An Example Analysis

Gökhan Ünel / *UC Irvine*

HPFBU okulu - Ocak 2009

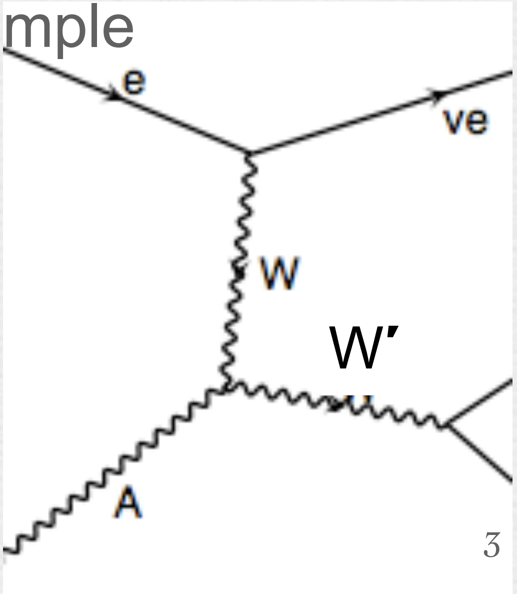
10 Ocak 2009

In previous lectures

- LHC is providing data
  - Discovery of new particles: exciting
- New accelerators are also being planned (LHeC, ILC, CLIC..)
  - We need to investigate the discovery prospects of such machines
- So what do we need? (After a topic selection)
  - Prepose a measurement
  - Know the collider
  - Enter the model into the computer & produce MC events
  - Simulations, analysis, obtain the results

Topic

- Some BSM theories (e.g. GUTs) propose a new W-like vector boson : W'
- The new boson W'
  - W' has the same SM interactions, there is the γWW' vertex.
  - W' has its mass greater than W
- If we build a γ e collider, can we discover this particle?
  - Production example



starting from W' decay products, is a discovery possible

Example channel for signal= 2j+MET

How to do it?

- Is it worth the effort?
  - We put our model in a signal generator.
  - How many signal events per year?
  - What is the signature of the signal?
  - What are the background events from SM?

What is the most distinctive feature of the signal?

- Produce MC events for signal, using BSM
  - Do the detector simulation.
  - How can I distinguish the signal "easily" from the background?
  - How can I reconstruct the signal events?

How can I trigger on the signal?

- Produce MC events for background, using SM
  - Do the detector simulation.
  - Is the signal distinguishable?

if the answer is "Yes" then we can send the proposal..

The goals & the Tools

signal

γ e→W′ v→ 2j+MET

background

γ e→ 2j+MET

- Cain, GuineaPIG
  - Luminosity calculation
- Pythia, CompHEP, MadGraph
  - Event generators
- Pythia
  - Parton showering, hadronization..
- PGS / Delphes
  - Detector simulation
- ROOT
  - Data Analysis

The algorithm

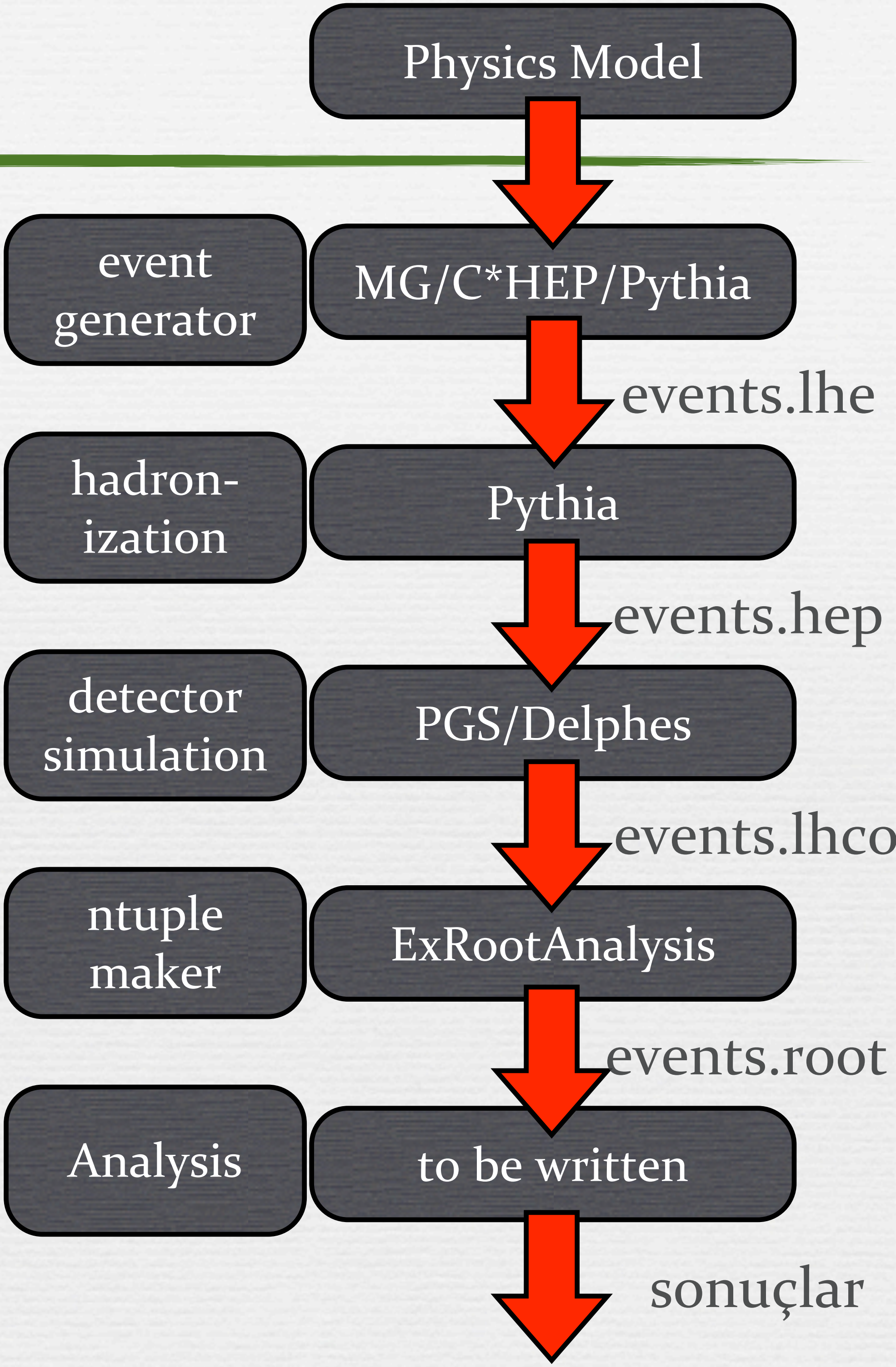
1. Signal events can be produced by our first tool: CompHEP

   1.1. γ e→W′ ν→ 2j+MET

2. MG can calculate the SM background for the same final state particles.

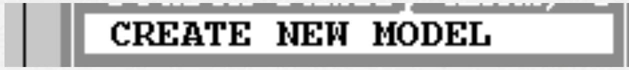
3. Both signal and background events to be processed by the detector simulation.

4. An analysis program is to be written by us to separate the signal events from the BG ones.



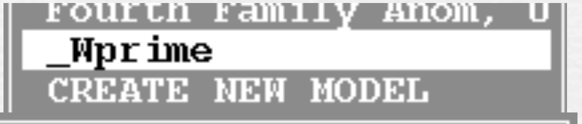
writing a BSM model .

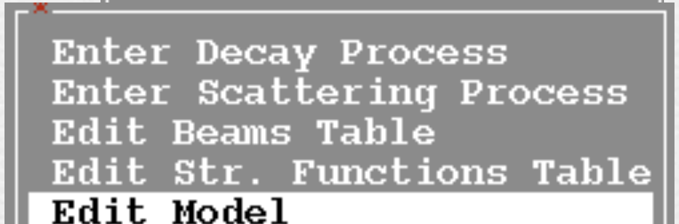
- Get from the list at the start of CompHEP:



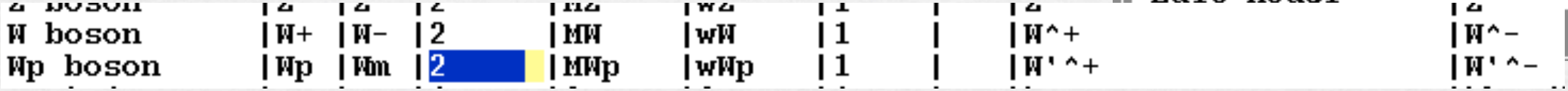
CREATE NEW MODEL

  - Lets call the new model Wprime, start from “SM, Unitary gauge”.
  - Choose the new model, start modifying.



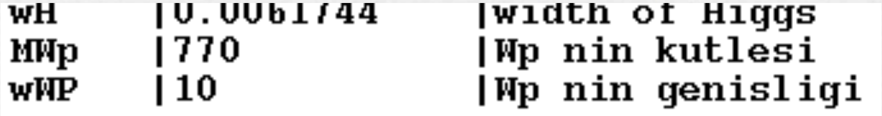


- Add the new boson (Particles)
  - Same as W, except mass and width



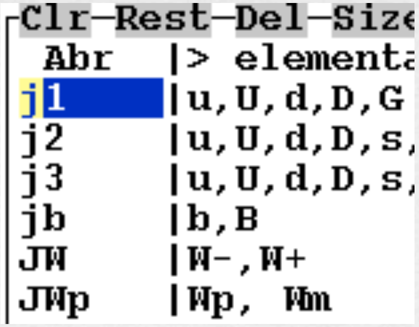
Z boson	Z	Z	2	MZ	wZ	1		Z	Z
W boson	W+	W-	2	MW	wW	1		W^+	W^-
Wp boson	Wp	Wm	2	MWp	wWp	1		W'^+	W'^-

- Wp’s properties are set in the (Variables) part.
  - assume m=770, random width:



wH	0.0061744	width of Higgs
MWp	770	Wp nin kutlesi
wWP	10	Wp nin genisligi

- (Constraints) not used,
- W and Wp jets will be used later (Composite)

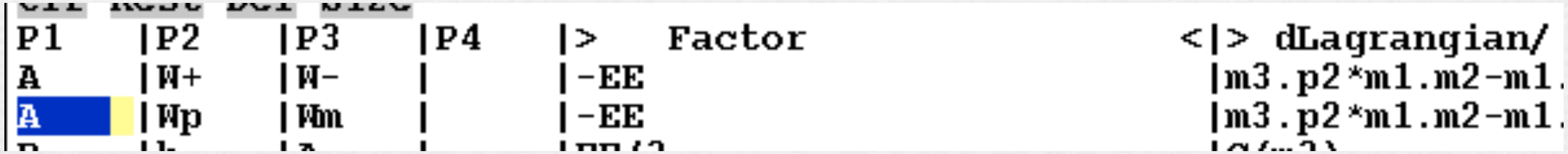


Clr-Rest-Del-Size

Abr	> elementa
j1	u,U,d,D,G
j2	u,U,d,D,s,
j3	u,U,d,D,s,
jb	b,B
JW	W-,W+
JWp	Wp, Wm

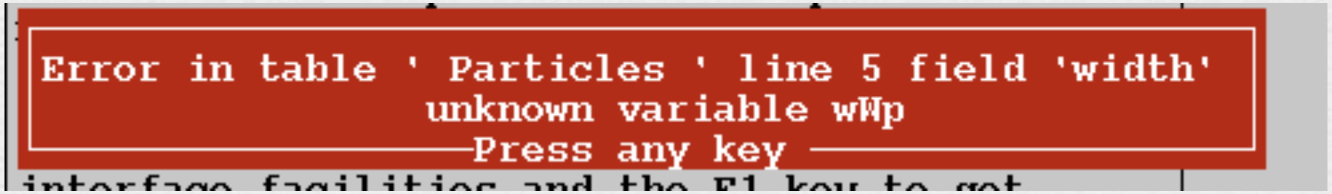
writing a BSM model ..

- Check all lines in the (Lagrangian) containing a W.
  - copy & paste *(or use an editor, though I don't recommend it)*
  - in the copied lines replace W with Wp :

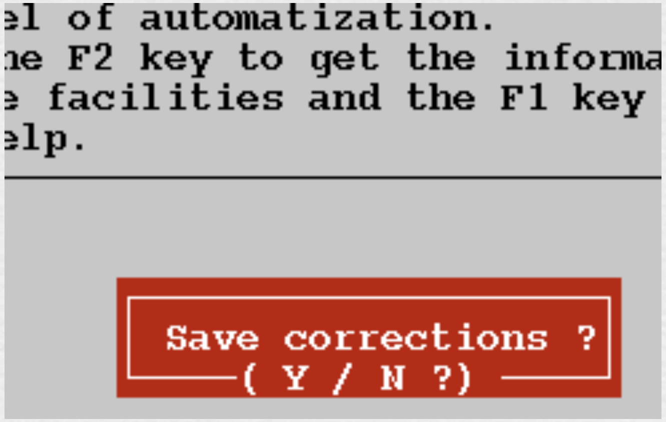


```
P1     |P2     |P3     |P4     |>   Factor                 <|> dLagrangian/
A      |W+     |W-     |       |-EE                         |m3.p2*m1.m2-m1.
A      |Wp     |Wm     |       |-EE                         |m3.p2*m1.m2-m1.
```

- Check that what we did so far is correct:
  - An error window pops up in case of problems

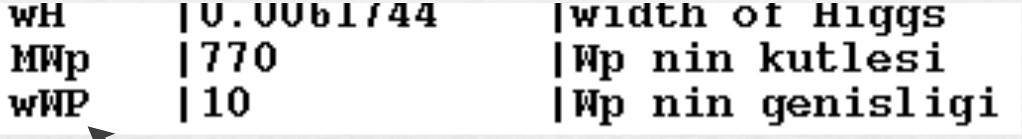


```
Error in table ' Particles ' line 5 field 'width'
            unknown variable wWp
              Press any key
interface facilities and the F1 key to get
```



```
el of automatization.
he F2 key to get the informa
e facilities and the F1 key
elp.

      Save corrections ?
          ( Y / N ?)
```

