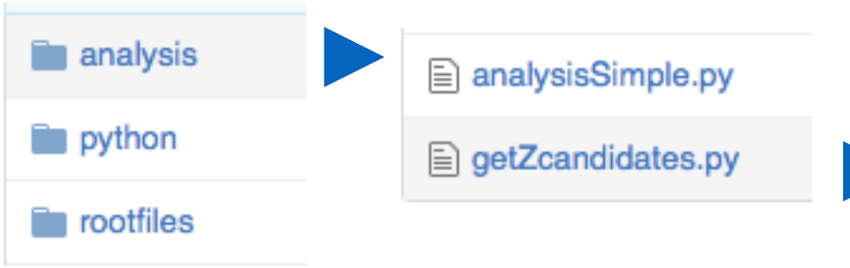
i'nci olayin icine girdim

daha once yazdigimiz fonksiyon ile elektronlari al.

degismez kutleyi verir

daha once yazdigimiz fonksiyon ile Z adaylarinin lorentz vectorunu al.

bu fonksiyonu bir onceki dersten biliyoruz.

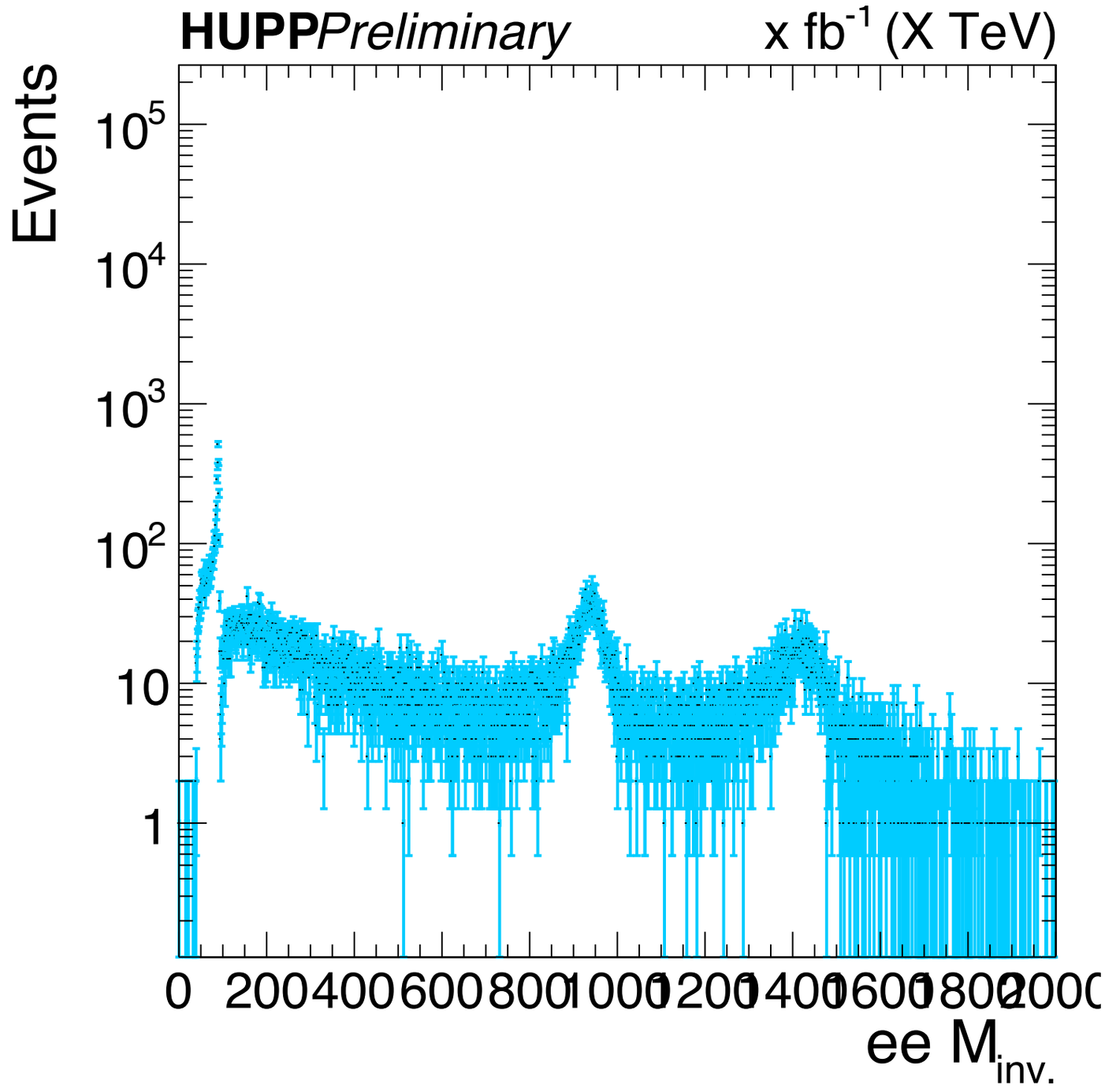


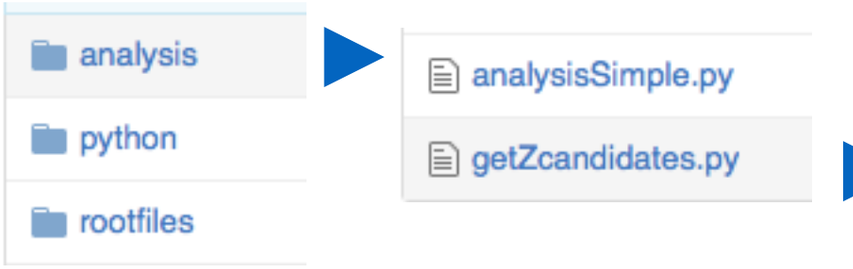
Sonuc:

Unutmayalim,
Histogramimiza sadece yuksuz bir kaynaktan gelen electron ciftlerinin degismez kutlelerini doldurduk.

