TOYEE TREE

Mass distribution

- Draw the mass distribution
- Fit by gaus/expo/gaus+expo
- Change line color and fill color of the picture
- Add legends

Draw the mass distribution

```
double mass_mu = 105.658E-3;
double mu1_px = mu1_pt *TMath::Cos(mu1_phi);
double mu1_py = mu1_pt *TMath::Sin(mu1_phi);
double mu1_pz = mu1_pt *TMath::SinH(mu1_eta);
double mu2_px = mu2_pt *TMath::Cos(mu2_phi);
double mu2_py = mu2_pt *TMath::Sin(mu2_phi);
double mu2_pz = mu2_pt *TMath::SinH(mu2_eta);

double E1 = sqrt(TMath::Sq(mass_mu)+TMath::Sq(mu1_px)+TMath::Sq(mu1_py)+TMath::Sq(mu1_pz));
double E2 = sqrt(TMath::Sq(mass_mu)+TMath::Sq(mu2_px)+TMath::Sq(mu2_py)+TMath::Sq(mu2_pz));

double mass_Q = sqrt(TMath::Sq(E1+E2)-TMath::Sq(mu1_px+mu2_px)-TMath::Sq(mu1_py+mu2_py)-TMath::Sq(mu1_pz+mu2_pz));
```

Fit by gaus/expo/gaus+expo

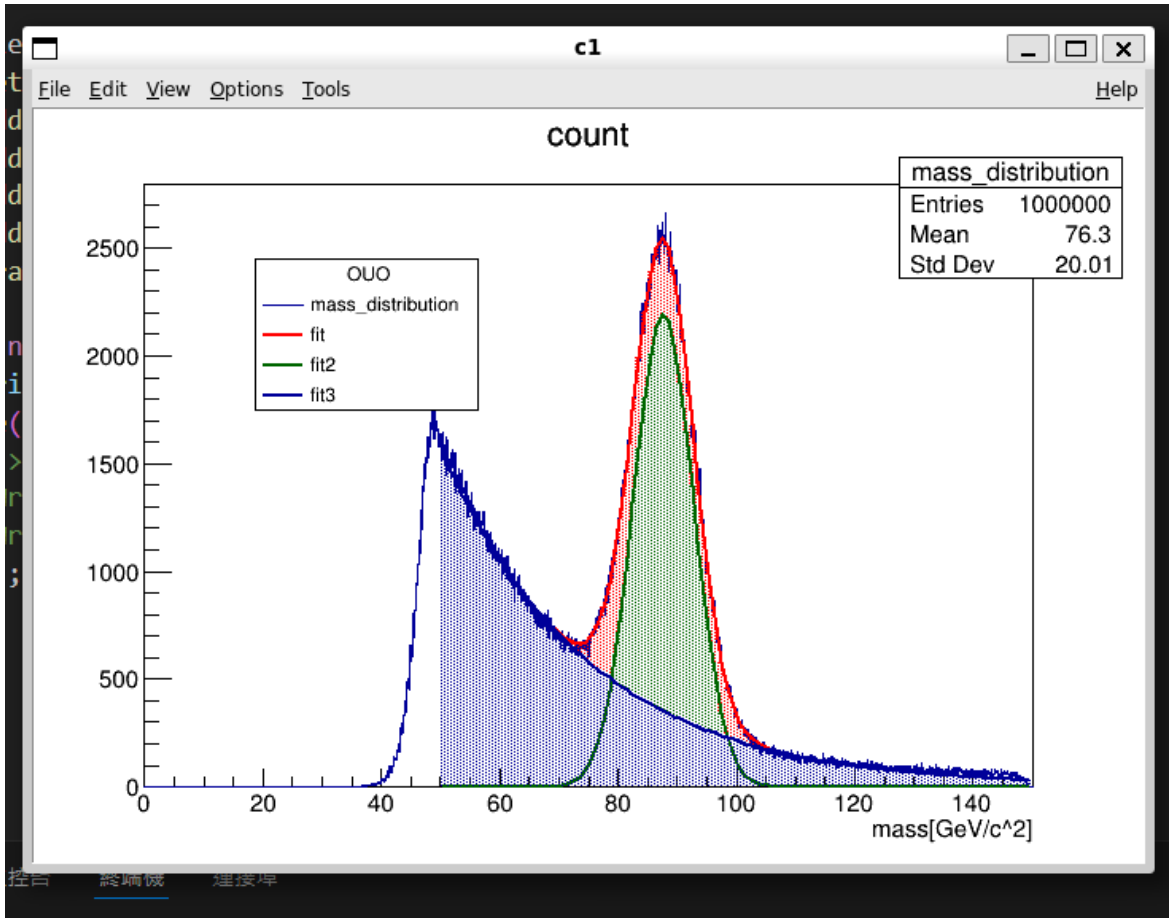
```
TF1*fit = new TF1("fit","[0]*exp(-0.5*((x-[1])/[2])^2)+exp([3]+[4]*x)",50,150);
fit->SetParameter(0,0.5);//從0.5開始跑計算
fit->SetParameter(1,0);
fit->SetParameter(2,0.45);
fit->SetParameter(3,20);
fit->SetParameter(4,-0.0901);

TF1*fit2 = new TF1("fit2","[0]*exp(-0.5*((x-[1])/[2])^2)",50,150);//[1]是平均值,[2]是標準差

TF1*fit3 = new TF1("fit3","exp([3]+[4]*x)",50,150);

TH1F*mass_distribution = (TH1F*)input->Get("mass_distribution");
mass_distribution->Fit(fit,"R");
fit2->SetParameter(0,fit->GetParameter(0));
fit2->SetParameter(1,fit->GetParameter(1));
fit2->SetParameter(2,fit->GetParameter(2));
fit3->SetParameter(3,fit->GetParameter(3));
fit3->SetParameter(4,fit->GetParameter(4));
```

