mu 子探测实验数据处理补充材料

地点：物理楼 &220
更新时间：March 24, 2019
邮箱: zhaor25@mail2.sysu.edu.cn

版本: 1.0

第 1 章 数据处理背景简介

实验过程中我们给 PMT 设置合理的高压，设置合理的 ADC 采集参数，通过 ADC 模块采集数据并存储到一个 lvm 格式的文件。这个文件存储了大量（取决于你设置的事件数）的波形信息，我们需要计算每一个信号波形的积分面积，并算出此工作电压下 PMT 的增益大小。

由于我们要处理的波形数量比较多，需要进行简单的编程来计算和统计；python，MATLAB，C++ 等都可以满足需求[1]。

1.1 读取波形

我们存储下来的 lvm 文件每行只有一个数字，代表一个电压值（单位 mV); 如果你在 labview 中设置采样长度是 1000，那么每 1000 行就组成一个波形；每一个点的时间间隔是 1ns。

```
592.000000
592.000000
592.000000
592.000000
591.000000
592.000000
592.000000
592.000000
592.000000
593.000000
592.000000
```

图 1.1: lvm 文件

1.2 计算 PMT 的增益

数据采集 ADC 的阻抗是 50Ω，我们用波形的面积 ($\int UT$) 除以电阻可以得到每一个信号的电荷量大小。实验中我们看到的信号来自于 μ 子在闪烁体中沉积能量产生的发光，我们简单假设每个 μ 子事件平均产生光子数为 5000 个，系统总的的探测效率是 10%，那么我们预期看到的 PMT 信号就是 $5000 \times 10\% = 500$ 个电子被放大之后的结果。PMT 对电子的放大倍数，也就是增益可以用公式1.1来计算：

[1]可以参考附录的 python，ROOT 程序

公式 1.1: 增益计算

$$\delta = \frac{Q_{out}}{q} = \frac{\overline{\int UT}}{Rq}. \tag{1.1}$$

其中 Q_{out} 是放大之后波形的电荷量，q 是未经放大的电荷量；我们假设每个事件的平均光电子数为 500，也就是 $q = 500 \times e$ 。[2]

1.3 波形和积分

在 example 文件中，采样长度是 2002ns，也就是每 2002 点组成一个波形。我们需要算出没有信号时的基线的大小，再用基线值减去对应的信号大小积分出信号的面积。

对于每一个波形，我们都可以计算出一个积分面积，然后算出所有波形[3]积分面积的平均值也可以填充直方图进行拟合作为期望的电荷积分；再利用公式1.1就可以计算出当前工作电压下的增益大小。

注意： 选择合理的时间范围进行积分，不要让信号超出积分的区间。

1.4 电压和增益的关系

调节不同的 PMT 工作电压，计算相应的增益大小，可以做出一条增益和电压相关的曲线，并在一定范围内可以用直线对其拟合。

1.5 μ 子的能量和通量测量

当我们同时[4]在闪烁体两端的 PMT 看到信号时，我们认为测到了一个 μ 子的信号，通过统计一定时间的符合计数并结合探测器的有效探测面积就可以估算 μ 子的通量大小。

