

# 粒子物理与核物理实验中的数据 分析

---

杨振伟  
清华大学

第三讲：ROOT在数据分析中的应用(1)

# 上讲摘要

## ■ C++基本概念

类的定义与实现...

## ■ Linux下用g++编译C++程序

```
g++ -o hello.exe -I<include> ./src/*.cc
```

当前目录下输出  
可执行文件hello.exe

指定include目录  
如-I./include

源文件

## ■ 用makefile进行C++编译

gmake 进行编译

gmake clean 清除编译结果

## ■ 使用ROOT脚本 root -l hello.C

# 本讲要点

---

- 什么是ROOT ?
- 登录ROOT环境和体验中心
- ROOT的语法简介
- ROOT的函数, **直方图**, **随机数**, 文件, 散点图

TF1, TH1I, TH1F, TH1D, TRandom (gRandom)

TF2, TF3, TH2F, ...

TFile

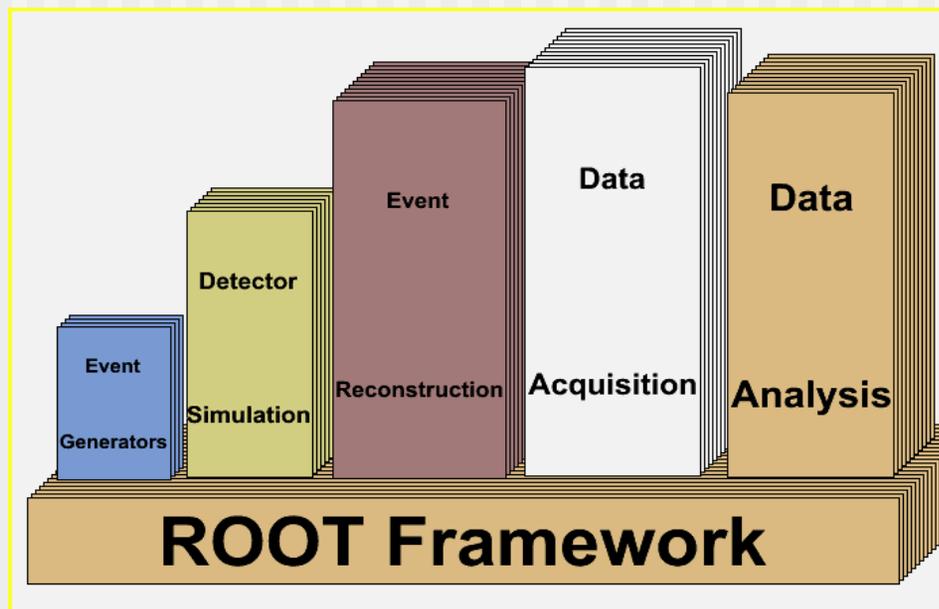
# 什么是 ROOT ?

## ROOT: Executive Summary

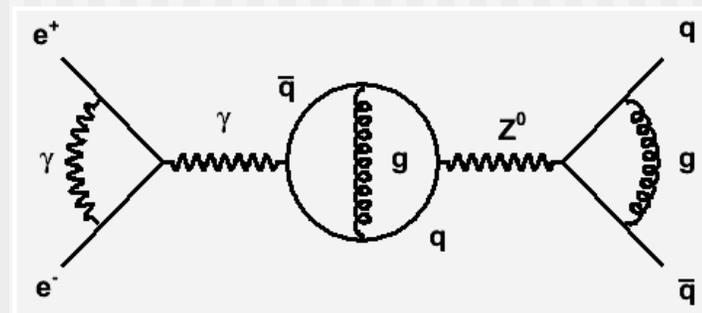
... provides a set of **OO frameworks** with **all the functionality** needed to handle and analyse **large amounts of data** in a **very efficient** way....

(摘自 <http://root.cern.ch/root/Mission.html>)

关键字：面向对象的框架、所有功能、海量数据、非常有效



结论：很不谦虚！



# 安装ROOT(1)

到ROOT主页下载需要的版本到指定目录。

比如要在SLC3系统的/projects/yangzw目录下安装5.16.00版本

(注：最新版本的ROOT已经不为SLC3提供预编译版本了，而为SLC4和SLC5提供)

```
cd /projects/$USER
```

 (注：对用户yangzw, \$USER=yangzw)

```
wget ftp://root.cern.ch/root/root\_v5.16.00.Linux.slc3.gcc3.2.3.tar.gz
```

```
tar -zxvf root_v5.16.00.Linux.slc3.gcc3.2.3.tar.gz
```