Eger ciftlerin geldigi belli kaynaklar olmasaydi, ciftler rastgele olusuyor olsaydi duz bir dagilim gorurdum.

Sadece bir kaynak olsaydi, bu kaynagin kutlesi etrafinda bir gaussian dagilim olustururlardi.





Sonuc:

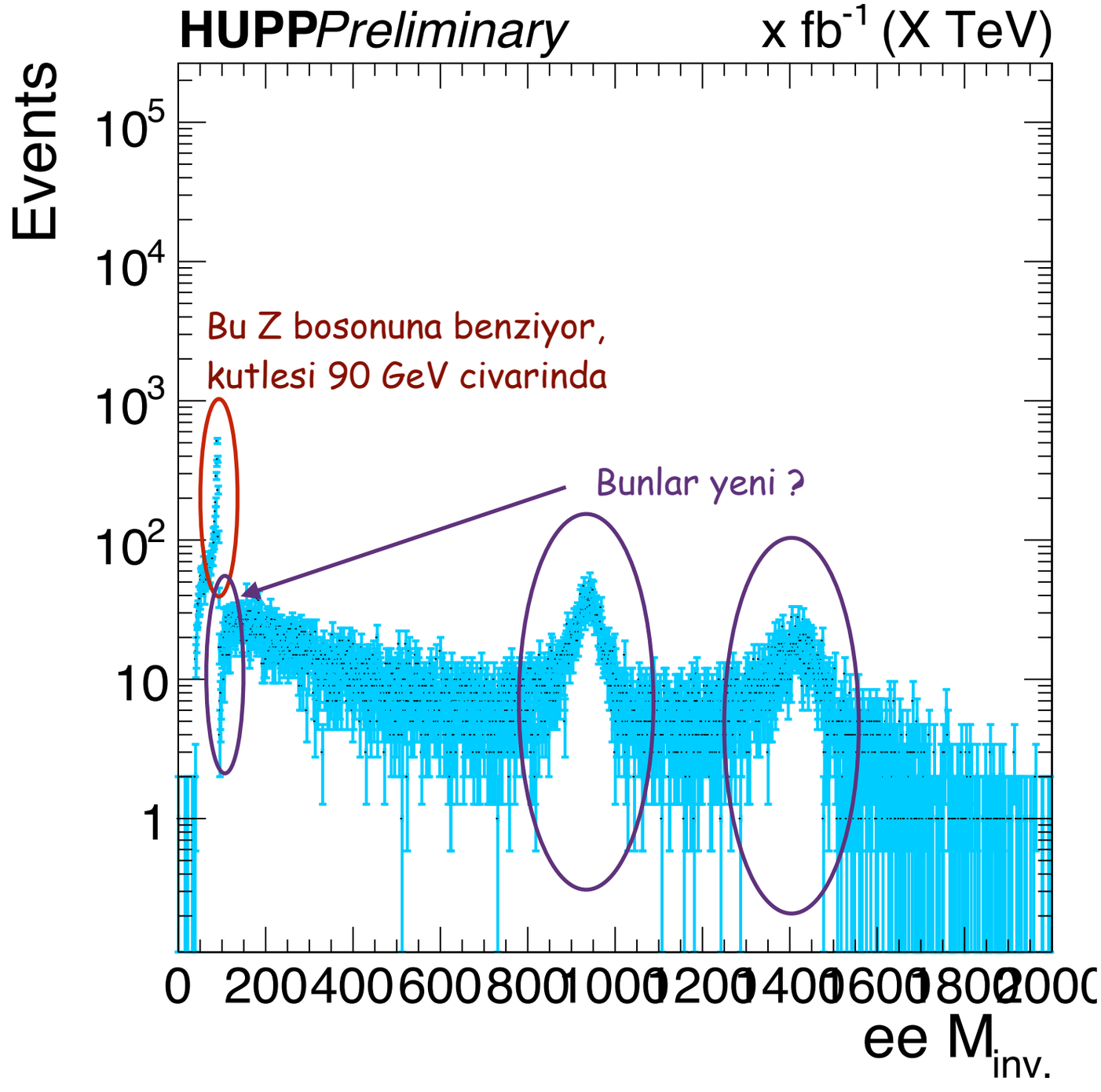
Unutmayalim,
Histogramimiza sadece yuksuz bir kaynaktan gelen electron ciftlerinin degismez kutlelerini doldurduk.