  - *Small typo in the previous page !!*



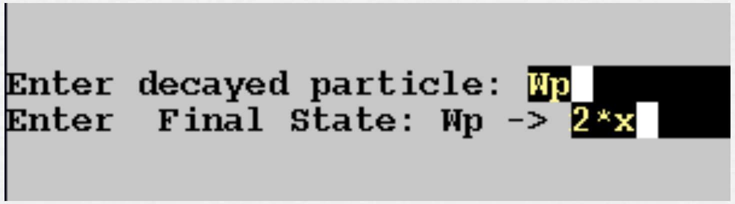
```
wH     |0.0061744      |width of Higgs
MWp    |770            |Wp nin kutlesi
wWP    |10             |Wp nin genisligi
```

should have been wWp

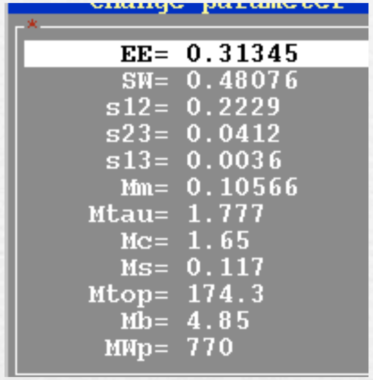
W′ width:

W′→2x

- Once the model is written we can obtain the decay width

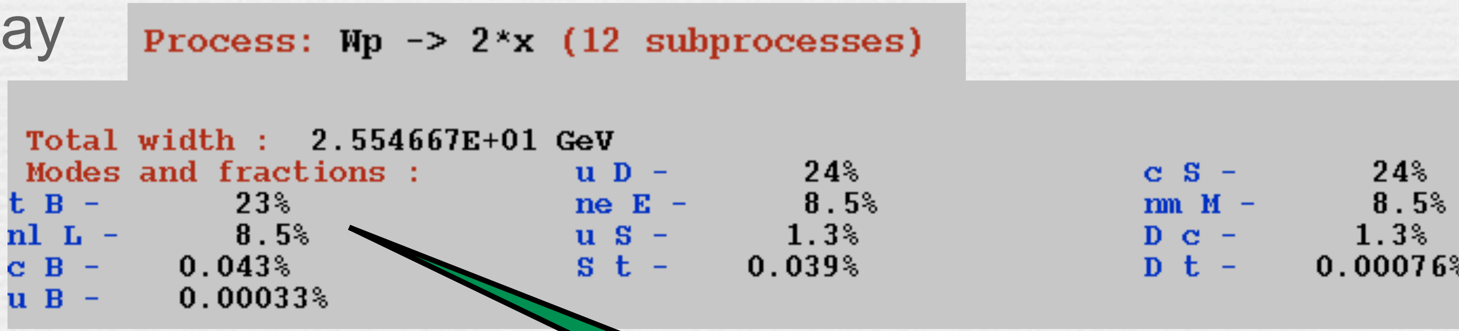


```
Enter decayed particle: Wp
Enter  Final State: Wp -> 2*x
```



```
change parameter
   EE= 0.31345
   SW= 0.48076
  s12= 0.2229
  s23= 0.0412
  s13= 0.0036
   Mm= 0.10566
 Mtau= 1.777
   Mc= 1.65
   Ms= 0.117
 Mtop= 174.3
   Mb= 4.85
  MWp= 770
```

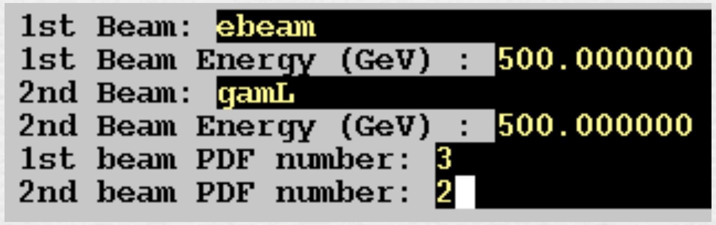
- at m=770 GeV, Γ=25.5GeV
  - ✓ 8.5x3=25.5% leptonic decay
  - 74.5% hadronic decay
    - 23% top-bottom pairs
    - top FS is not



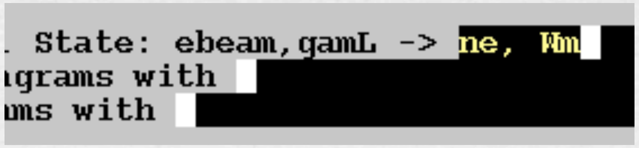
```
Process: Wp -> 2*x (12 subprocesses)

 Total width :  2.554667E+01 GeV
 Modes and fractions :           u D -       24%                c S -       24%
t B -       23%                  ne E -      8.5%               nm M -      8.5%
nl L -      8.5%                 u S -       1.3%               D c -       1.3%
c B -     0.043%                 S t -     0.039%               D t -    0.00076%
u B -     0.00033%
```

~51% 2j decays

- Consider the collisions
  - incoming particles:
  - outgoing particles:



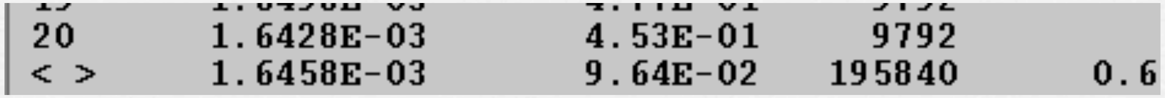
```
1st Beam: ebeam
1st Beam Energy (GeV) : 500.000000
2nd Beam: gamL
2nd Beam Energy (GeV) : 500.000000
1st beam PDF number: 3
2nd beam PDF number: 2
```



```
. State: ebeam,gamL -> ne, Wm
agrams with
ams with
```

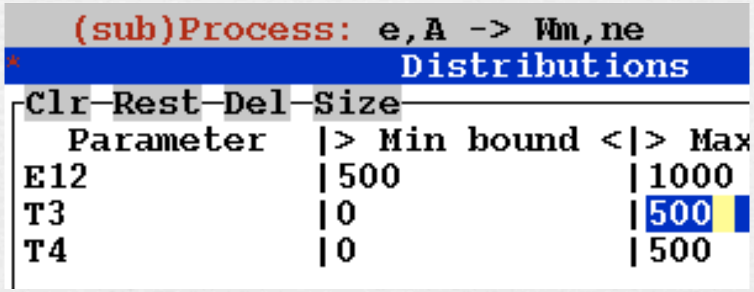
W′ production /year

e γ→ W′ν



```
20        1.6428E-03        4.53E-01     9792
< >       1.6458E-03        9.64E-02   195840        0.6
```

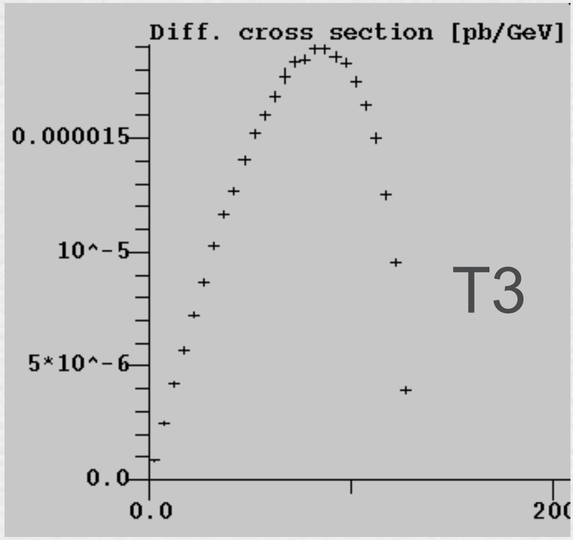
- Undecayed signal
  - calculate the xsection and check the kinematical distributions
- 1.64fb x section expected
- ILC Luminosity $2.8x10^{34}/cm^2/s$
  - eγ: $\sim 2.8x10^{33}/cm^2/s$
    - 1 year $\sim 10^7$ seconds, or equivalently: $\sim 28fb^{-1}$

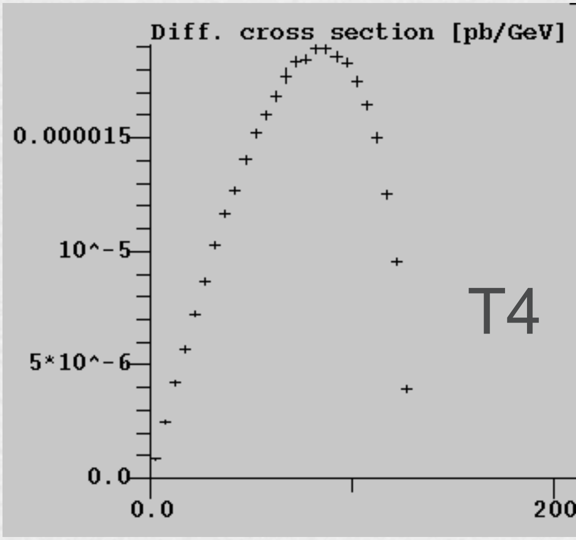


```
   (sub)Process: e,A -> Wm,ne
*                        Distributions
Clr-Rest-Del-Size
   Parameter    |> Min bound <|> Max
E12             |500             |1000
T3              |0               |500
T4              |0               |500
```

- Expected W′ / year is
  - 1.64 × 28 ~= 46 .
- 50% decays to 2jets
  - we expect 23 event/year
  - or in other words,

  $\sigma_{effective}$ = 1.64x50%≈0.8fb

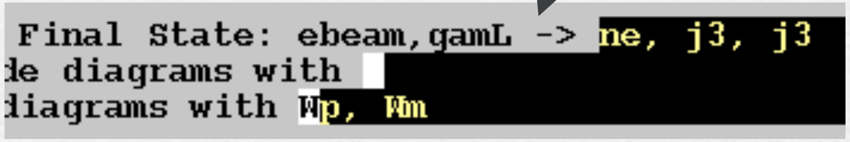




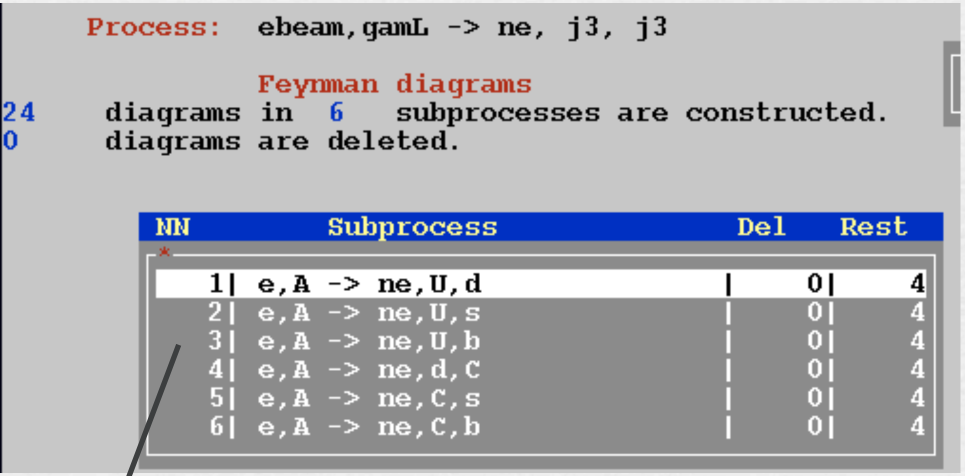
W′ decay - x-check

e γ→ W′ν→jj ν

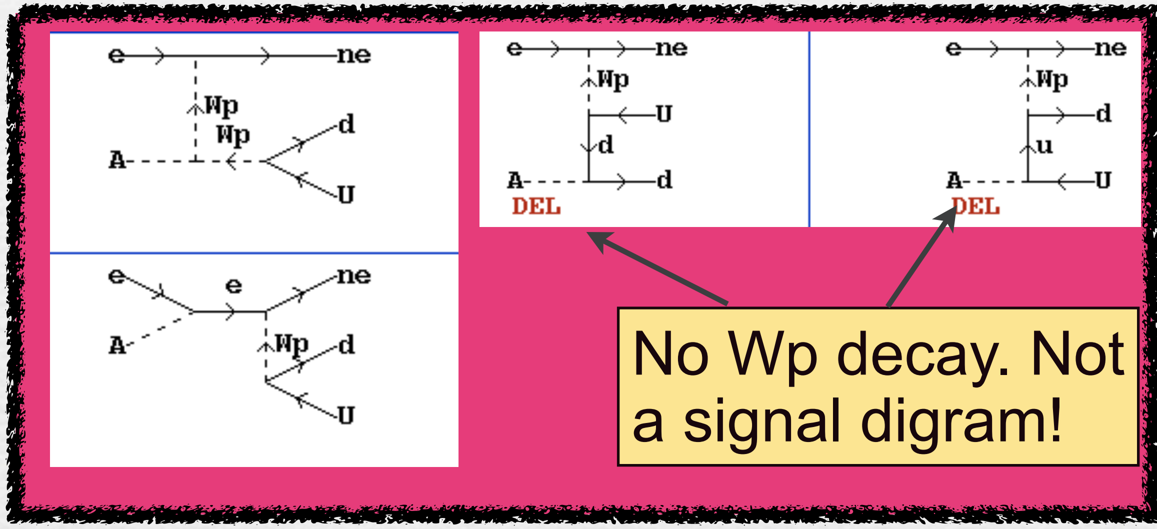
- Force W′ to decay hadronically, consider only the “signal part”: a bit of *smart choice* and a bit of *cleaning*:



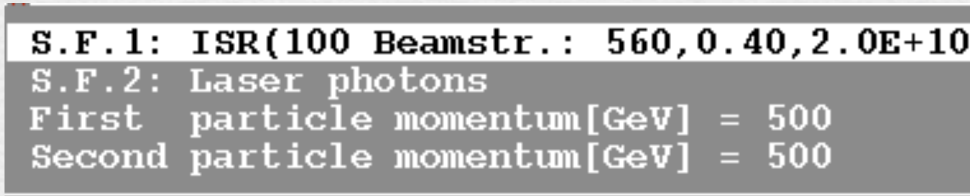
```
 Final State: ebeam,gamL -> ne, j3, j3
de diagrams with
diagrams with Wp, Wm
```