设置ROOT的环境变量

```
export ROOTSYS=/projects/$USER/root
```

```
export PATH=$ROOTSYS/bin:$PATH
```

```
export LD_LIBRARY_PATH=$ROOTSYS/lib:$LD_LIBRARY_PATH
```

可以把上面这3行放到\$HOME/.login或者.bashrc或者.tcshrc文件中，

这样每次登录到Linux系统，系统就自动设置ROOT的环境变量

这样，进入linux系统之后，在终端提示行输入：

```
root 或
```

```
root -l
```

即可进入ROOT环境。

# 安装ROOT(2)

如果是其它发行版的Linux，首先查看是否ROOT网站上是否有预编译好的程序包，一般情况下，官方提供SLC4和SLC5在各种不同CPU以及不同gcc版本下的二进制包，

ROOT官网也提供包括Solaris以及Mac OS X以及Windows下的预编译包。

如果没有适合你的操作系统的预编译包，就需要到官网<http://root.cern.ch> 下载ROOT的源代码，按照安装指南用gmake编译安装。

Window用户在官网下载相应的.msi文件直接安装即可。

Ubuntu8.10用户可以到下面网页下载5.22.00版本的二进制代码，根据Readme.txt说明安装使用。

# 安装ROOT(3)

实际上，Linux下安装程序的基本套路很简单：

1. 如果需要用源码编译
  - a) 下载源码压缩包
  - b) 解压缩
  - c) 编译
  - d) 设置环境变量(如果需要)
2. 如果已有预编译的包
  - a) 下载
  - b) 解压缩
  - c) 设置环境变量(如果需要)
3. yum/apt-get直接用网络源安装(预编译的包)
4. ...

# 登录ROOT环境

■ 运行 `>root`

■ 退出 `root[0].q`

■ 键入 `help` 指令，如

`root[0]?`

`root[1].ls`

`root[2].!ls`

```
[training] /home/yangzw > root
*****
*
*      W E L C O M E  to  R O O T      *
*
*   Version   5.20/00      24 June 2008  *
*
* You are welcome to visit our Web site *
*      http://root.cern.ch      *
*
*****
ROOT 5.20/00 (trunk@24524, De 05 2008, 11:09:00 on linux)

CINT/ROOT C/C++ Interpreter version 5.16.29, Jan 08, 2008
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0]
```

ROOT环境其它常用指令：

`.L macro.C`    Load文件macro.C

`.x macro.C`    执行文件macro.C

`.ls`            显示ROOT当前环境的所有信息

`!ls`            显示Linux系统当前目录的所有信息

注：ROOT环境中，**ROOT指令都以“.”开头**  
**系统指令都以“!”开头**

# ROOT体验中心(1)

在\$ROOTSYS/tutorials目录下，有五花八门的例子。  
以后会经常与这个目录打交道。先尝试一下吧。

尝试方法：

```
>cd /projects/$USER
```

```
>cp -r $ROOTSYS/tutorials . (注意不要把这个"."漏掉了)
```

```
>cd tutorials
```