Eger ciftlerin geldigi belli kaynaklar olmasaydi, ciftler rastgele olusuyor olsaydi duz bir dagilim gorurdum.

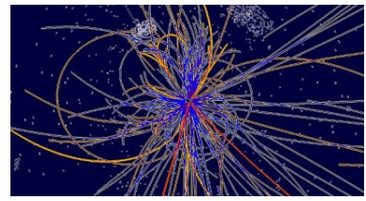
Sadece bir kaynak olsaydi, bu kaynagin kutlesi etrafinda bir gaussian dagilim olustururlardi.

Ama yanda gordugum Z'den hemen sonra bir dusme ki bu da yeni bir fizige isaret, ve iki tane daha yuksek kutlelerde tepe.

Bunlarin ne olabilecegini ve bu sekillere nasil fit edebilecegim gelecek dersin konusu !!



Neden Z bosonu ?



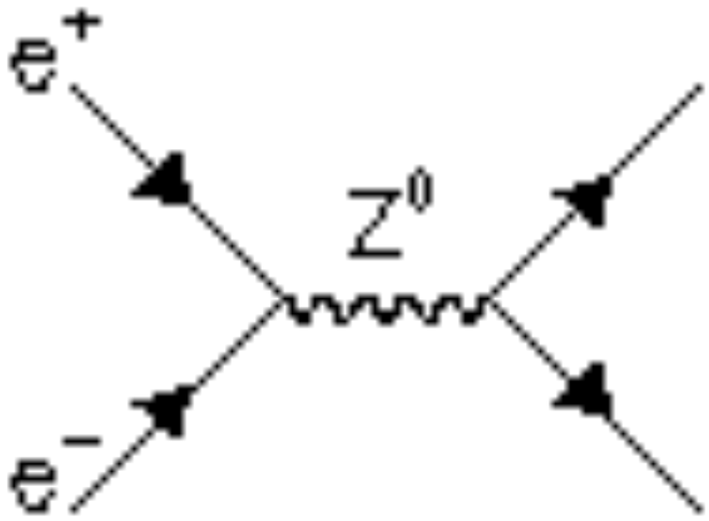
Z->ee/mumu ...

Kutlesini cok iyi biliyorum 91.1876±0.0021 GeV

=> eger electronlarimin muonlarimin Z'den gelmeyenlerini istiyorsam kolaylikla bu kutle araligini (+-20GeV) disarliyabilirim.

=> tam tersi de olabilir o zaman bu araligi secerim.

=> electronu veya muon'u ne kadar duzgun bulabildigimi (Tag and Prob), isaretle ve olc, yolu ile olcebilirim.



e isaretle

Oyle katlar koy ki electron olduguna nerdeyse emin olayim.

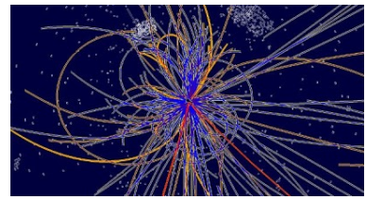
e ölç

En esnek electron secimini kullan.

Her bir olaydaki olcmelik electronlarini say. **Payda**ya koy. Sonra neyin olasiligini olcmek istiyorsan o kati uygula ve bu kati gecenlerin sayisini **pay**'a koy.

payda = butun olcmelik electronlarin sayisi
Trigger Verimliligi:
pay = trigger'i atesleyen olcmelik electronlarin sayisi
Yeniden yapilandirma (Reconstruction) Verimliligi:
pay = generated bir esi olan olcmelik electronlarin sayisi

Neden Z bosonu ?



Z->ee/mumu ...

Kutlesini cok iyi biliyorum 91.1876±0.0021 GeV

=> eger electronlarimin muonlarimin Z'den gelmeyenlerini istiyorsam kolaylikla bu kutle araligini (+-20GeV) disarliyabilirim.

=> tam tersi de olabilir o zaman bu araligi secerim.

=> electronu veya muon'u ne kadar duzgun bulabildigimi (Tag and Prob), isaretle ve olc, yolu ile olcebilirim.