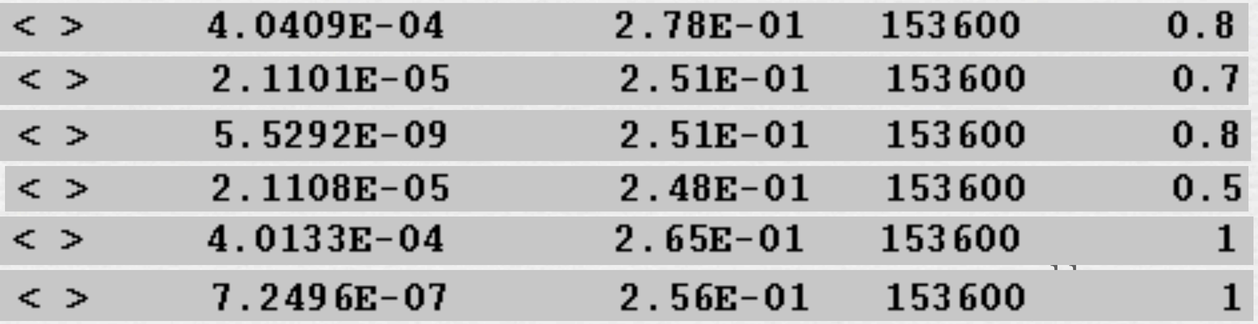
```
     Process:   ebeam,gamL -> ne, j3, j3

                Feynman diagrams
24      diagrams in  6    subprocesses are constructed.
0       diagrams are deleted.


      NN          Subprocess              Del    Rest
       *
           1|  e,A -> ne,U,d            |      0|     4
           2|  e,A -> ne,U,s            |      0|     4
           3|  e,A -> ne,U,b            |      0|     4
           4|  e,A -> ne,d,C            |      0|     4
           5|  e,A -> ne,C,s            |      0|     4
           6|  e,A -> ne,C,b            |      0|     4
```



No Wp decay. Not a signal digram!



```
 S.F.1:  ISR(100 Beamstr.:  560,0.40,2.0E+10
 S.F.2:  Laser photons
 First   particle momentum[GeV]  =  500
 Second  particle momentum[GeV]  =  500
```



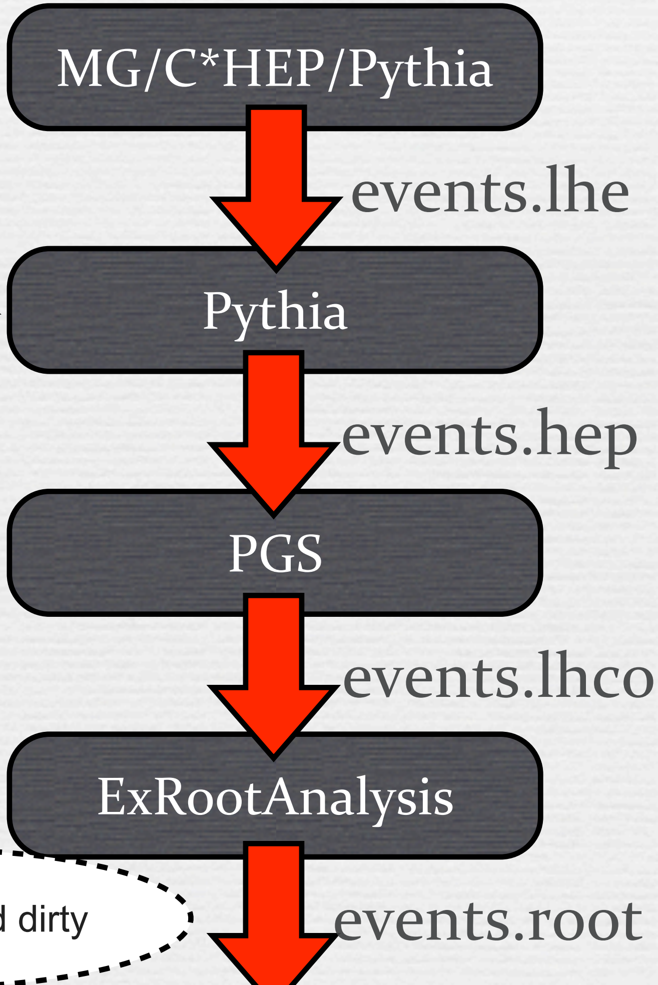
```
< >        4.0409E-04         2.78E-01     153600          0.8
< >        2.1101E-05         2.51E-01     153600          0.7
< >        5.5292E-09         2.51E-01     153600          0.8
< >        2.1108E-05         2.48E-01     153600          0.5
< >        4.0133E-04         2.65E-01     153600            1
< >        7.2496E-07         2.56E-01     153600            1
```

✓total cross section = 0.82fb, as expected

W′ event production

- No need to produce the signal events for all sub processes.
  - Produce events only from those with cross sections of 10^{-7}pb or more.
- produce events from the 4 remaining subprocess and merge
  - use the `mix` as before to get Mixed.lhe
- To inject into the chain we need a new sw
  - `comphep-interfaces-3.0.2.tgz`
  - `tar xzf comphep-interfaces-3.0.2.tgz`
  - `cd ./comphep-interfaces-3.0.2`
  - `./configure ; make`
  - `make setup WDIR= $HOME/istapp2011/chep_if/`
  - `cd $HOME/istapp2011/chep_if/`
- We’ll use pyhtia
  - copy the `Mixed.lhe` in chep_if directory
  - edit the `pytmain.f` :
    - implement writing out events in STDHEP format
  - `make pyhia ; nedit INPARM.dat`
  - `./cpythia.exe`

proper way. Quick and dirty solution?



background - cross section & event production

event generator

hadronization

detector simulator

ntuple maker

Analysis

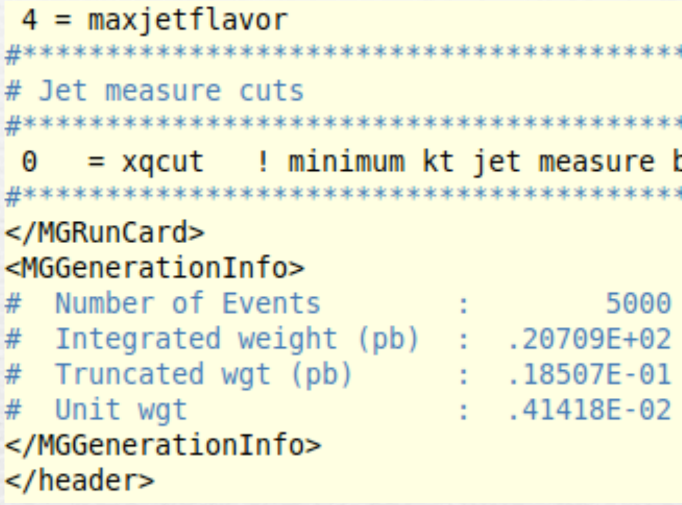
- Produce events (example: MadGraph)
  - cd Events ; ls
  - unweighted_events.lhe *should be there*
- Hadronization
  - ../bin/run_pythia ; ls
  - pythia_events.hep *should be there*
- Detector simulation
  - ../bin/run_delphes ; ls
  - delphes_events.lhco *should be there*
- Prepare the ROOT file
  - ../../ExRootAnalysis/ExRootLHCOlympicsConverter delphes_events.lhco sm.root ; ls
  - sm.root

done in MG homework

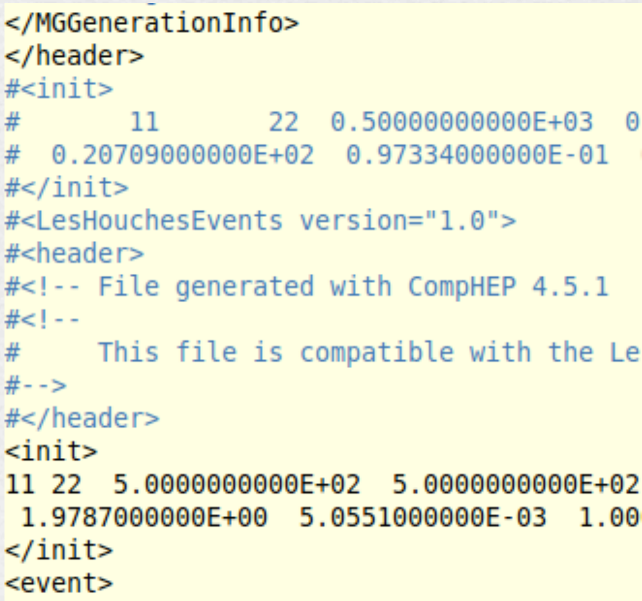
the file to analyze

Warning: Quick and dirty

- Both MG and Comphep output in LHE format
- in MG/phytia/Delphes suite, we can run pythia on LHE events
- BUT it will work with some gymnastics
  - `cp  bg_unweighted_events.lhe.gz si.lhe.gz`
  - `gunzip si.lhe.gz; cp si.lhe si_backup.lhe`
  - edit the file, remove all lines after the </header>
  - `cat si.lhe Mixed.lhe >> unweighted_events.lhe`
  - `tail -1 si_backup.lhe >> unweighted_events.lhe`
  - edit the file, to remove Comphep specific few lines
  - `../bin/run_pythia`
  - `../bin/run_delphes`
  - `../../ExRootAnalysis/ExRootLHCOlympics Converter unweighted_events.lhe si.root`
- `the output is your si.root`



```
 4 = maxjetflavor
#***********************************
# Jet measure cuts
#***********************************
 0   = xqcut   ! minimum kt jet measure b
#***********************************
</MGRunCard>
<MGGenerationInfo>
#  Number of Events         :        5000
#  Integrated weight (pb)   :  .20709E+02
#  Truncated wgt (pb)       :  .18507E-01
#  Unit wgt                 :  .41418E-02
</MGGenerationInfo>
</header>
```



```
</MGGenerationInfo>
</header>
#<init>
#       11        22  0.50000000000E+03  0
#  0.20709000000E+02  0.97334000000E-01
#</init>
#<LesHouchesEvents version="1.0">
#<header>
#<!-- File generated with CompHEP 4.5.1
#<!--
#     This file is compatible with the Le
#-->
#</header>
<init>
11 22  5.0000000000E+02  5.0000000000E+02
 1.9787000000E+00  5.0551000000E-03  1.00
</init>
<event>
```

Analysis

each hero has a different (yogurt) eating style.

- her yiğidin yoğurt yiyişi farklıdır.

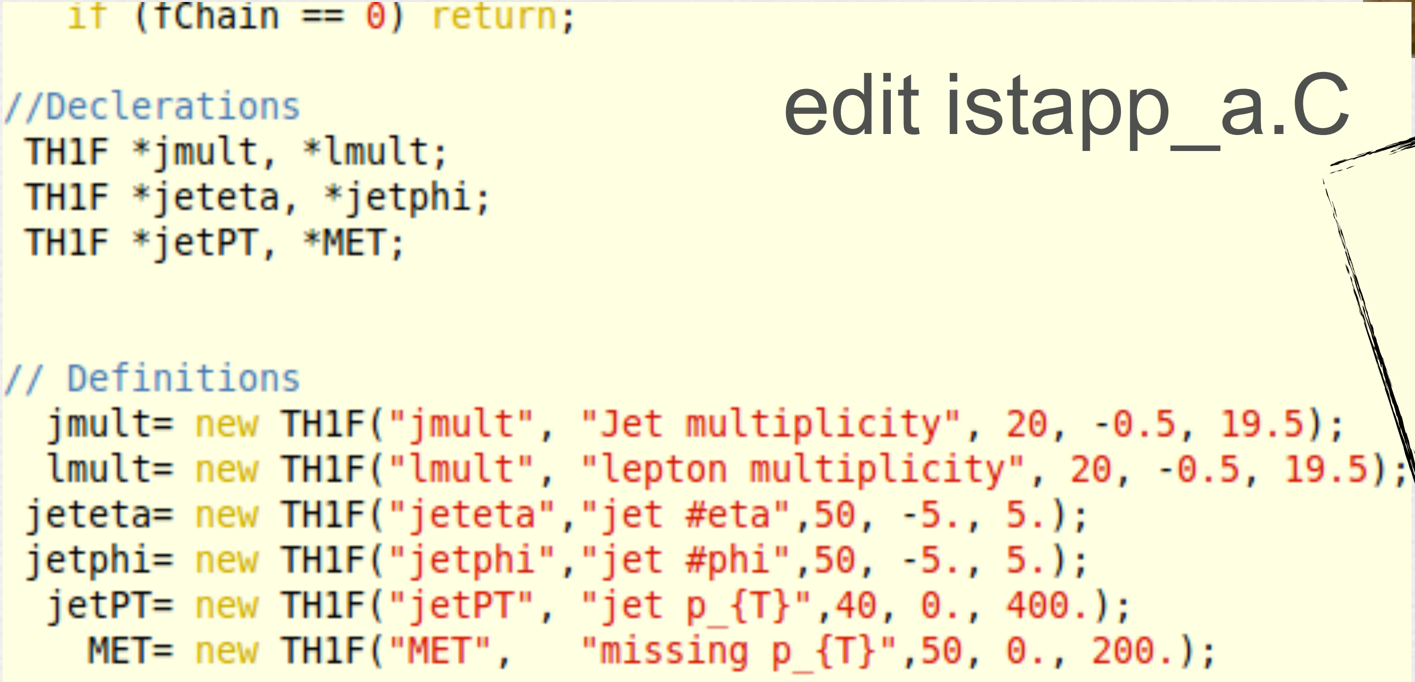
Lets first check the signal

- Work with your Laptop (ROOT analysis) such that you are can work all the time & everywhere. “Laptopable” analysis.

