Homework 1.2 : ADC/TDC信号的读取和处理

报告人：阳黎升 学号：1901110113

1.作业要求：

在中子探测器前加入厚度为1cm的薄塑料闪烁体(两端读出)，当有带电粒子穿过有信号，中子和γ不产生信号，假设能量沉积为原带电粒子能量的1/10,时间分辨、能量分辨参数与中子探测器相同。

要求：

- **用1.3的代码生成root文件，读入该root文件。**
- **用tree-makeclass的方法，将上述信息加入到代码，并生成新的root文件。**
- **通过数据分析，验证薄塑料闪烁体可作为带电粒子的veto探测器。**
- **保留原ROOT文件的所有参数。将代码上传到github指定目录上，代码以附件方式贴在页面。**

2.生成treeADC.root及ppac.h类：

首先利用1.3节中代码生成treeADC.root文件：

```
[yangls@pkuser cp1h2]$ root -l
root [0] .x tree.c
```

利用treeADC.root文件生成ppac.h类及接下来利用其添加veto探测器的.C文件：

```
[yangls@pkuser cp1h2]$ root -l treeADC.root
root[0] Attaching file treeADC.root as _file0...
(TFile *) 0x1e59c90
root[1] tree->MakeClass("ppac");
Info in <TTreePlayer::MakeClass>: Files: ppac.h and ppac.C generated from TTree: tree
(int) 0
```

3.在ppac.C文件中为中子探测器添加Veto探测器:

首先在ppac.C文件中定义两组Branch：veto和tof，其中veto相关变量表示通过veto探测器探测的粒子信息，而tof相关变量表示中子探测器的探测信息。需要注意的是各种变量的类型，其中pid变量是int型：

```
    tree->Branch("veto_x",&veto_x,"veto_x/D");
    tree->Branch("veto_e",&veto_e,"veto_e/D");
    tree->Branch("veto_tof",&veto_tof,"veto_tof/D");
    tree->Branch("veto_pid",&veto_pid,"veto_pid/I");
    tree->Branch("veto_tu",&veto_tu,"veto_tu/D");
    tree->Branch("veto_td",&veto_td,"veto_td/D");
    tree->Branch("veto_qu",&veto_qu,"veto_qu/D");
    tree->Branch("veto_qd",&veto_qd,"veto_qd/D");
```

```
    tree->Branch("veto_itu",&veto_itu,"veto_itu/I");
    tree->Branch("veto_itd",&veto_itd,"veto_itd/I");
    tree->Branch("veto_iqu",&veto_iqu,"veto_iqu/I");
    tree->Branch("veto_iqd",&veto_iqd,"veto_iqd/I");

    tree->Branch("tof_x",&tof_x,"tof_x/D");
    tree->Branch("tof_e",&tof_e,"tof_e/D");
    tree->Branch("tof_tof",&tof_tof,"tof_tof/D");
    tree->Branch("tof_pid",&tof_pid,"tof_pid/I");
    tree->Branch("tof_tu",&tof_tu,"tof_tu/D");
    tree->Branch("tof_td",&tof_td,"tof_td/D");
    tree->Branch("tof_qu",&tof_qu,"tof_qu/D");
    tree->Branch("tof_qd",&tof_qd,"tof_qd/D");
    tree->Branch("tof_itu",&tof_itu,"tof_itu/I");
    tree->Branch("tof_itd",&tof_itd,"tof_itd/I");
    tree->Branch("tof_iqu",&tof_iqu,"tof_iqu/I");
    tree->Branch("tof_iqd",&tof_iqd,"tof_iqd/I");
```