Change line color and fill color of the picture

```
fit->SetFillStyle(3002);
fit->SetFillColor(kRed);
fit2->SetLineColor(kGreen+3);
fit2->SetFillStyle(3002);
fit2->SetFillColor(kGreen+3);
fit3->SetLineColor(kBlue+2);
fit3->SetFillStyle(3002);
fit3->SetFillColor(kBlue+2);
```

Add legends

```
TLegend* legend = new TLegend(0.2,0.6,0.4,0.8);//按照圖的比例設定legend的左下右上
legend->SetHeader("OUO","C"); // option "C" allows to center the header
legend->AddEntry(mass_distribution,"mass_distribution","l");
legend->AddEntry(fit,"fit","l");
legend->AddEntry(fit2,"fit2","l");
legend->AddEntry(fit3,"fit3","l");
legend->Draw("SAME");
```


c1
File Edit View Options Tools Help
count
mass_distribution
Entries 1000000
Mean 76.3
Std Dev 20.01
OUO
mass_distribution
fit
fit2
fit3
2500
2000
1500
1000
500
0
0 20 40 60 80 100 120 140
mass[GeV/c^2]

pT distribution method-1

- Tell sideband region from signal region by standard deviation
- Draw pT distribution of them
- Scale the sideband region
- Get pT distribution of the signal

Tell sideband region from signal region by SD

```
double mean = fit2->GetParameter(1); // [1]是平均值
double sigma = fit2->GetParameter(2); // [2]是標準差
double lowerBound = mean - 3 * sigma;
double upperBound = mean + 3 * sigma;
std::cout << "Mean: " << mean << std::endl;
std::cout << "Sigma: " << sigma << std::endl;
std::cout << "Range: [" << lowerBound << ", " << upperBound << "]" << std::endl;
```

Draw pT distribution of them

```
sideband = new TH1F("sideband", "count;pT[GeV/c]", 150, 0, 150);
signal_region = new TH1F("signal_region", "count;pT[GeV/c]", 150, 0, 150);
```

```
double pT = sqrt(TMath::Sq(mu1_px+mu2_px)+TMath::Sq(mu1_py+mu2_py));


if (mass_Q >= 50 && mass_Q < 72)
{
sideband->Fill(pT);
i=i+1;
}
else if (mass_Q > 72 && mass_Q < 103)
{
signal_region->Fill(pT);
j=j+1;
}
else if (mass_Q >= 103 && mass_Q < 150)
{
sideband->Fill(pT);
k=k+1;
}
```

Scale the sideband region

```
double a1 = fit3->Integral(50, 72)*(2000/150);
double a2 = fit3->Integral(72, 102)*(2000/150);
double a3 = fit3->Integral(102, 150)*(2000/150);
double a4 = fit2->Integral(72, 102)*(2000/150);
double alpha = a2/(a1+a3) ;
sideband->Scale(alpha);
```

Get pT distribution of the signal

```
signal_region->Add(sideband,-1);
```