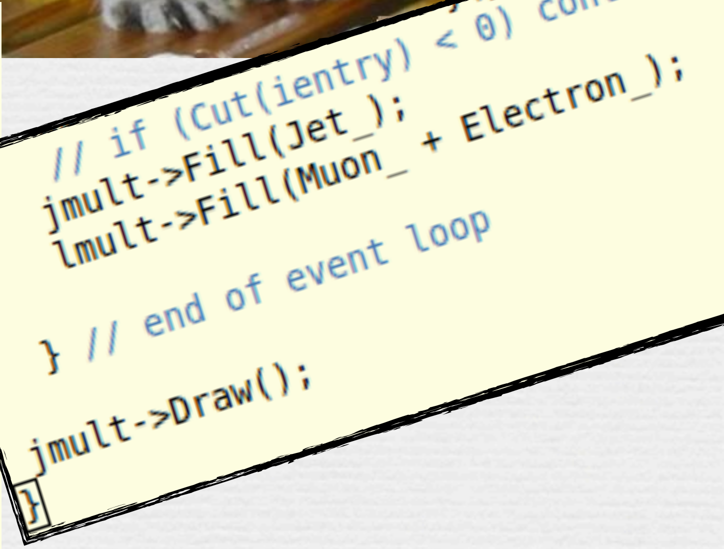
We saw in the ROOT Lecture

- load the events file to root and
- LHCO->MakeClass("istapp_a")
- Lets check few variables

edit istapp_a.C



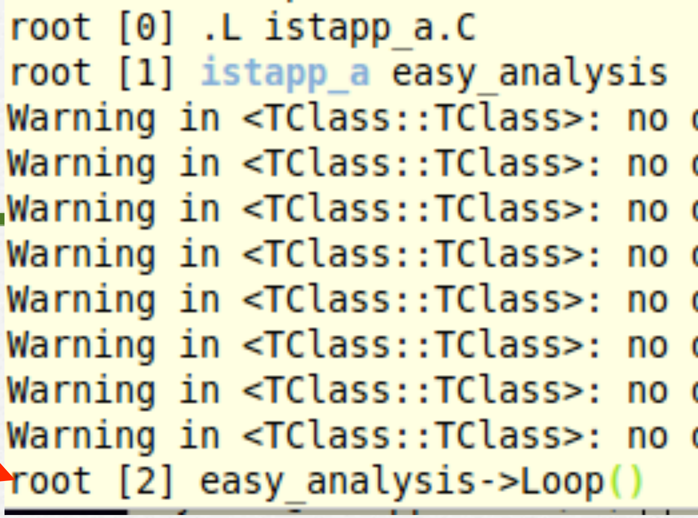
```
   if (fChain == 0) return;

//Declerations
 TH1F *jmult, *lmult;
 TH1F *jeteta, *jetphi;
 TH1F *jetPT, *MET;


// Definitions
  jmult= new TH1F("jmult", "Jet multiplicity", 20, -0.5, 19.5);
  lmult= new TH1F("lmult", "lepton multiplicity", 20, -0.5, 19.5);
 jeteta= new TH1F("jeteta","jet #eta",50, -5., 5.);
 jetphi= new TH1F("jetphi","jet #phi",50, -5., 5.);
  jetPT= new TH1F("jetPT", "jet p_{T}",40, 0., 400.);
    MET= new TH1F("MET",   "missing p_{T}",50, 0., 200.);
```



```
   // if (Cut(ientry) < 0) continue;
   jmult->Fill(Jet_);
   lmult->Fill(Muon_ + Electron_);

 } // end of event loop

jmult->Draw();
}
```

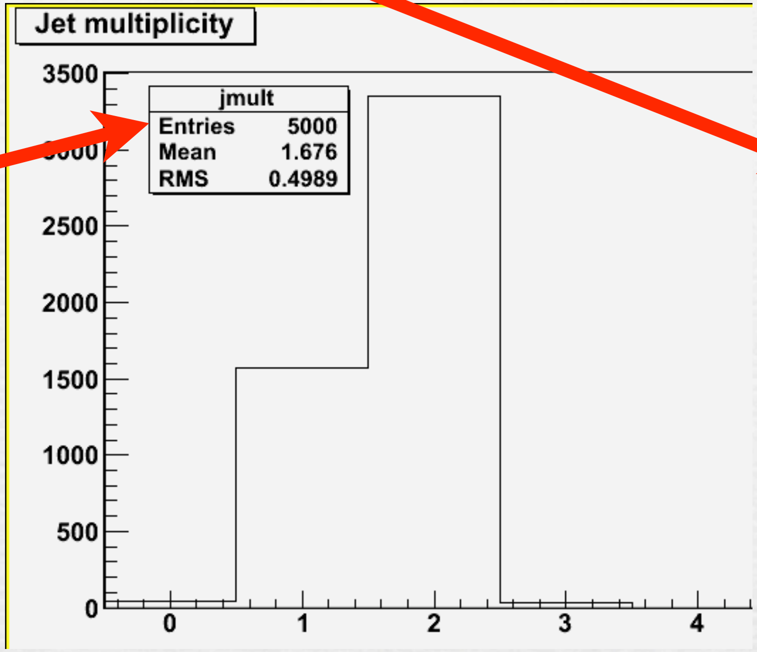
Analysis 2

- execute to see the result.

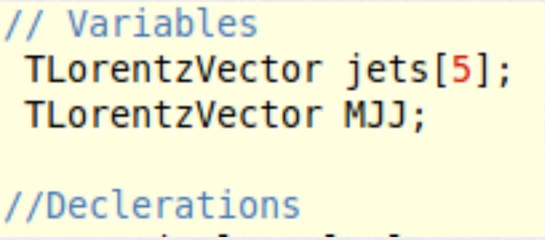


```
root [0] .L istapp_a.C
root [1] istapp_a easy_analysis
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
Warning in <TClass::TClass>: no
root [2] easy_analysis->Loop()
```

  - we expect 1 plot



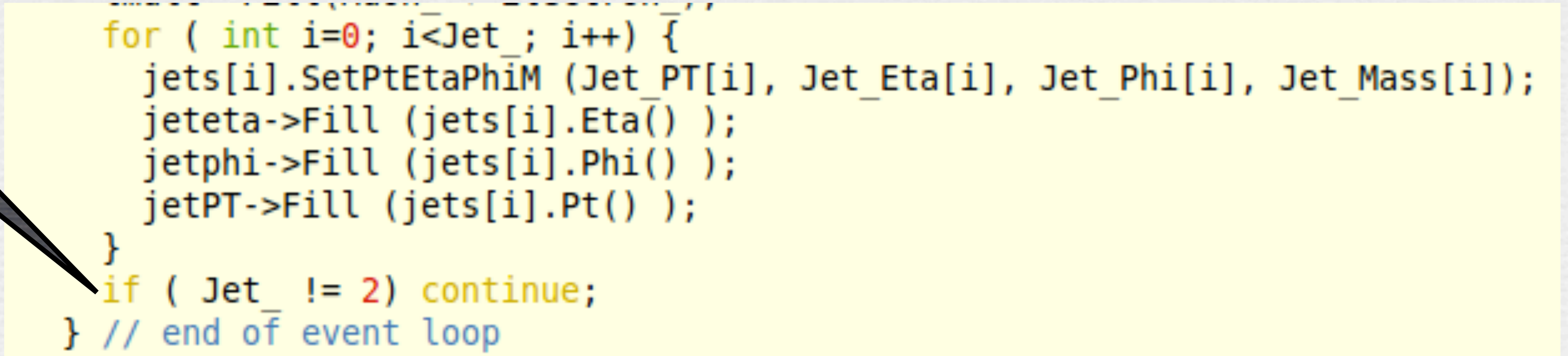
OK!



```
// Variables
 TLorentzVector jets[5];
 TLorentzVector MJJ;

//Declerations
```

- What are the jet properties?

2 jet requirement

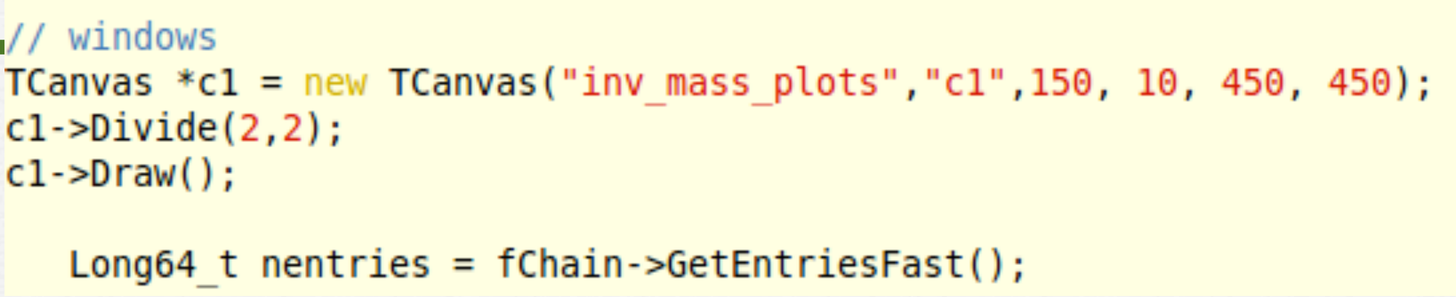


```
    for ( int i=0; i<Jet_; i++) {
      jets[i].SetPtEtaPhiM (Jet_PT[i], Jet_Eta[i], Jet_Phi[i], Jet_Mass[i]);
      jeteta->Fill (jets[i].Eta() );
      jetphi->Fill (jets[i].Phi() );
      jetPT->Fill (jets[i].Pt() );
    }
    if ( Jet_ != 2) continue;
  } // end of event loop
```

No new plots so far

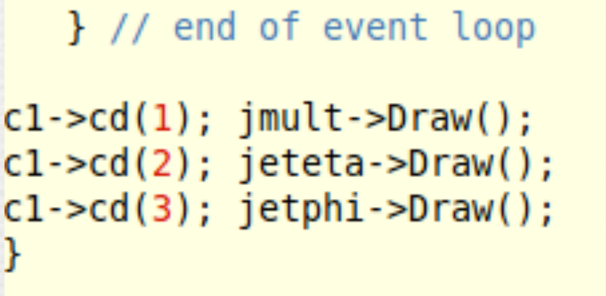
Analysis 3a

- Make a window=>



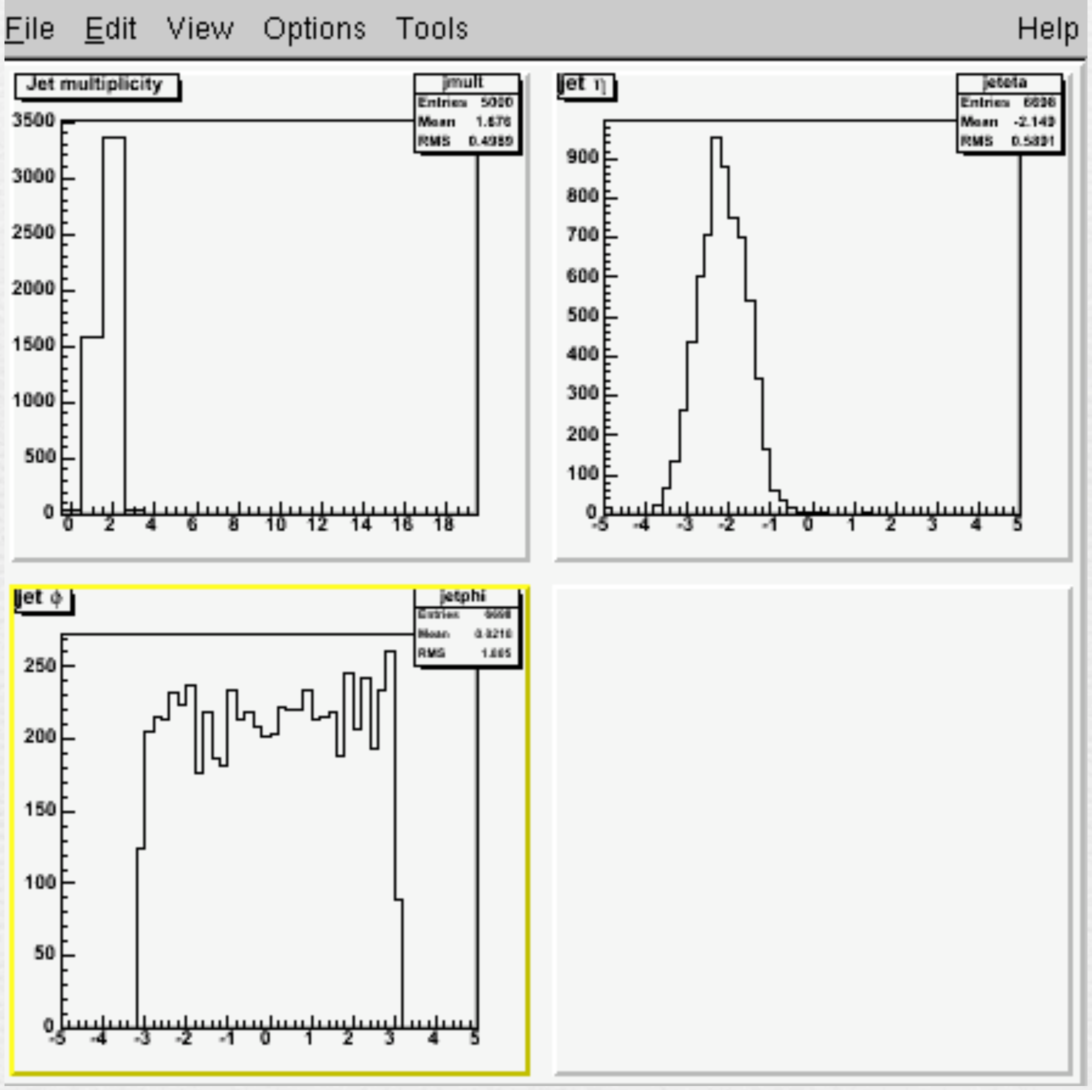
```
// windows
TCanvas *c1 = new TCanvas("inv_mass_plots","c1",150, 10, 450, 450);
c1->Divide(2,2);
c1->Draw();

   Long64_t nentries = fChain->GetEntriesFast();
```

- Plot more histograms

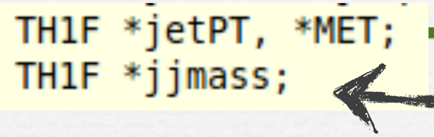


```
   } // end of event loop

c1->cd(1); jmult->Draw();
c1->cd(2); jeteta->Draw();
c1->cd(3); jetphi->Draw();
}
```

- run to get the plots=====>



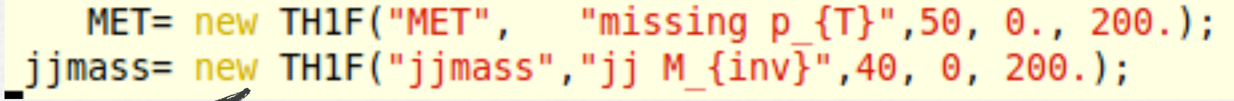
Analysis 3b



```
TH1F *jetPT, *MET;
TH1F *jjmass;
```

New

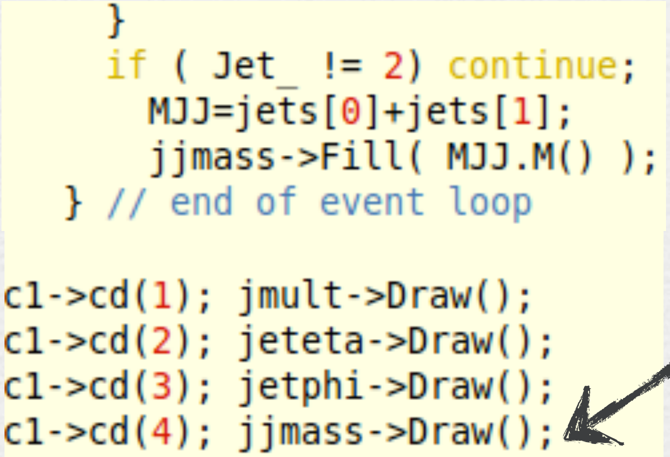
- Prepare the 2 jet invariant mass:



```
   MET= new TH1F("MET",    "missing p_{T}",50, 0., 200.);
jjmass= new TH1F("jjmass","jj M_{inv}",40, 0, 200.);
```

