void Week13Merged100(){

TFile \*FF = new TFile("TeLoadedTe130\_2n2b\_TeDiol\_rat-6.1.2\_merged100.ntuple.root");

TTree \*output1 = (TTree\*)FF->Get("output");

const Double\_t pi = TMath::Pi();

cout << "pi = " << pi << endl;

float energymid[5] = {0.25,0.75,1.25,1.75,2.25};

float energyerror[5] = {0.25,0.25,0.25,0.25,0.25};

float energylow[5];

float energyhi[5];

for(int i=0;i<5;i++){

energylow[i] = energymid[i] - energyerror[i];

energyhi[i] = energymid[i] + energyerror[i];

}

float volume = 696909970300; //mm cubed =696.91m cubed

float shells = 5;

float VPS = volume/shells;

cout << "VPS= " << VPS << endl;

float volumehi[5];

float volumelow[5];

float volumemid[5];

float volumeerror[5];

float rangehi[5];

float rangelow[5];

float rangemid[5];

float rangeerror[5];

for(int i=0;i<5;i++){

volumehi[i] = ((i+1)\*VPS);

cout << "volumehi " << i << " = " << volumehi[i] << endl;

volumelow[i] = ((i)\*VPS);

cout << "volumelow " << i << " = " << volumelow[i] << endl;

//volumemid[i] = ((i+0.5)\*VPS);

//cout << "volumemid " << i << " = " << volumemid[i] << endl;

//volumeerror[i] = (volumehi[i] - volumelow[i])/2;

rangehi[i] = pow(((3\*volumehi[i])/(4\*pi)),1./3.);

cout << "rangehi " << i << " = " << rangehi[i] << endl;

rangelow[i] = pow(((3\*volumelow[i])/(4\*pi)),1./3.);

cout << "rangelow " << i << " = " << rangelow[i] << endl;

rangemid[i] = (rangehi[i] + rangelow[i])/2.;

cout << "new range mid " << i << " = " << rangemid[i] << endl;

rangeerror[i] = (rangehi[i]-rangelow[i])/2.;

cout << "new rangeerror " << i << " = " << rangeerror[i] << endl;

//rangemid[i] = pow(((3\*volumemid[i])/(4\*pi)),1./3.);

//cout << "rangemid " << i << " = " << rangemid[i] << endl;

//rangeerror[i] = (rangehi[i] - rangelow[i])/2.;

//cout << "rangeerror " << i << " = " << rangeerror[i] << endl;

}

//float rangeval[5] = {2552.873858,3681.881225,4365.352893,4883.47201,5310.191615}; //need to get this calculation automatically, now calculated automatically above.

//float rangeerror = 663.545654; //same here

TH1F \*hist[5][5];

TF1 \*fithist[5][5];

TCanvas \*canvas[5][5];

TGraphErrors \*MeanVsEnergy[5];

TGraphErrors \*SigmaVsEnergy[5];

TGraphErrors \*ChisquareVsEnergy[5];

TGraphErrors \*MeanVsRange[5];

TGraphErrors \*SigmaVsEnergy[5];

TGraphErrors \*ChisquareVsRange[5];

TGraphErrors \*SigmaVsRange[5];

float mean[5][5];

float meanerror[5][5];

float sigma[5][5];

float sigmaerror[5][5];

float chisquare[5][5];

for(int i=0;i<5;i++){

for (int j=0;j<5;j++){

hist[i][j] = new TH1F(Form("hist%d%d",i,j),"",100,-0.4,0.4);

output1->Draw(Form("energy-mcEdep>>hist%d%d",i,j),Form("fitValid==1&&mcEdep>%f&&mcEdep<%f&&mcPosr>%f&&mcPosr<%f",energylow[i],energyhi[i],rangelow[j],rangehi[j]));

fithist[i][j] = new TF1(Form("fithist%d%d",i,j),"gaus",-0.4,0.4);

hist[i][j]->Fit(Form("fithist%d%d",i,j),"R");

mean[i][j] = fithist[i][j] -> GetParameter("Mean");

meanerror[i][j] = fithist[i][j]->GetParError(1);

sigma[i][j] = fithist[i][j]->GetParameter("Sigma");

sigmaerror[i][j] = fithist[i][j]->GetParError(2);

chisquare[i][j] = fithist[i][j] ->GetChisquare();

canvas[i][j] = new TCanvas(Form("canvas%d%d",i,j),"",800,800); //TCanvas \* canvas = new TCanvas(Form("canvas%d%d",i,j),"", 800, 800); old code

canvas[i][j]->SaveAs(Form("canvasMerged100%d%d.png",i,j));