[2] e 是电子电荷量
[3] 取决于事件数
[4] 信号时间差小于一定范围

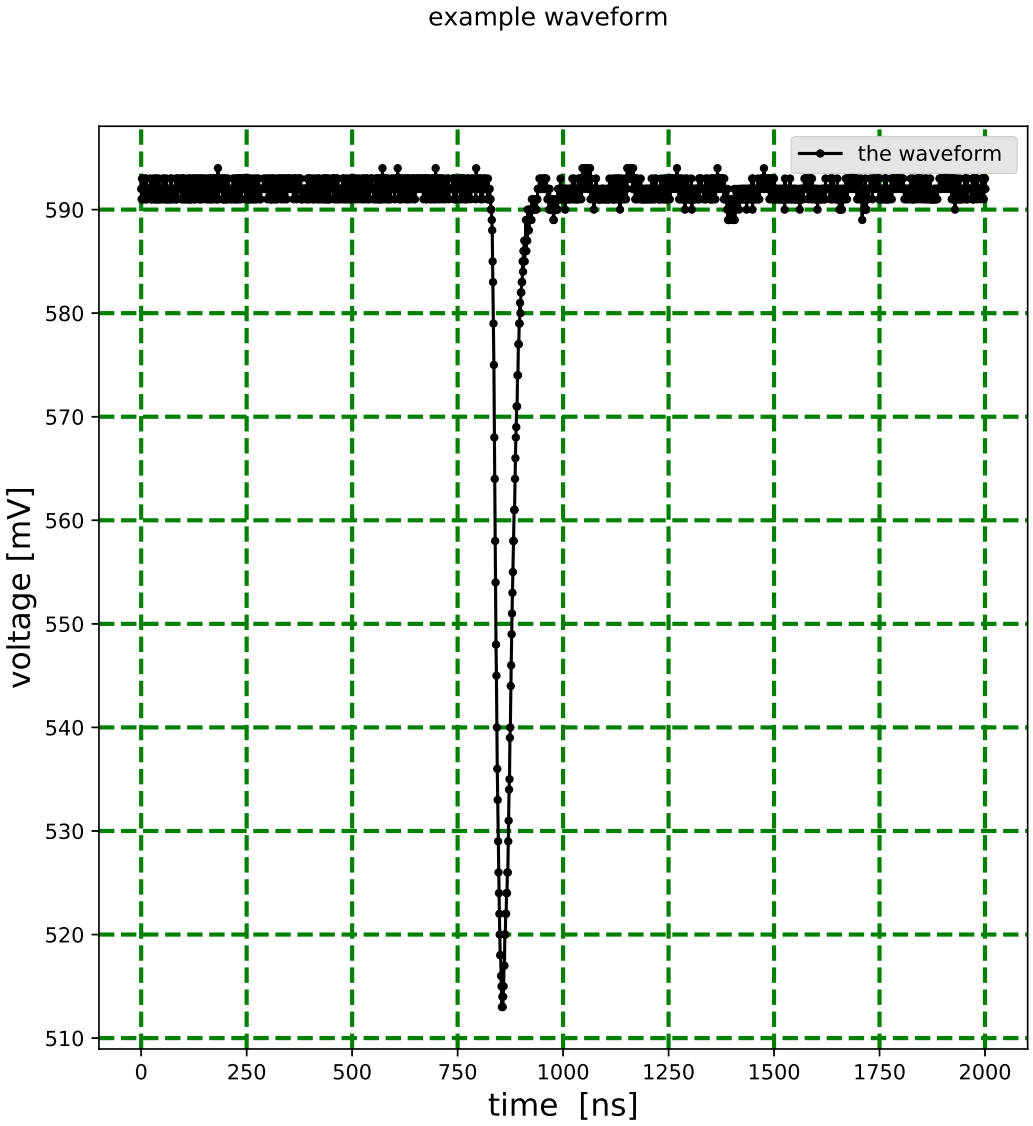

图 1.2: 一个 PMT 信号波形

附录 读取 example 文件的 python 程序

A.1 python

Listing A.1: A simple listing.

```
#!/usr/bin/env python
# -*-coding:utf-8 -*-
import numpy as np
from scipy import interpolate
import pylab as pl

f = open('/media/tao/_dde_data/arlierwork/simulation/tao/exp220/example.
s = f.readline(/media/tao/_dde_data/arlierwork/simulation/tao/exp220)
a1=[]
a2=[]
nevent=1
rec_length=2002    #2002ns 是取数时的采样长度。
count=0
eventnum=0
while (count<rec_length*nevent):
        arr=s.split('␣')
#        print arr
        a1.append(float(arr[0]))
        a2.append(count)
        s=f.readline()
        count+=1
#print(a1)
#print(a2)
#x=np.linspace(0,10,11)
#x=[  0.   1.   2.   3.   4.   5.   6.   7.   8.   9.  10.]
x=np.array(a1)
y=np.array(a2)
#print(x)
```

参考上面的程序，处理多个波形的电荷积分。