由于带电粒子穿过veto时有信号，中子和不产生信号，且在Veto中沉积原带电粒子的能量，则在ppac.C文件中修改pid=2的粒子(即质子)的数据，利用tree.C文件中对pid=2的粒子信息进行设置，其veto信息与ppac.root内记录的信息基本一致，但需要将其能量e和q0设置为之前的$\frac{1}{10}$：

```
        if(pid==2){//proton
            //put the web code about proton
            veto_pid=pid;
            veto_x=x;
            veto_e=e*0.1;//1/10
            veto_tof=72./TMath::Sqrt(e)*(d*0.01);
            //TDC
            veto_tu=veto_tof+(L-x)/Vsc+gr->Gaus(0,TRes/2.35)+tu_off;
            veto_td=veto_tof+(L+x)/Vsc+gr->Gaus(0,TRes/2.35)+td_off;
            veto_tu*=TDCch2ns;
            veto_td*=TDCch2ns;

            Double_t q0;
            q0=e*ADCgain*0.1;//1/10
            q0=q0*gr->Gaus(q0,q0*QRes/2.35);
            veto_qu=q0*TMath::Exp(-(L-x)/Lambda);
            veto_qd=q0*TMath::Exp(-(L+x)/Lambda);
            //ADC
            if(veto_qu<0) veto_qu=0;//no nagatetive values
            if(veto_qd<0) veto_qd=0;
            veto_qu += gr->Gaus(ADCuPed,ADCnoise);
            veto_qd += gr->Gaus(ADCdPed,ADCnoise);

            if(veto_qu<100) veto_tu=TDCoverflow;//threshold of time of tu and td
            if(veto_qd<100) veto_td=TDCoverflow;
            //overflow check
            if(veto_tu>TDCoverflow) veto_tu=TDCoverflow;
            if(veto_td>TDCoverflow) veto_td=TDCoverflow;
            if(veto_qu>ADCoverflow) veto_qu=ADCoverflow;
            if(veto_qd>ADCoverflow) veto_qd=ADCoverflow;
            //digtization
            veto_itu=Int_t(veto_tu);
            veto_itd=Int_t(veto_td);
            veto_iqu=Int_t(veto_qu);
```

```
            veto_iqd=Int_t(veto_qd);

            //from this line wo define tof variate
            tof_x = x;
            tof_e = 0.9*e;
            tof_pid = pid;
            tof_tof = tof;
            tof_tu = tu;
            tof_td = td;
            tof_qu = qu;
            tof_qd = qd;
            //ADC
            if(tof_qu<0) tof_qu=0;//no nagatetive values
            if(tof_qd<0) tof_qd=0;
            tof_qu += gr->Gaus(ADCuPed,ADCnoise);
            tof_qd += gr->Gaus(ADCdPed,ADCnoise);

            if(tof_qu<100) tof_tu=TDCoverflow;//threshold of time of tu and td
            if(tof_qd<100) tof_td=TDCoverflow;
            //overflow check
            if(tof_tu>TDCoverflow) tof_tu=TDCoverflow;
            if(tof_td>TDCoverflow) tof_td=TDCoverflow;
            if(tof_qu>ADCoverflow) tof_qu=ADCoverflow;
            if(tof_qd>ADCoverflow) tof_qd=ADCoverflow;
            //digtization
            tof_itu=Int_t(tof_tu);
            tof_itd=Int_t(tof_td);
            tof_iqu=Int_t(tof_qu);
            tof_iqd=Int_t(tof_qd);
        }
```

对于pid≠2的粒子则无需改变相关信息，但其veto数据必须符合veto探测器无法探测到这些粒子：

```
        else{//gamma and neutron
            veto_x=-300;//make veto do not detect them
            veto_e=-100;
            veto_tof=-100;
            veto_pid=pid;
            //tdc
            veto_tu=TDCoverflow;
            veto_td=TDCoverflow;
            veto_itu=Int_t(veto_tu);
            veto_itd=Int_t(veto_td);
            //adc
            veto_qu=gr->Gaus(ADCdPed,ADCnoise);
            veto_qd=gr->Gaus(ADCdPed,ADCnoise);
            veto_iqu=Int_t(veto_qu);
            veto_iqd=Int_t(veto_qd);
            //from this line wo define tof variate
            tof_x=x;//make veto do not detect them
            tof_e=e;
            tof_tof=tof;
            tof_pid=pid;
            //tdc
            tof_tu=tu;
```

```
            tof_td=td;
            tof_itu=itu;
            tof_itd=itd;
            //adc
            tof_qu=qu;
            tof_qd=qd;
            tof_iqu=iqu;
            tof_iqd=iqd;
        }
```

在此操作后生成ppac.root文件，记录经veto和中子探测器探测生成的数据：

```
[yangls@pkuser cp1h2]$ root -l
root [0] .L ppac.C
root [1] ppac t
(ppac &) @0x7f0664160020
root [2] t.Loop()
```