Hepimize Ödev:

1)iki electronlu olaylari sec.
2)birinci+ikinci electronun degismez kutlesinin Z kutle araliginda olmasini iste. (90+-30 GeV)
3)bunlardan birtanesi cok siki electron kriterlerini gecsin.
4)payda: diger electronlarin sayisi
5)diger electronlara otekine uyguladigin daha siki katlari uygula.
6) pay: katlardan sonra digerlerinin sayisi
7) duzgun electron bulma olasigin pay/payda
8) bu olasiligi olctugun electronun Et'si ve etasina gore cizdir.

Bazı yardımcı bilgiler burada bulunabilir:
http://atlas.physicsmasterclasses.org/tr/zpath_messung.htm

- **ROOT TLorentzVector sinifi:**
  https://root.cern.ch/doc/master/classTLorentzVector.html

Gelecek Ders:

Once .. odeve bakicaz

Sonra .. Buldugumuz tepelere fit yapicaz

Ek Sayfalar

git

install git: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

```
pc00:HUPP eceasilar$ git init
Initialized empty Git repository in /Users/eceasilar/HUPP/.git/
pc00:HUPP eceasilar$ git remote add hupp-central  https://github.com/easilar/HUPP-codeHelper
pc00:HUPP eceasilar$ git remote show
hupp-central
pc00:HUPP eceasilar$ git fetch hupp-central
remote: Counting objects: 24, done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 24 (delta 6), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (24/24), done.
From https://github.com/easilar/HUPP-codeHelper
 * [new branch]      master     -> hupp-central/master
pc00:HUPP eceasilar$ git branch
pc00:HUPP eceasilar$ git checkout -b master-hupp
Switched to a new branch 'master-hupp'
pc00:HUPP eceasilar$ git branch
pc00:HUPP eceasilar$ git push hupp-central master-hupp
```

Bu ismi biz uydurduk

Henuz bir dal'imiz yok

"-b" yeni dal yaptik. Dal'in adi master-huppp

gorebilmek icin once dal'imizi git havuzuna atmaliyiz

Boyle hemen atamayiz once aticam sozu vermeliyim git'e

git

Bir dosya oulsturup bunu yeni dal'imiz ile birlikte git'a gonderelim.

```
pc00:HUPP eceasilar$ vi analysisSimple.py
pc00:HUPP eceasilar$
pc00:HUPP eceasilar$
pc00:HUPP eceasilar$ git add analysisSimple.py
pc00:HUPP eceasilar$ git commit -am "first file: analysisSimple.py"
[master-hupp (root-commit) aa61545] first file: analysisSimple.py
 Committer: ece asilar <eceasilar@pc00.idemog.oeaw.ac.at>
[...Biraz daha fazla ileti...]
pc00:HUPP eceasilar$
pc00:HUPP eceasilar$ git push hupp-central master-hupp
Username for 'https://github.com': easilar
Password for 'https://easilar@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 532 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/easilar/HUPP-codeHelper
 * [new branch]      master-hupp -> master-hupp
pc00:HUPP eceasilar$
pc00:HUPP eceasilar$ git branch
* master-hupp
pc00:HUPP eceasilar$
```

Dosyami actim , kaydettim.

git havuzuna ekledim

Ve git'e bunu bildirdim

Simdi olusturdugum dal'i git'e gonderebilirim.
bildirdigim tum degisiklikleri
git'e tasiyacak.

Yuklemek icin git
kullanici adimi ve
sifremi istedi.

Artik hersey git havuzunda,
bana bulundugum dal'i gosteriyor.

.bash_profile

```
###Python and ROOT setting
export ROOTSYS=/cern/newROOT/root/
#export PYTHONPATH=/usr/bin/python
export PYTHONPATH=$ROOTSYS/lib:$PYTHONPATH
#export LD_LIBRARY_PATH=$ROOTSYS/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=$ROOTSYS/lib:$PYTHONDIR/lib:$LD_LIBRARY_PATH
export DYLD_LIBRARY_PATH=$ROOTSYS/lib:$DYLD_LIBRARY_PATH
export PATH=$ROOTSYS/bin:$PATH

export PATH=$PATH:/usr/local/bin:/usr/X11R6/bin
export CFLAGS="-I/usr/local/include"
export CPPFLAGS="-I/usr/local/include"
export LDFLAGS="-L/usr/local/lib"

#history search
bind "^[[4~":end-of-line
bind "^[[1~":beginning-of-line
bind "^[[3~":delete-char
bind "\e[5~":history-search-backward
bind “\e[6~":history-search-forward

if [[ $- == *i* ]]
then
    bind '"\e[A": history-search-backward'
    bind '"\e[B": history-search-forward'
fi

##Some useful alias
alias ll='ls -la'
alias ..='cd ..'
alias ...='cd ../..'
alias ....='cd ../../..'
alias hupp='cd /Users/eceasilar/HUPP/analysis/'
```