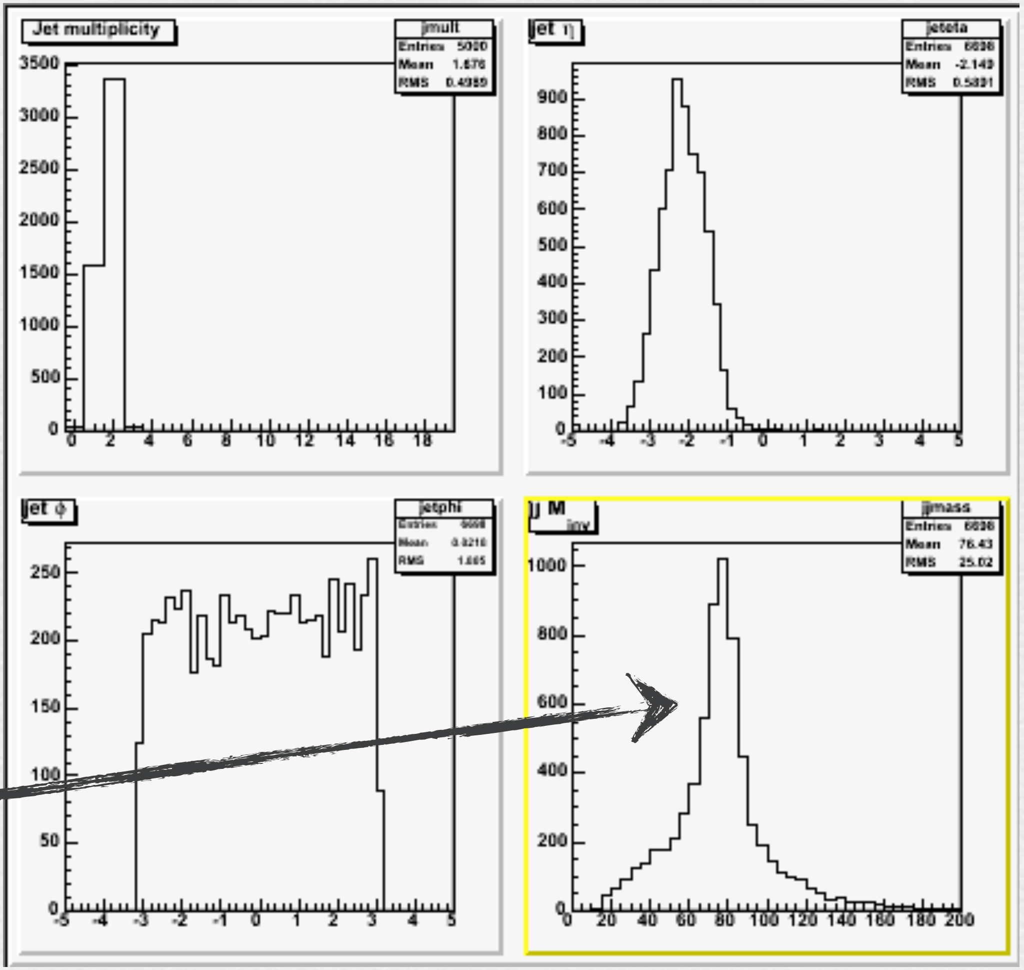
New

- Calculate & draw:



```
      }
      if ( Jet_ != 2) continue;
        MJJ=jets[0]+jets[1];
        jjmass->Fill( MJJ.M() );
   } // end of event loop

c1->cd(1); jmult->Draw();
c1->cd(2); jeteta->Draw();
c1->cd(3); jetphi->Draw();
c1->cd(4); jjmass->Draw();
```

New



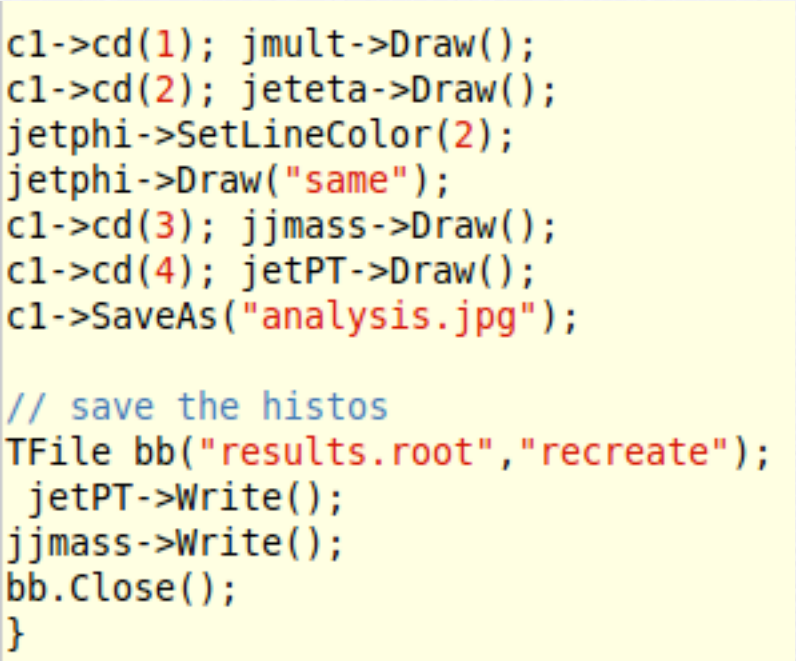
peak ~mW

Background=noise

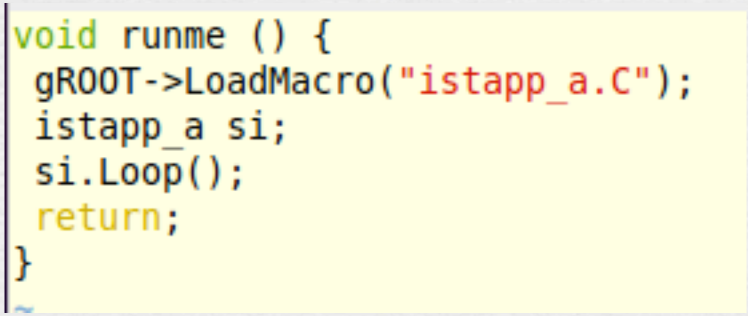
- Signature similar to what we search
- But doesn’t contain the object we search (W′)
  - We need to eliminate these
- in BSM search, SM is background
- We need to separate the needle from the haystack. We will rely on some properties of the needle.
  - “CUT”s
- Which properties could be used?
  - W′ is heavy, so decay products jets carry high p_T .
  - M_{jj} is what we want to measure: dont cut on it.

Analysis 4

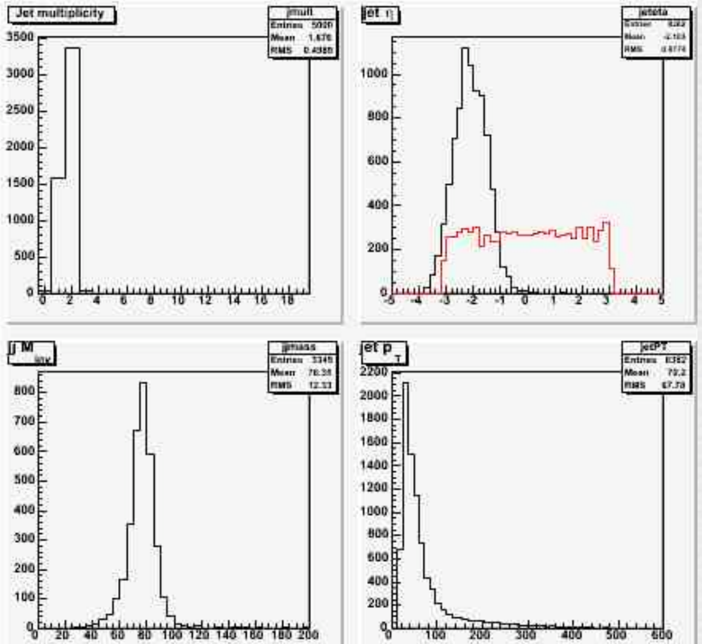
- Draw the Jet pT & record the image.
- Save the produced histograms.
- We will maximally automatize the task.
  - root -q -x runme.C



```
c1->cd(1); jmult->Draw();
c1->cd(2); jeteta->Draw();
jetphi->SetLineColor(2);
jetphi->Draw("same");
c1->cd(3); jjmass->Draw();
c1->cd(4); jetPT->Draw();
c1->SaveAs("analysis.jpg");

// save the histos
TFile bb("results.root","recreate");
 jetPT->Write();
jjmass->Write();
bb.Close();
}
```

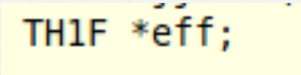


```
void runme () {
 gROOT->LoadMacro("istapp_a.C");
 istapp_a si;
 si.Loop();
 return;
}
```

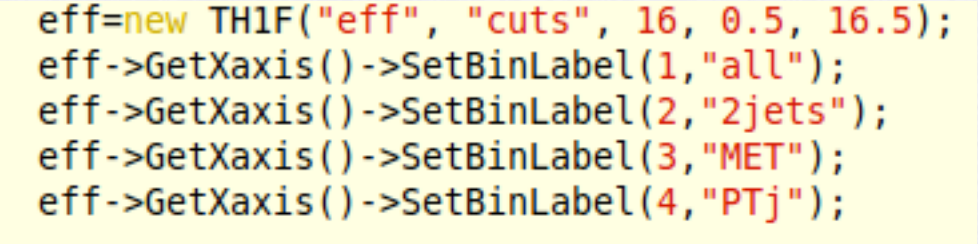


cuts

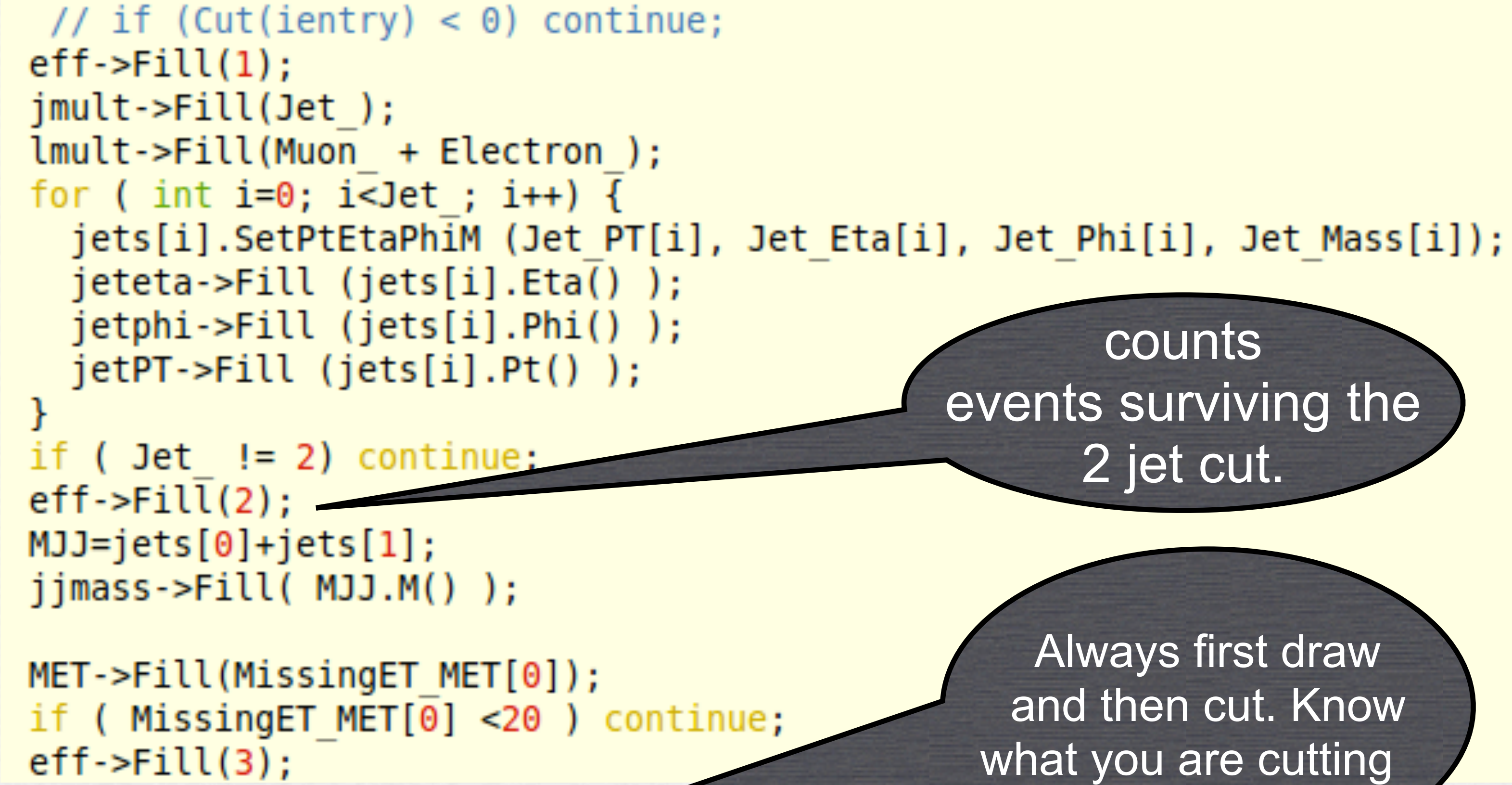
- How many events were lost ?
  - after each cut, count the number of remaining events & save in a histogram



```
TH1F *eff;
```



```
eff=new TH1F("eff", "cuts", 16, 0.5, 16.5);
eff->GetXaxis()->SetBinLabel(1,"all");
eff->GetXaxis()->SetBinLabel(2,"2jets");
eff->GetXaxis()->SetBinLabel(3,"MET");
eff->GetXaxis()->SetBinLabel(4,"PTj");
```

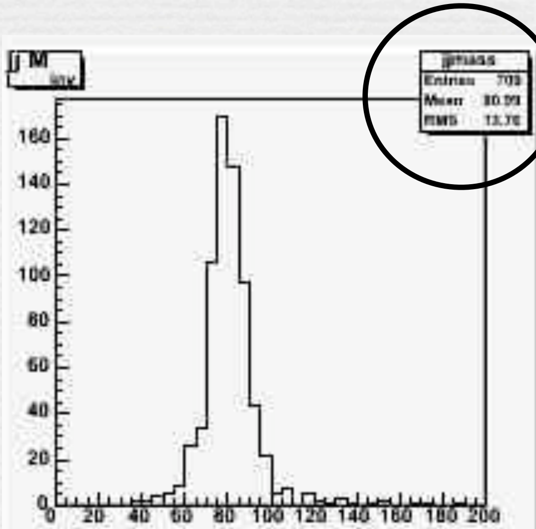


```
 // if (Cut(ientry) < 0) continue;
eff->Fill(1);
jmult->Fill(Jet_);
lmult->Fill(Muon_ + Electron_);
for ( int i=0; i<Jet_; i++) {
  jets[i].SetPtEtaPhiM (Jet_PT[i], Jet_Eta[i], Jet_Phi[i], Jet_Mass[i]);
  jeteta->Fill (jets[i].Eta() );
  jetphi->Fill (jets[i].Phi() );
  jetPT->Fill (jets[i].Pt() );
}
if ( Jet_ != 2) continue;
eff->Fill(2);
MJJ=jets[0]+jets[1];
jjmass->Fill( MJJ.M() );

MET->Fill(MissingET_MET[0]);
if ( MissingET_MET[0] <20 ) continue;
eff->Fill(3);
```

counts events surviving the 2 jet cut.