查看生成的ppac.root文件：

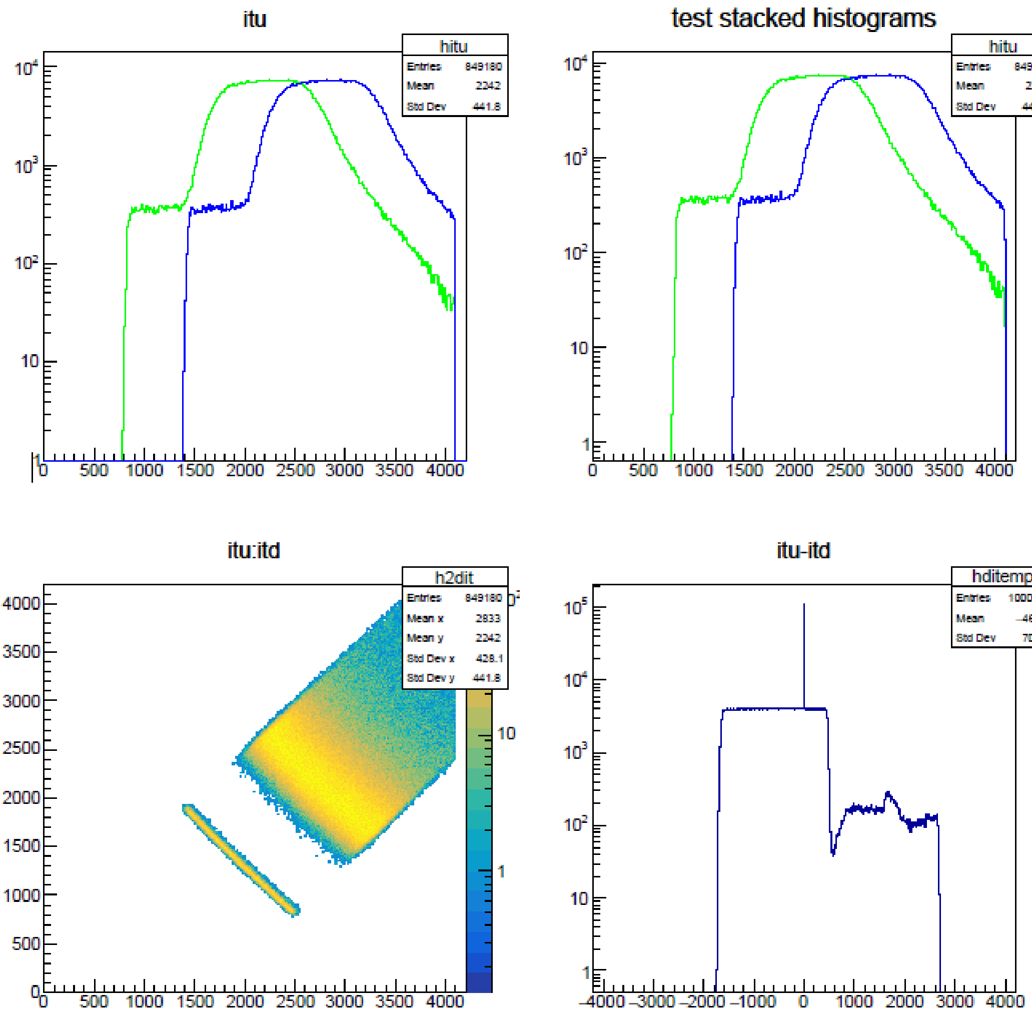
```
[yangls@pkuser cp1h2]$ root -l ppac.root
root [0]
Attaching file ppac.root as _file0...
(TFile *) 0x2494c70
root [1] tree->Print()
******************************************************************************
*Tree    :tree      : ppac                                                   *
*Entries :  1000000 : Total =       152182096 bytes  File  Size = 79233924 *
*        :          : Tree compression factor =   1.92                       *
******************************************************************************
*Br    0 :veto_x    : veto_x/D                                               *
*Entries :  1000000 : Total  Size=    8009338 bytes  File Size  =   938411 *
*Baskets :       97 : Basket Size=    6071808 bytes  Compression=   8.53   *
*............................................................................*
*Br    1 :veto_e    : veto_e/D                                               *
*Entries :  1000000 : Total  Size=    8009338 bytes  File Size  = 1085229 *
*Baskets :       97 : Basket Size=    6071808 bytes  Compression=   7.38   *
*............................................................................*
*Br    2 :veto_tof  : veto_tof/D                                             *
*Entries :  1000000 : Total  Size=    8009540 bytes  File Size  = 1091971 *
*Baskets :       97 : Basket Size=    6072320 bytes  Compression=   7.33   *
*............................................................................*
*Br    3 :veto_pid  : veto_pid/I                                             *
*Entries :  1000000 : Total  Size=    4005067 bytes  File Size  = 456453  *
*Baskets :       50 : Basket Size=    4552704 bytes  Compression=   8.77   *
*............................................................................*
```