}

TCanvas \*MeanVSRcanvMerged100 = new TCanvas("MeanVSRcanvMerged100","",800,800);

MeanVsRange[i] = new TGraphErrors(5,rangemid,mean[i],rangeerror,meanerror[i]); //unsure if MeanVsEnergy[i], [j] or [5]

MeanVsRange[i]->Draw("APL");

MeanVsRange[i]->SetMarkerStyle(3);

MeanVsRange[i]->SetTitle(Form("Mean against range values: Energy cut %d",i));

MeanVsRange[i]->GetXaxis()->SetTitle("range values (mm)");

MeanVsRange[i]->GetYaxis()->SetTitle("Mean");

MeanVsRange[i]->SetName(Form("RangeVsMean: Energy cut %d",i));

MeanVsRange[i]->Draw();

MeanVSRcanvMerged100->SaveAs(Form("MeanVSRcanvMerged100%d.png",i));

TCanvas\* SigmaVSRcanvMerged100 = new TCanvas("SigmaVSRcanvMerged100","",800,800);

SigmaVsRange[i] = new TGraphErrors(5,rangemid,sigma[i],rangeerror,sigmaerror[i]);

SigmaVsRange[i]->Draw("APL");

SigmaVsRange[i]->SetMarkerStyle(3);

SigmaVsRange[i]->SetTitle(Form("Sigma against range values: Energy cut %d",i));

SigmaVsRange[i]->GetXaxis()->SetTitle("Range values (mm)");

SigmaVsRange[i]->GetYaxis()->SetTitle("Sigma");

SigmaVsRange[i]->SetName(Form("RangeVSigma: Energy cut %d",i));

SigmaVsRange[i]->Draw();

SigmaVSRcanvMerged100->SaveAs(Form("SigmaVSRcanvMerged100%d.png",i));

TCanvas\* ChisquareVSRcanvMerged100 = new TCanvas("ChisquareVSRcanvMerged100","",800,800);

ChisquareVsRange[i] = new TGraphErrors(5,rangemid,chisquare[i],rangeerror,0);

ChisquareVsRange[i]->Draw("APL");

ChisquareVsRange[i]->SetMarkerStyle(3);

ChisquareVsRange[i]->SetTitle(Form("Chisquare against range values: Energy cut %d",i));

ChisquareVsRange[i]->GetXaxis()->SetTitle("Range values (mm)");

ChisquareVsRange[i]->GetYaxis()->SetTitle("Chisquare");

ChisquareVsRange[i]->SetName(Form("RangeVChisquare: energy cut %d",i));

ChisquareVsRange[i]->Draw();

ChisquareVSRcanvMerged100->SaveAs(Form("ChisquareVSRcanvMerged100%d.png",i));

}

float mean\_RE[5][5];

float meanerror\_RE[5][5];

float sigma\_RE[5][5];

float sigmaerror\_RE[5][5];

float chisquare\_RE[5][5];

for(int i=0;i<5;++i){

for(int j=0;j<5;++j){

mean\_RE[i][j] = mean[j][i];

meanerror\_RE[i][j] = meanerror[j][i];

sigma\_RE[i][j] = sigma[j][i];

sigmaerror\_RE[i][j] = sigmaerror[j][i];

chisquare\_RE[i][j] = chisquare[j][i];

}

}

for(int i=0;i<5;i++){

TCanvas\* MeanVSEcanvMerged100 = new TCanvas("MeanVSEcanvMerged100","",800,800);

MeanVsEnergy[i] = new TGraphErrors(5,energymid,mean\_RE[i],energyerror,meanerror\_RE[i]); //unsure if MeanVsEnergy[i], [j] or [5]

MeanVsEnergy[i]->Draw("APL");

MeanVsEnergy[i]->SetMarkerStyle(3);

MeanVsEnergy[i]->SetTitle(Form("Mean against energy values: Range cut %d",i));

MeanVsEnergy[i]->GetXaxis()->SetTitle("Energy values (eV)");

MeanVsEnergy[i]->GetYaxis()->SetTitle("Mean");

MeanVsEnergy[i]->SetName(Form("EnergyVsMean: Range cut %d",i));