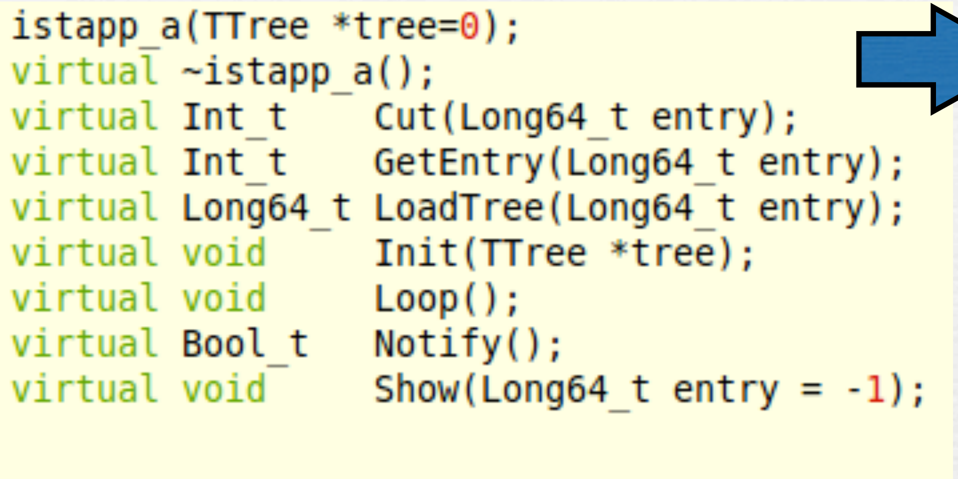
Always first draw and then cut. Know what you are cutting

surviving number of events

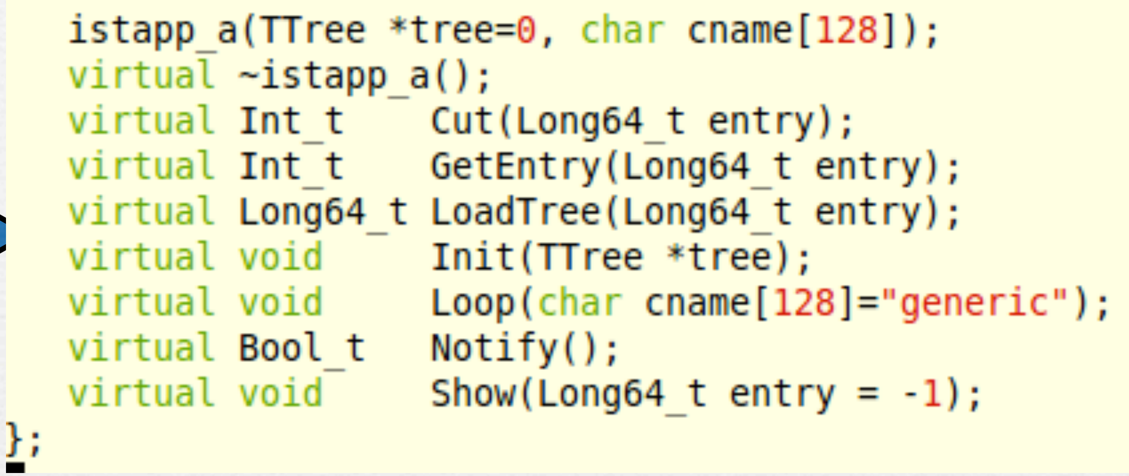


si & sm

edit the .h file

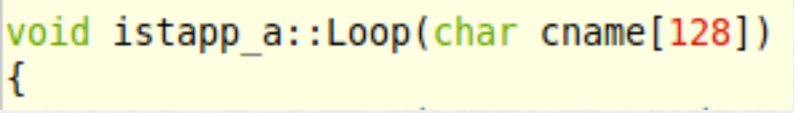


```
istapp_a(TTree *tree=0);
virtual ~istapp_a();
virtual Int_t    Cut(Long64_t entry);
virtual Int_t    GetEntry(Long64_t entry);
virtual Long64_t LoadTree(Long64_t entry);
virtual void     Init(TTree *tree);
virtual void     Loop();
virtual Bool_t   Notify();
virtual void     Show(Long64_t entry = -1);
```

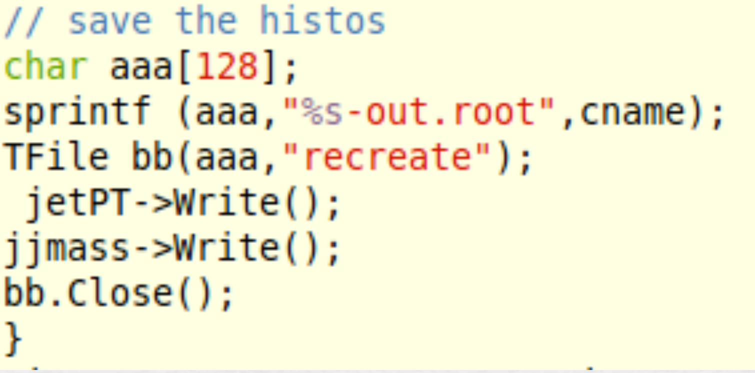


```
   istapp_a(TTree *tree=0, char cname[128]);
   virtual ~istapp_a();
   virtual Int_t    Cut(Long64_t entry);
   virtual Int_t    GetEntry(Long64_t entry);
   virtual Long64_t LoadTree(Long64_t entry);
   virtual void     Init(TTree *tree);
   virtual void     Loop(char cname[128]="generic");
   virtual Bool_t   Notify();
   virtual void     Show(Long64_t entry = -1);
};
```

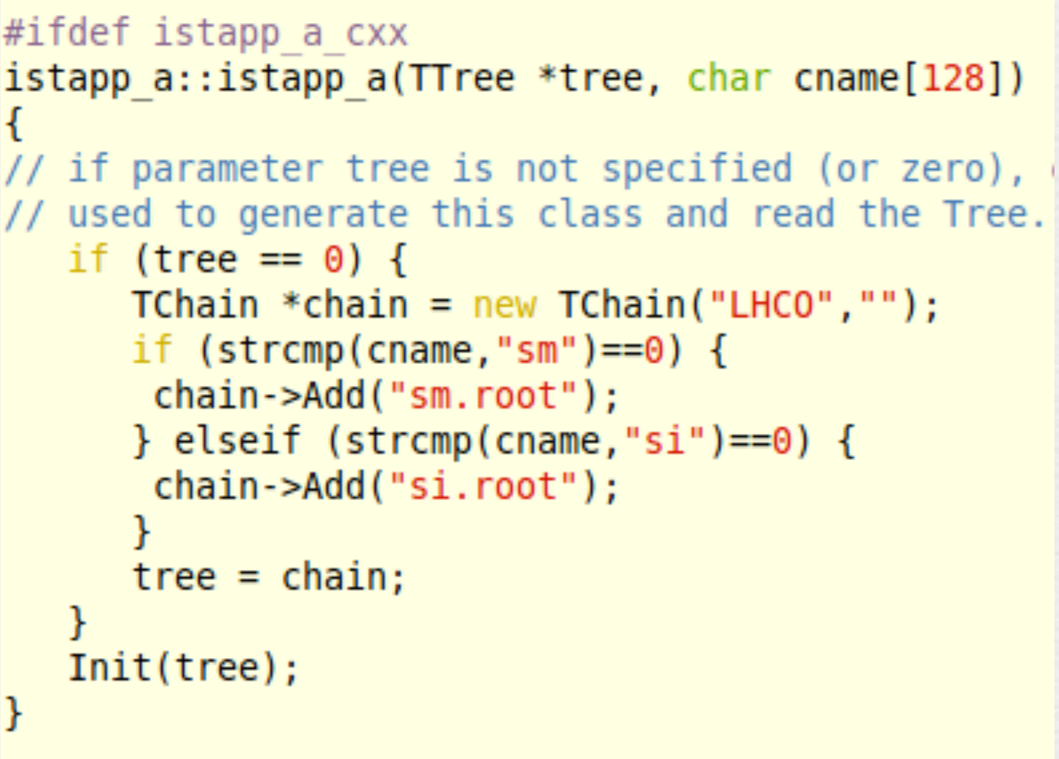
edit the .C file too..



```
void istapp_a::Loop(char cname[128])
{
```

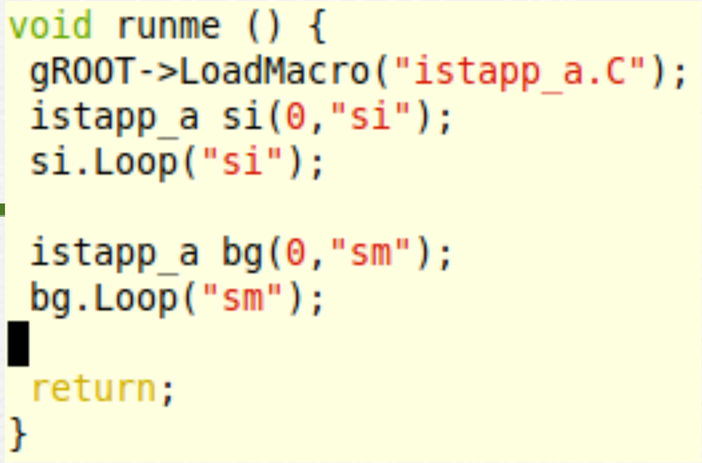


```
// save the histos
char aaa[128];
sprintf (aaa,"%s-out.root",cname);
TFile bb(aaa,"recreate");
 jetPT->Write();
jjmass->Write();
bb.Close();
}
```



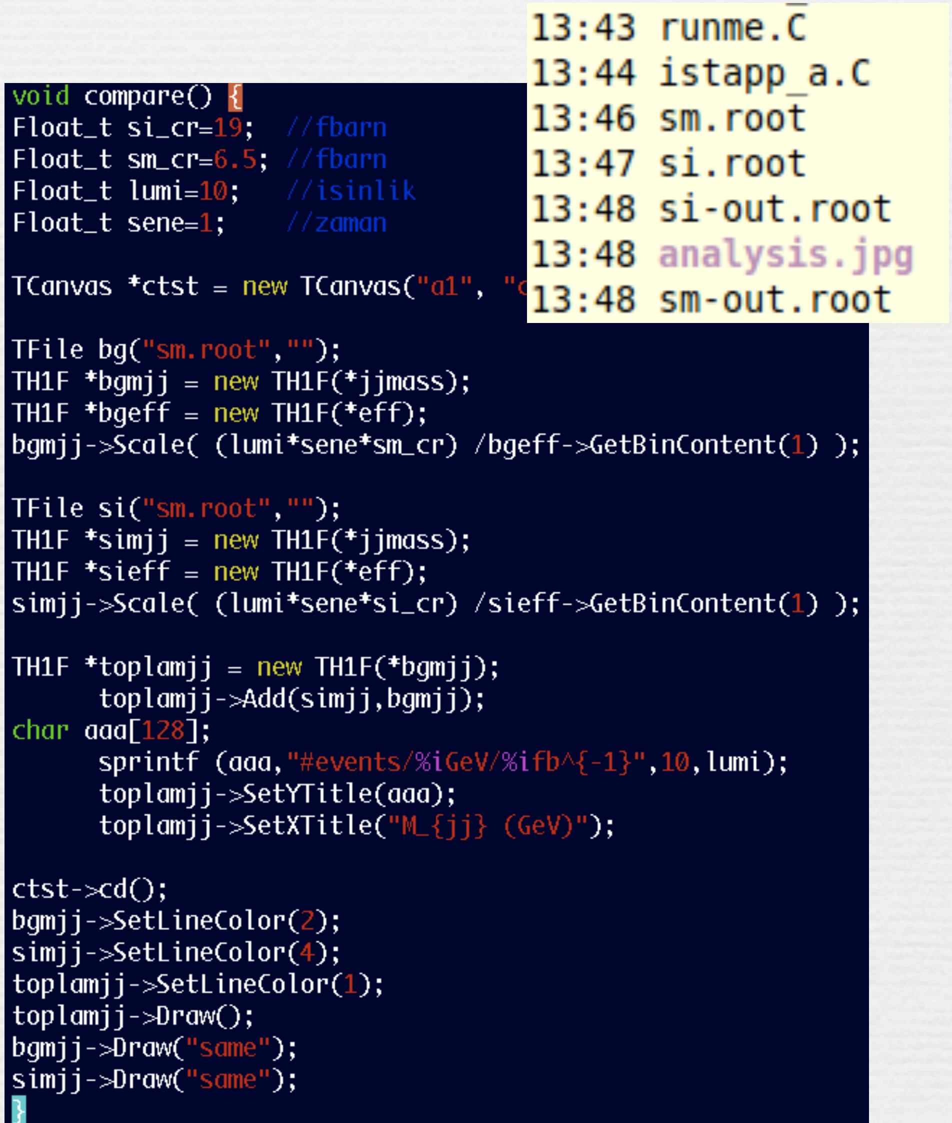
```
#ifdef istapp_a_cxx
istapp_a::istapp_a(TTree *tree, char cname[128])
{
// if parameter tree is not specified (or zero),
// used to generate this class and read the Tree.
   if (tree == 0) {
      TChain *chain = new TChain("LHCO","");
      if (strcmp(cname,"sm")==0) {
       chain->Add("sm.root");
      } elseif (strcmp(cname,"si")==0) {
       chain->Add("si.root");
      }
      tree = chain;
   }
   Init(tree);
}
```

si & sm - comparison



```
void runme () {
 gROOT->LoadMacro("istapp_a.C");
 istapp_a si(0,"si");
 si.Loop("si");

 istapp_a bg(0,"sm");
 bg.Loop("sm");

 return;
}
```