通过ppac.root文件生成AnalyzeData.h类及.C文件处理ADC/TDC信号：

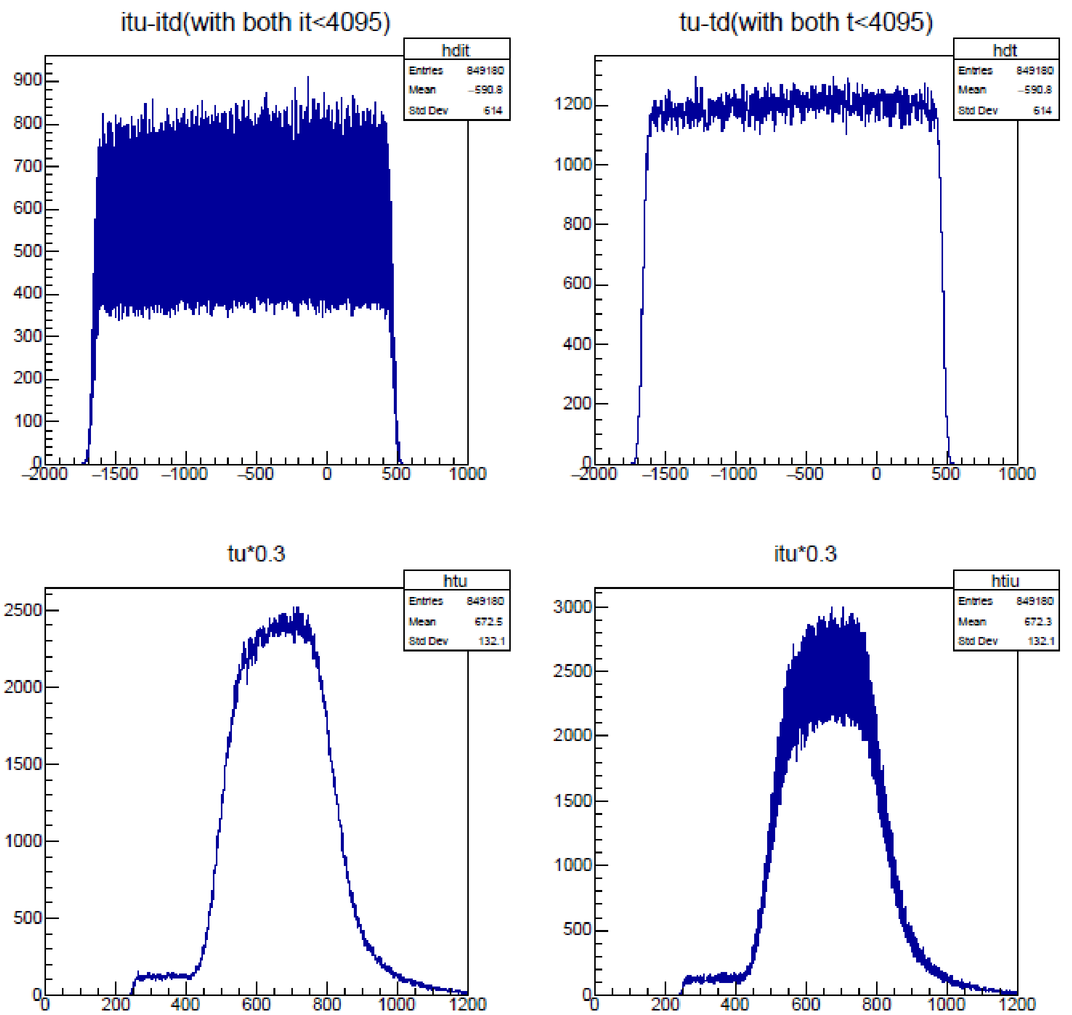
```
root [2] tree->MakeClass("AnalyzeData")
Info in <TTreePlayer::MakeClass>: Files: AnalyzeData.h and AnalyzeData.C
generated from TTree: tree
(int) 0
```

4.利用AnalyzeData.C程序处理ADC/TDC信号:

