**文件说明：**

1. “test#“文件夹下是每次宇宙线测试的原始数据，包括PSD数据和MWDC数据两部分。注意测试编号从6开始，到40结束，中间剔除了部分有问题的测试数据。
2. crate.json是MWDC的探测器通道对应关系文件，在MWDC解码时使用
3. ped.root是PSD正样的基线数据，作为种子文件，在PSD分析时使用
4. code文件夹下是分析软件代码
5. preliminary\_result内保存初步分析得到的刻度结果
6. 分析说明.docx即文件，包含了文件说明和分析软件说明

**软件安装说明：**

1. 依赖cmake与ROOT
2. ROOT需使用cmake方法安装，这样cmake才能用find\_package找到ROOT
3. 进入源代码树
4. 新建build目录，构建过程在此目录下进行
5. 新建install目录，这是分析软件的安装目录，分析过程在此目录下进行
6. 进入build目录，使用cmake命令行”cmake -DCMAKE\_INSTALL\_PREFIX=../install ../”

或者使用ccmake，cmake-gui的GUI界面

1. make install
2. 进入install目录，root -l，此时会自动加载分析代码
3. 安装完成，准备分析。分析代码，都以函数的形式处理数据

下面给出各分析函数的使用说明

**软件使用说明:**

解码：

1. int process\_mwdc(const char\* datadir,const char\* outfile,const char\* prefix,const char\* jsondir)

解码MWDC数据，参数如下：

datadir:MWDC原始数据所在目录，如”test6”

outfile: root文件名，保存解码后的结果，如“mwdc6.root”

prefix: MWDC数据一组共9个文件，每个文件用文件名前缀加获取板编号的方式来区分。其中获取板编号是固定的，此处的prefix指的是文件名前缀如“mwdc6\_”

jsondir: MWDC的通道对应关系配置文件crate.json所在的目录。crate.json是用标准的的JSON语法编写的通道对应关系文件

1. int process\_psd(const char\* datadir,const char\* rawfile,const char\* outfile,const char\* stdpedfile)

解码PSD数据，参数如下：

datadir: PSD原始数据所在目录，如”test6”

rawfile: PSD原始数据文件名，如“2015-02-23-09-20-03.dat”

outfile: root文件名，保存解码后的结果，如“psd6.root”，注意不要使用与MWDC相同的root文件

stdpedfile: PSD的基线种子文件的绝对路径，即ped.root文件的绝对路径（包括文件名）

简单重建（在解码的基础上）：

1. int extract\_event(const char\* datadir,int testindex,const char\* outdir,const char\* psd\_pedfile,float psdoffset\_x=6.35,float psdoffset\_y=-2.21,int ped\_cut=5)

合并MWDC和PSD数据，参数如下：

datadir: 宇宙线测试数据（包括PSD和MWDC）所在的文件夹

testindex: 测试编号，如“test6”中的6

outdir: 合并后的结果保存路径，文件名为run\_simple\_#.root,#为测试编号

psd\_pedfile:PSD的基线种子文件的绝对路径，即ped.root文件的绝对路径（包括文件名）

psdoffset\_x: PSD在X方向上的位置偏差，默认值即可

psdoffet\_y: PSD在Y方向上的位置偏差，默认值即可

psd\_cut： 多少个sigma作为PSD的击中判选阈值

4. int draw\_trajectory\_final(char\* infile)

观察重建后的径迹，可以验证重建准确性：

infile: 合并后的MWDC和PSD的文件，如“run\_simple\_6.root”

分析（在重建基础上）：

5. int ana\_efficiency(char \*infile, char \*outfile,float psdoffset\_x=6.35,float psdoffset\_y=- 2.21,float mwdc\_range=13.0)

PSD效率计算：

infile: 合并后的MWDC和PSD的文件，如“run\_simple\_6.root”

outfile: 保存计算结果的文本文件绝对路径

其它参数使用默认值即可

6. int draw\_mips(char\* infile,char\* outfile,float psdoffset\_x=6.35,float psdoffset\_y=-2.21,float segment\_limit=10)

得到各个位置的MIP谱图（每根单元条41个位置点,2cm一个点）：

infile: 合并后的MWDC和PSD的文件，如“run\_simple\_6.root”

outfile： 保存结果的root文件的绝对路径

其它参数使用默认值即可

7. int fit\_mips(char\* infile,char\* outfile,bool gflag=false)

得到各个位置的MIP峰位，并画出衰减曲线：

infile： 步骤6得到的输出文件

outfile: 保存结果的root文件的绝对路径

8. int draw\_energy\_resolution(char\* infile,char\* outfile)

计算各单元条的能量分辨：

infile： 步骤6得到的输出文件

outfile: 保存结果的root文件的绝对路径

9. int draw\_consistency(char\* infile,char\* outfile)

得到各单元条的能量响应一致性：

infile： 步骤6得到的输出文件

outfile: 保存结果的root文件的绝对路径

10. int ana\_position\_resolution(char\* infile,float psdoffset\_x=6.35,float psdoffset\_y=-2.21)

得到位置分辨：

infile:合并后的MWDC和PSD的文件，如“run\_simple\_6.root”

结果输出到infile文件中

11. int draw\_dy58(char\* infile,char\* outfile)

输出dy58谱线图：

infile: infile： 步骤2得到的输出文件，如“psd6.root”

outfile: 保存结果的root文件的绝对路径

12. int fit\_dy58(char\* infile,char\* pedfile,char\* outfile,int pedcut=7,float range=200.0)

拟合并得到dy58比值：

infile: infile： 步骤11得到的输出文件，如“psd6.root”

pedfile: PSD的基线种子文件的绝对路径，即ped.root文件的绝对路径（包括文件 名）

outfile: 保存结果的文件的绝对路径