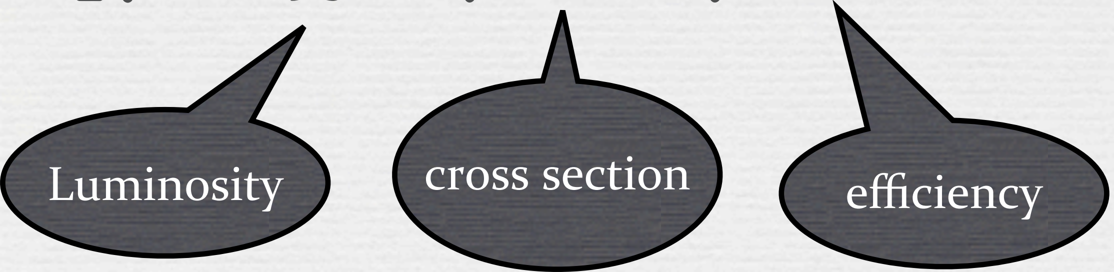
- runme.C should also change
- Lets run, results:
  - Now we can compare the signal and background events.
- Write a C program to do that:
  - it reads si & sm files
  - it calculates the yearly event yield according to x-sections
  - it tells us the number of events if we had made such an experiment



```
13:43 runme.C
13:44 istapp_a.C
13:46 sm.root
13:47 si.root
13:48 si-out.root
13:48 analysis.jpg
13:48 sm-out.root
```

```
void compare() {
Float_t si_cr=19;   //fbarn
Float_t sm_cr=6.5;  //fbarn
Float_t lumi=10;    //isinlik
Float_t sene=1;     //zaman

TCanvas *ctst = new TCanvas("a1", "c

TFile bg("sm.root","");
TH1F *bgmjj = new TH1F(*jjmass);
TH1F *bgeff = new TH1F(*eff);
bgmjj->Scale( (lumi*sene*sm_cr) /bgeff->GetBinContent(1) );

TFile si("sm.root","");
TH1F *simjj = new TH1F(*jjmass);
TH1F *sieff = new TH1F(*eff);
simjj->Scale( (lumi*sene*si_cr) /sieff->GetBinContent(1) );

TH1F *toplamjj = new TH1F(*bgmjj);
      toplamjj->Add(simjj,bgmjj);
char aaa[128];
      sprintf (aaa,"#events/%iGeV/%ifb^{-1}",10,lumi);
      toplamjj->SetYTitle(aaa);
      toplamjj->SetXTitle("M_{jj} (GeV)");

ctst->cd();
bgmjj->SetLineColor(2);
simjj->SetLineColor(4);
toplamjj->SetLineColor(1);
toplamjj->Draw();
bgmjj->Draw("same");
simjj->Draw("same");
}
```

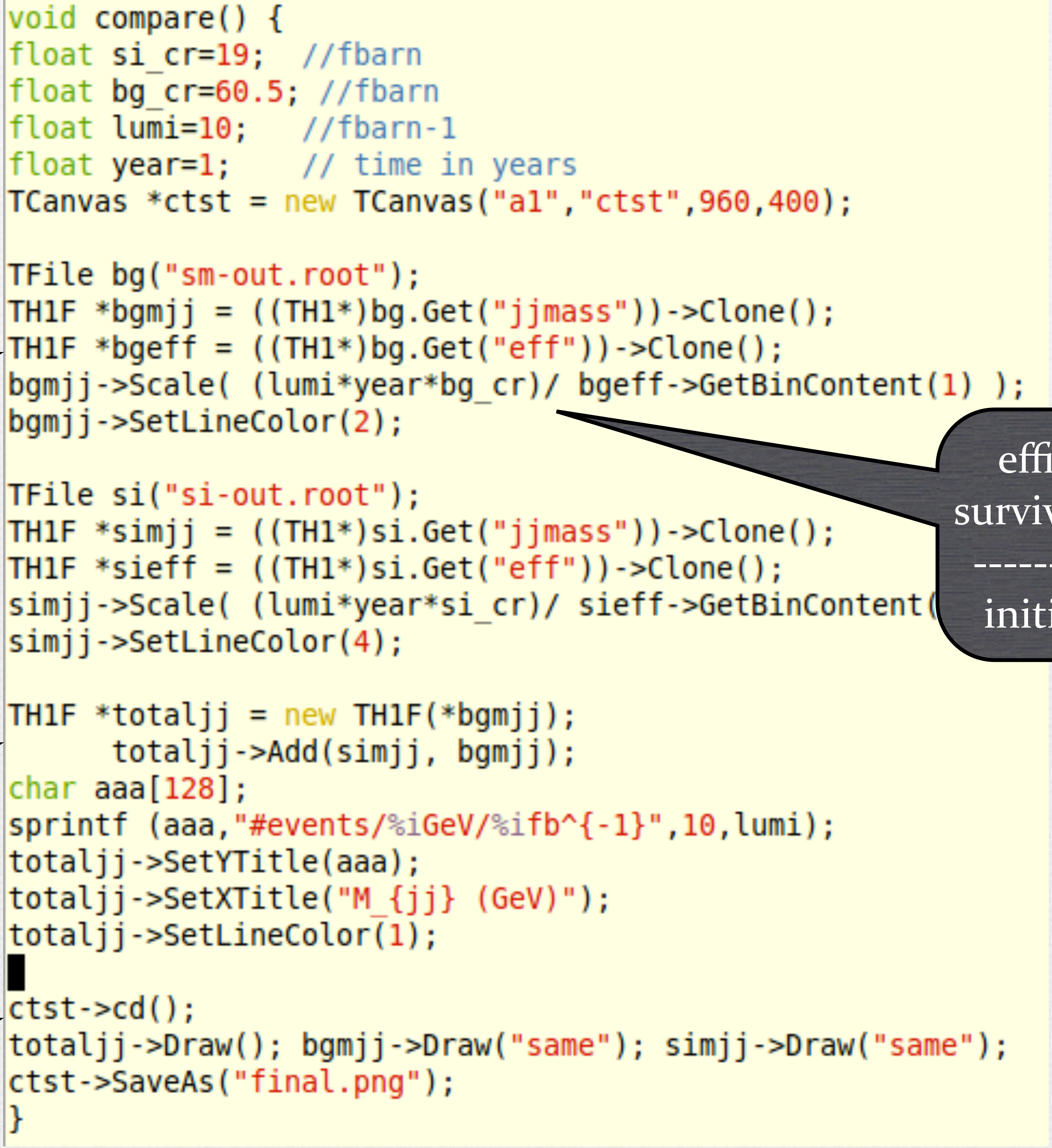
$$N = \mathcal{L} \times \sigma \times \varepsilon$$



Luminosity

cross section

efficiency



```
void compare() {
float si_cr=19;   //fbarn
float bg_cr=60.5; //fbarn
float lumi=10;    //fbarn-1
float year=1;     // time in years
TCanvas *ctst = new TCanvas("a1","ctst",960,400);

TFile bg("sm-out.root");
TH1F *bgmjj = ((TH1*)bg.Get("jjmass"))->Clone();
TH1F *bgeff = ((TH1*)bg.Get("eff"))->Clone();
bgmjj->Scale( (lumi*year*bg_cr)/ bgeff->GetBinContent(1) );
bgmjj->SetLineColor(2);

TFile si("si-out.root");
TH1F *simjj = ((TH1*)si.Get("jjmass"))->Clone();
TH1F *sieff = ((TH1*)si.Get("eff"))->Clone();
simjj->Scale( (lumi*year*si_cr)/ sieff->GetBinContent(
simjj->SetLineColor(4);

TH1F *totaljj = new TH1F(*bgmjj);
      totaljj->Add(simjj, bgmjj);
char aaa[128];
sprintf (aaa,"#events/%iGeV/%ifb^{-1}",10,lumi);
totaljj->SetYTitle(aaa);
totaljj->SetXTitle("M_{jj} (GeV)");
totaljj->SetLineColor(1);

ctst->cd();
totaljj->Draw(); bgmjj->Draw("same"); simjj->Draw("same");
ctst->SaveAs("final.png");
}
```

definitions

window

yearly event yield

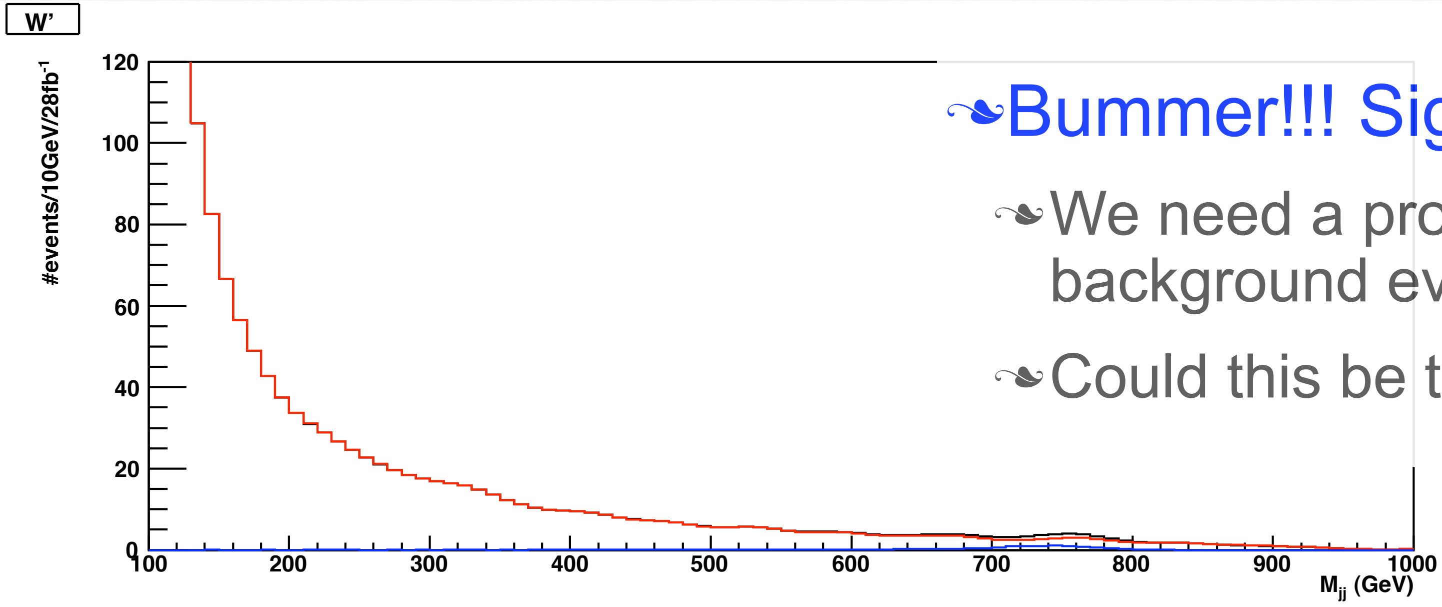
efficiency = surviving events / initial events