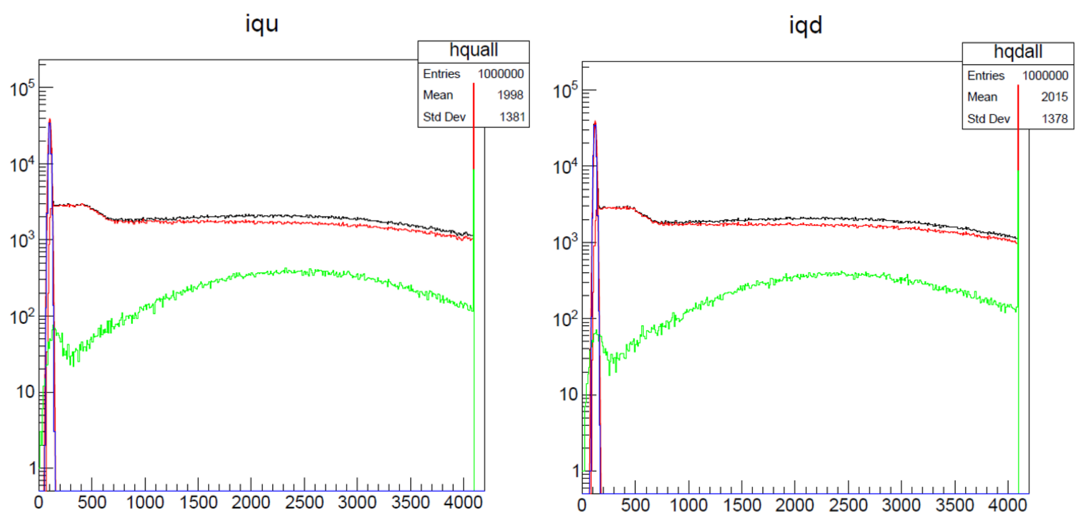
利用网页的办法绘制itu和itd的图像如下图上左所示，上右为尝试使用THStack将多个Histogram叠加到1张图上，下左为二维关联图，将其进行1维投影，如下右所示，与我们的常识不符，原因是未在合理的取值范围的前提下对参数进行运算：



将itu和itd均限制于小于4095道，则会得出下列结果，可以看到，与测量值itu相比，实际值的波动相对较小，原因在于itu实际上是整型数据，图像的分bin会导致某些道多一些某些道少一些的周期性现象。



接下来计算ped峰参数，确定噪声阈值，具体结果如下图所示，其中左图为iqu，右图为iqd：



参数如下

```
FCN=11562.1 FROM MIGRAD    STATUS=CONVERGED     119 CALLS     120 TOTAL
EDM=4.8011e-07    STRATEGY= 1      ERROR MATRIX ACCURATE
 EXT PARAMETER                                  STEP         FIRST
 NO.   NAME      VALUE            ERROR          SIZE       DERIVATIVE
  1  Constant     3.85194e+04   1.50772e+02   6.12022e+00   2.00215e-06
  2  Mean         1.01173e+02   3.61063e-02   1.79664e-03  -2.63510e-02
  3  Sigma        1.13056e+01   2.80454e-02   2.51211e-05   2.98256e-01
FCN=22882.2 FROM MIGRAD    STATUS=CONVERGED     140 CALLS     141 TOTAL
EDM=2.76132e-09     STRATEGY= 1  ERROR MATRIX UNCERTAINTY   1.9 per cent
```

```
 EXT  PARAMETER                                    STEP         FIRST
 NO.   NAME       VALUE            ERROR          SIZE       DERIVATIVE
  1  Constant     3.86561e+04   1.51316e+02  -2.89095e-02   4.70414e-07
  2  Mean         1.21229e+02   3.60569e-02   2.26724e-05   4.31139e-04
  3  Sigma        1.12871e+01   2.80486e-02   6.50160e-07   2.42478e-01
```

下对拟合得的参数进行格式化输出:

```
fgaus[0]=hquall->GetFunction("gaus");
fgaus[1]=hqdall->GetFunction("gaus");
for(int i=0;i<2;i++) {
    ped[i]=fgaus[i]->GetParameter(1);
    sigma[i]=fgaus[i]->GetParameter(2);
    //TString的格式化输出。用法与printf一致。复杂格式输出推荐用TString::Form()。
    TString ss;
    ss.Form("ped_%s=%.2f, sigma_%s=%.2f",sq[i].Data(),ped[i],sq[i].Data(),sigma[i]);
    cout<<ss<<endl;
}
```

格式化输出结果如下:

```
ped_qu=101.17,sigma_qu=11.31
ped_qd=121.23,sigma_qd=11.29
```