$$\text{efficiency} = \frac{\text{surviving events}}{\text{initial events}}$$

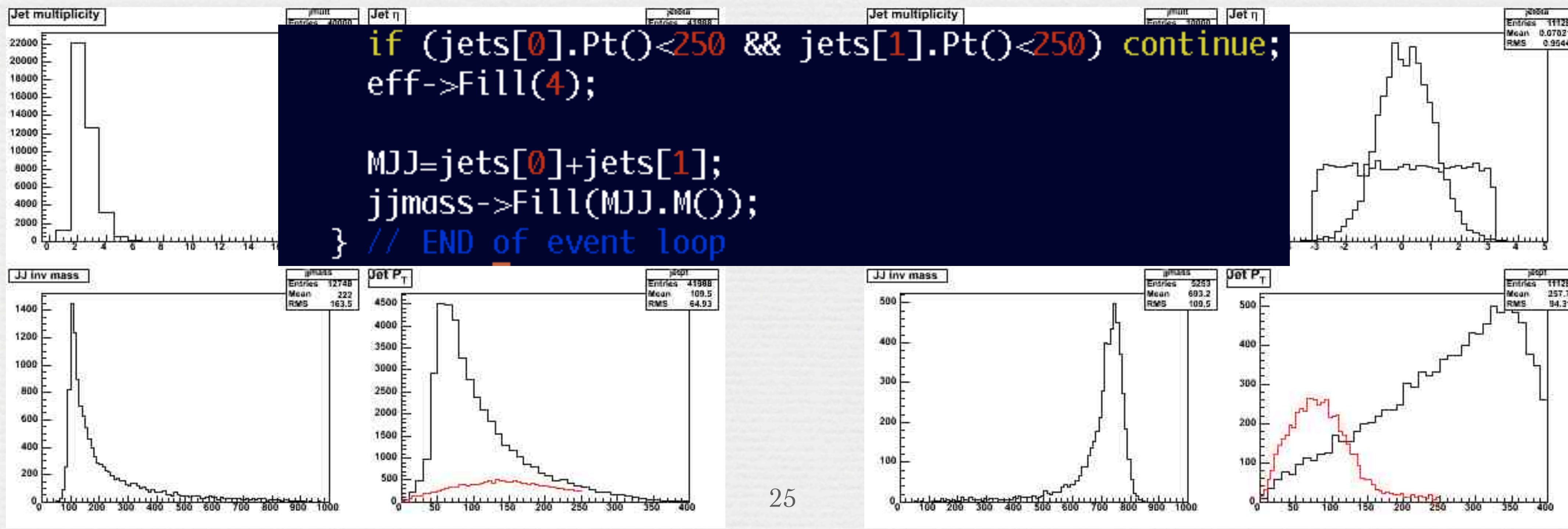
total event yield

drawing si + sm

Merging si and bg events



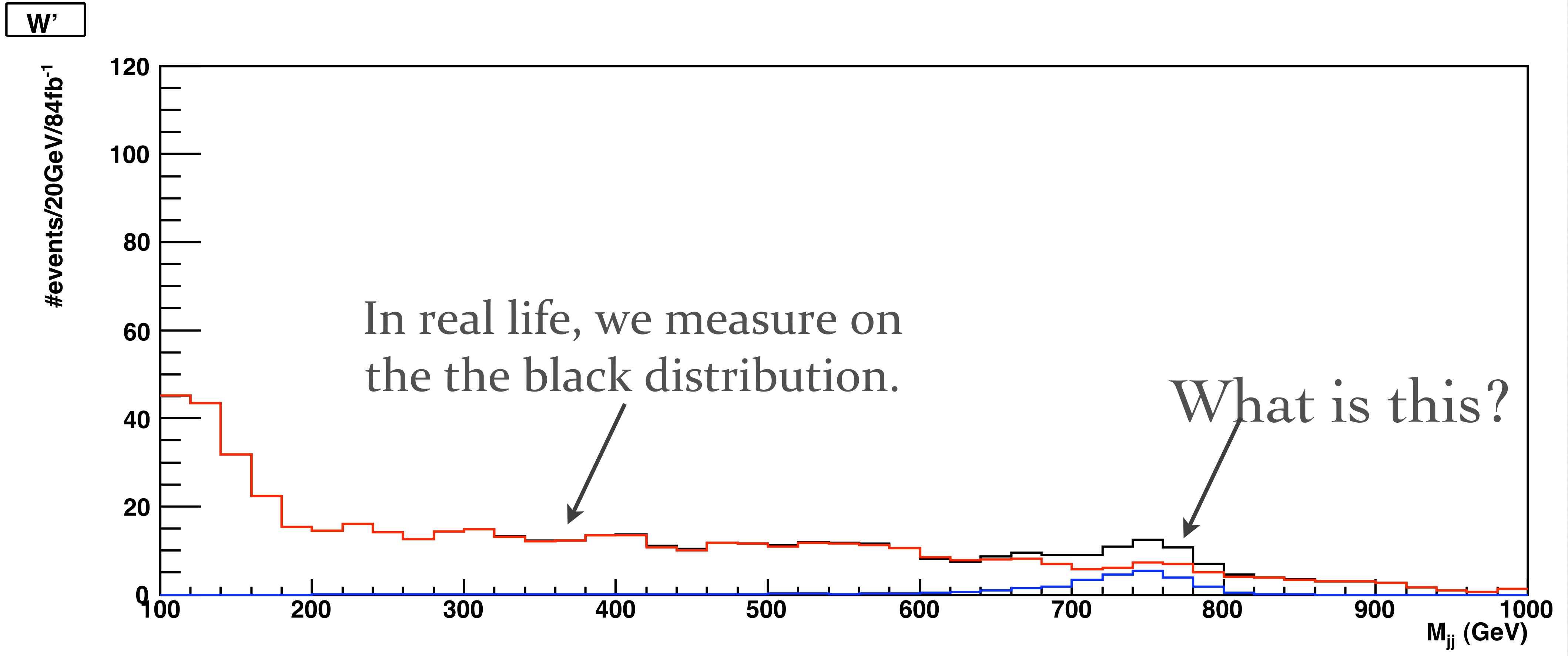
- Bummer!!! Signal not seen...
  - We need a property to eliminate the background events to see the signal.
  - Could this be the jet pT ?



```
    if (jets[0].Pt()<250 && jets[1].Pt()<250) continue;
    eff->Fill(4);

    MJJ=jets[0]+jets[1];
    jjmass->Fill(MJJ.M());
  } // END of event loop
```

After the cuts...

- Fewer events remain after cuts:
  - Lets “run” for 3 years & count the events in 20GeV bins.



In real life, we measure on the the black distribution.

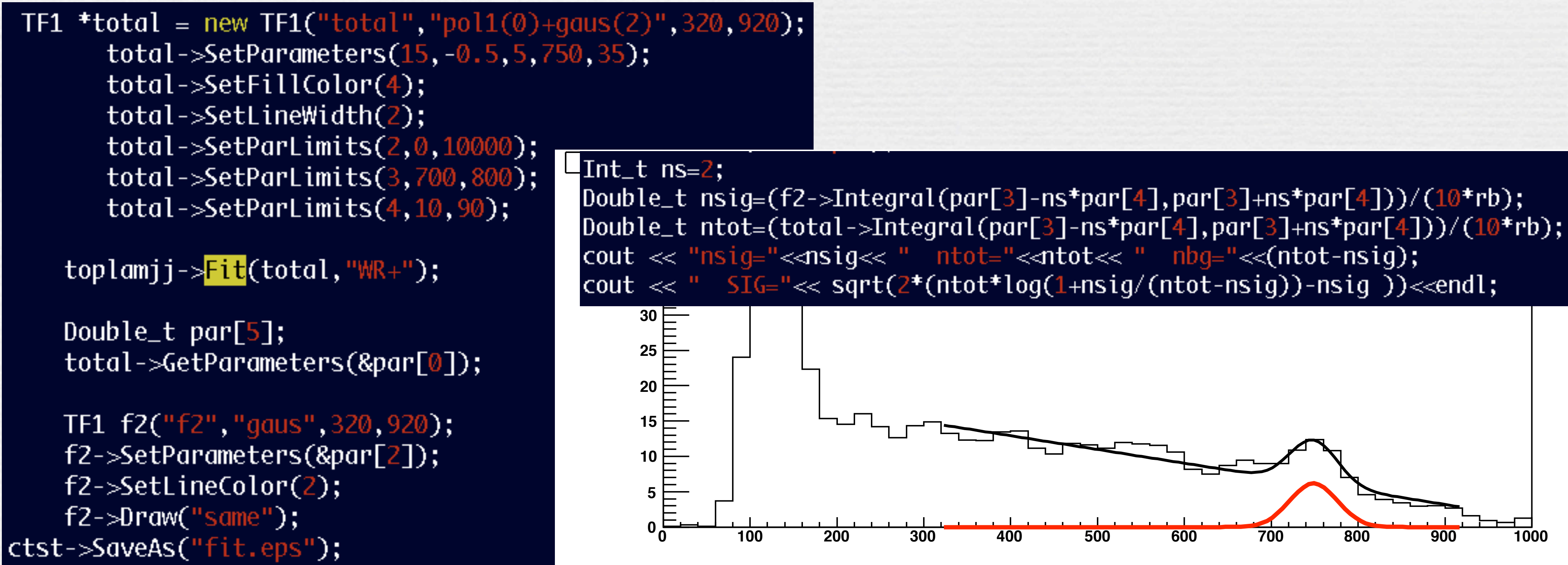
What is this?

Is the effect real ?

- Meaningful or statistical fluctuation?
  - Significance estimators (σ) →
  - if significance = 3σ => Observation
  - if significance 5σ => Discovery

$$\left\{\begin{array}{l} S/\sqrt{B} \\ S/\sqrt{S+B} \\ 2\times(\sqrt{S+B}-\sqrt{B}) \\ \sqrt{2\times\left[(s+b)\ln\left(1+\frac{s}{b}\right)-s\right]} \end{array}\right.$$

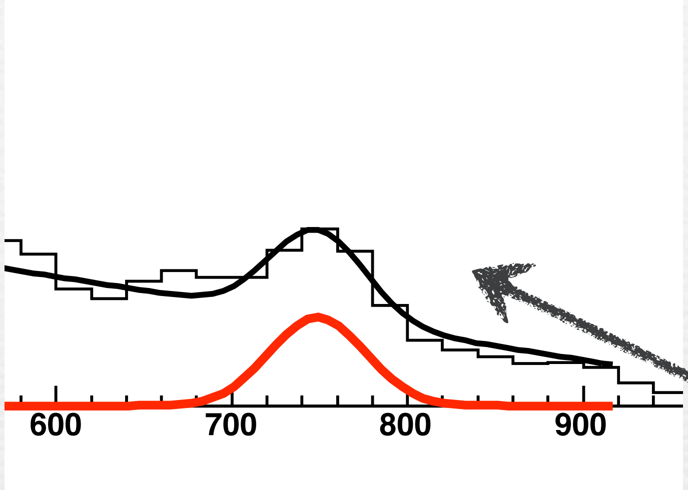
- Lets fit a poly1+Gauss to the total histogram to estimate signal and background event counts.

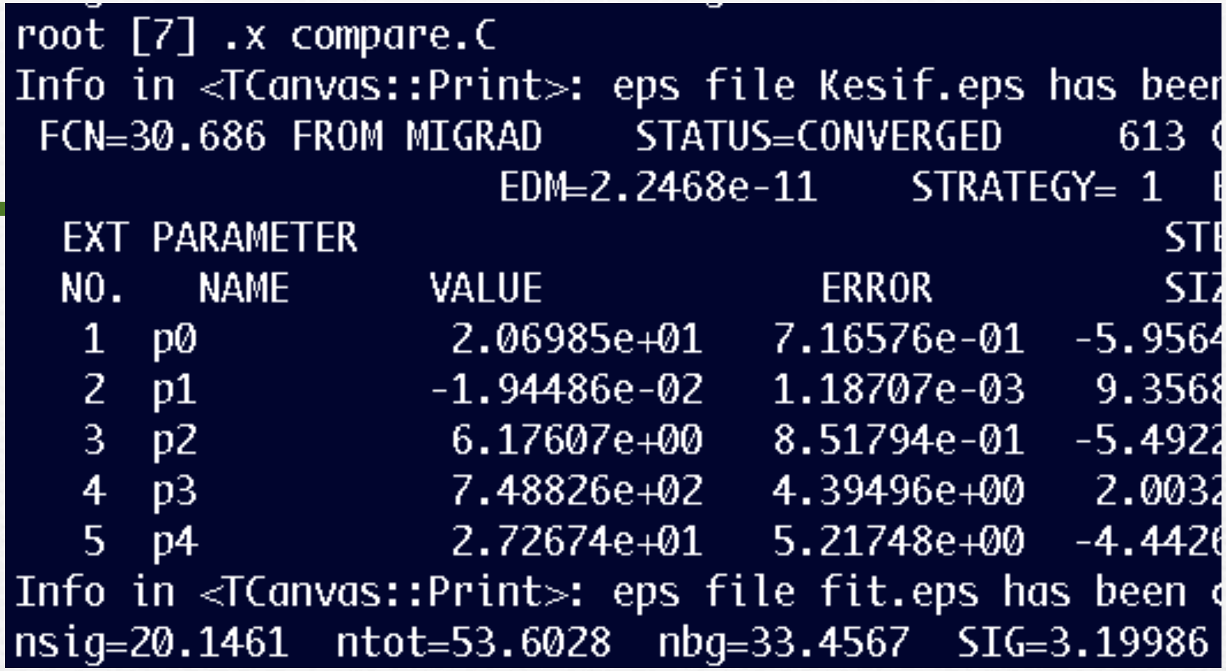


```
 TF1 *total = new TF1("total","pol1(0)+gaus(2)",320,920);
        total->SetParameters(15,-0.5,5,750,35);
        total->SetFillColor(4);
        total->SetLineWidth(2);
        total->SetParLimits(2,0,10000);
        total->SetParLimits(3,700,800);
        total->SetParLimits(4,10,90);

    toplamjj->Fit(total,"WR+");

    Double_t par[5];
    total->GetParameters(&par[0]);

    TF1 f2("f2","gaus",320,920);
    f2->SetParameters(&par[2]);
    f2->SetLineColor(2);
    f2->Draw("same");
ctst->SaveAs("fit.eps");
```

```
Int_t ns=2;
Double_t nsig=(f2->Integral(par[3]-ns*par[4],par[3]+ns*par[4]))/(10*rb);
Double_t ntot=(total->Integral(par[3]-ns*par[4],par[3]+ns*par[4]))/(10*rb);
cout << "nsig="<<nsig<< "  ntot="<<ntot<< "  nbg="<<(ntot-nsig);
cout << "  SIG="<< sqrt(2*(ntot*log(1+nsig/(ntot-nsig))-nsig ))<<endl;
```

Test it..

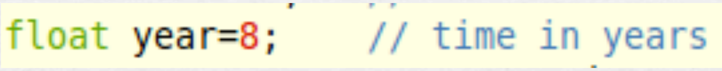




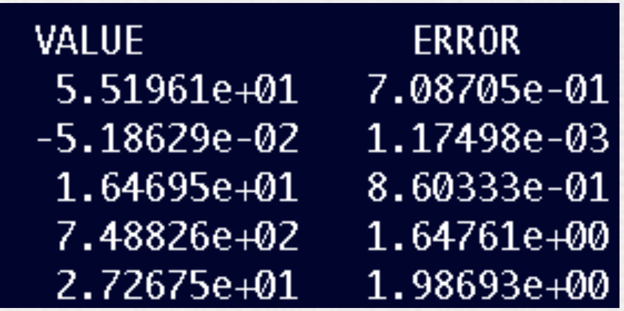
```
root [7] .x compare.C
Info in <TCanvas::Print>: eps file Kesif.eps has been
 FCN=30.686 FROM MIGRAD     STATUS=CONVERGED      613 C
                    EDM=2.2468e-11     STRATEGY= 1   E
  EXT PARAMETER                                    STE
  NO.   NAME        VALUE          ERROR          SIZ
   1  p0         2.06985e+01   7.16576e-01  -5.9564
   2  p1        -1.94486e-02   1.18707e-03   9.3568
   3  p2         6.17607e+00   8.51794e-01  -5.4922
   4  p3         7.48826e+02   4.39496e+00   2.0032
   5  p4         2.72674e+01   5.21748e+00  -4.4426
Info in <TCanvas::Print>: eps file fit.eps has been c
nsig=20.1461  ntot=53.6028  nbg=33.4567  SIG=3.19986
```

A “signal” is observed

- For a “discovery” we need more events.
- Lets “run” for a few more years:

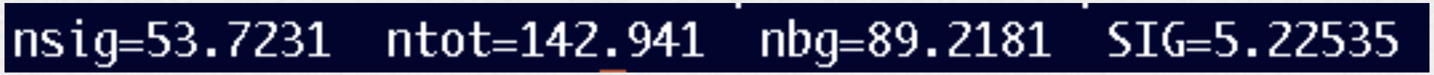


```
float year=8;    // time in years
```



```
VALUE           ERROR
 5.51961e+01    7.08705e-01
-5.18629e-02    1.17498e-03
 1.64695e+01    8.60333e-01
 7.48826e+02    1.64761e+00
 2.72675e+01    1.98693e+00
```

DISCOVERY!!!



```
nsig=53.7231  ntot=142.941  nbg=89.2181  SIG=5.22535
```

Conclusion

- Now you can eat by yourselves
  - keep eating until you find your own style!!
  - LHC, SLHC, ILC, LHeC, CLIC.. await you.
  - SuSY, GUT, ED, LH.. models await you.
- Work hard and be successful.