然后找个感兴趣的目录/文件，  
执行ROOT脚本，比如

```
>cd roofit
```

```
>root -l RoofitDemo.C
```

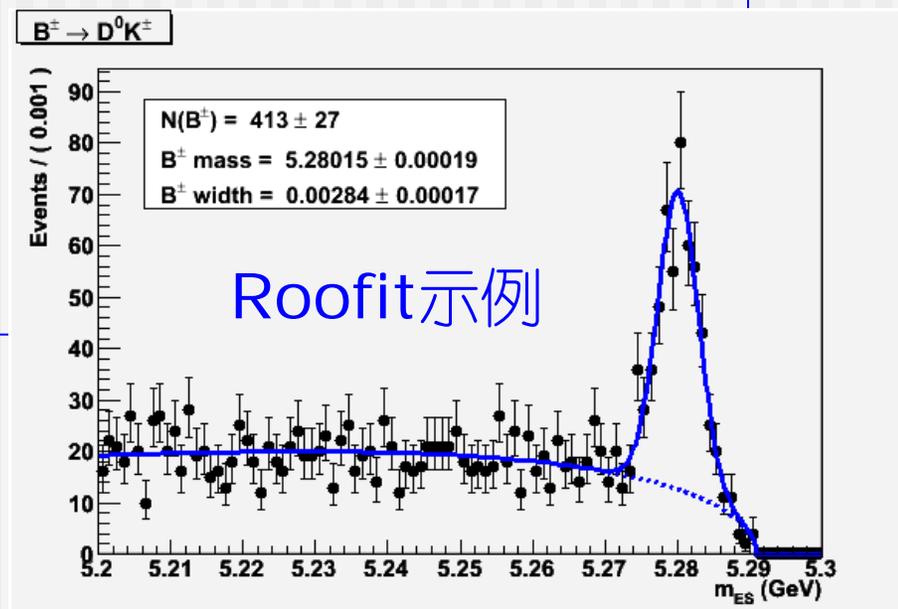
小技巧提示：

根据关键字"xxxx"从tutorials的例子中寻找线索

```
grep -sirn "xxxx" $ROOTSYS/tutorials
```

比如找随机数用法：

```
grep -sirn "random" $ROOTSYS/tutorials
```



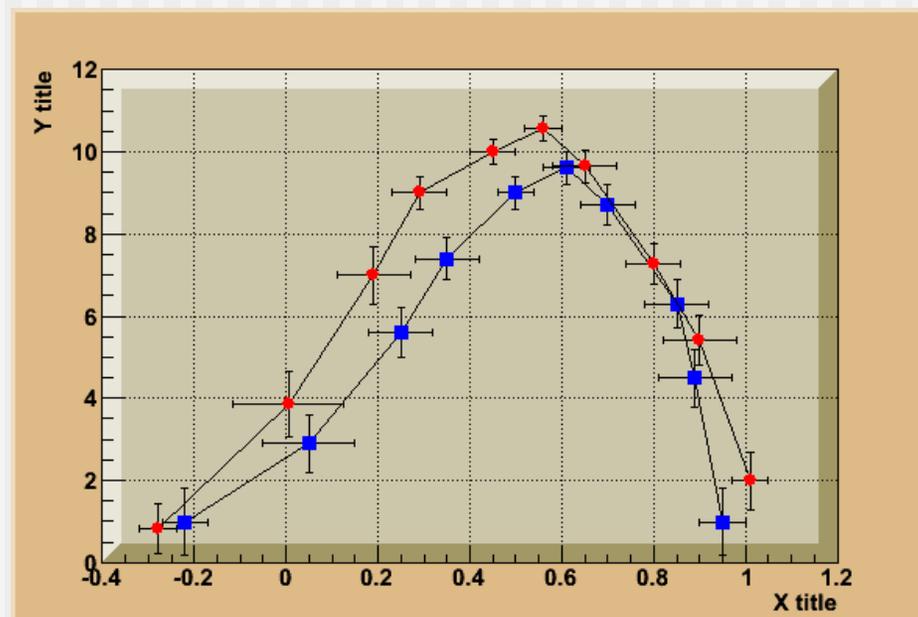
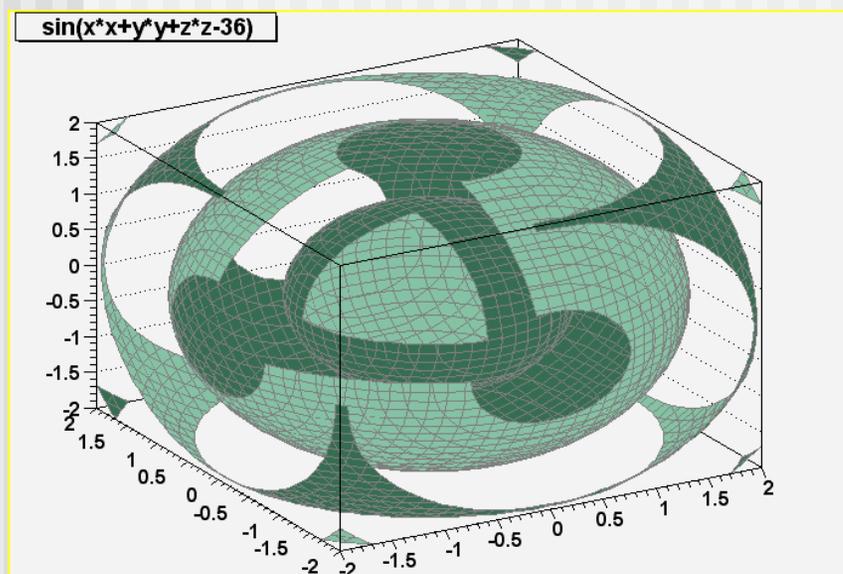
# ROOT体验中心(2)

还可以在ROOT网站上看到一些ROOT图片：

<http://root.cern.ch/drupal/image>

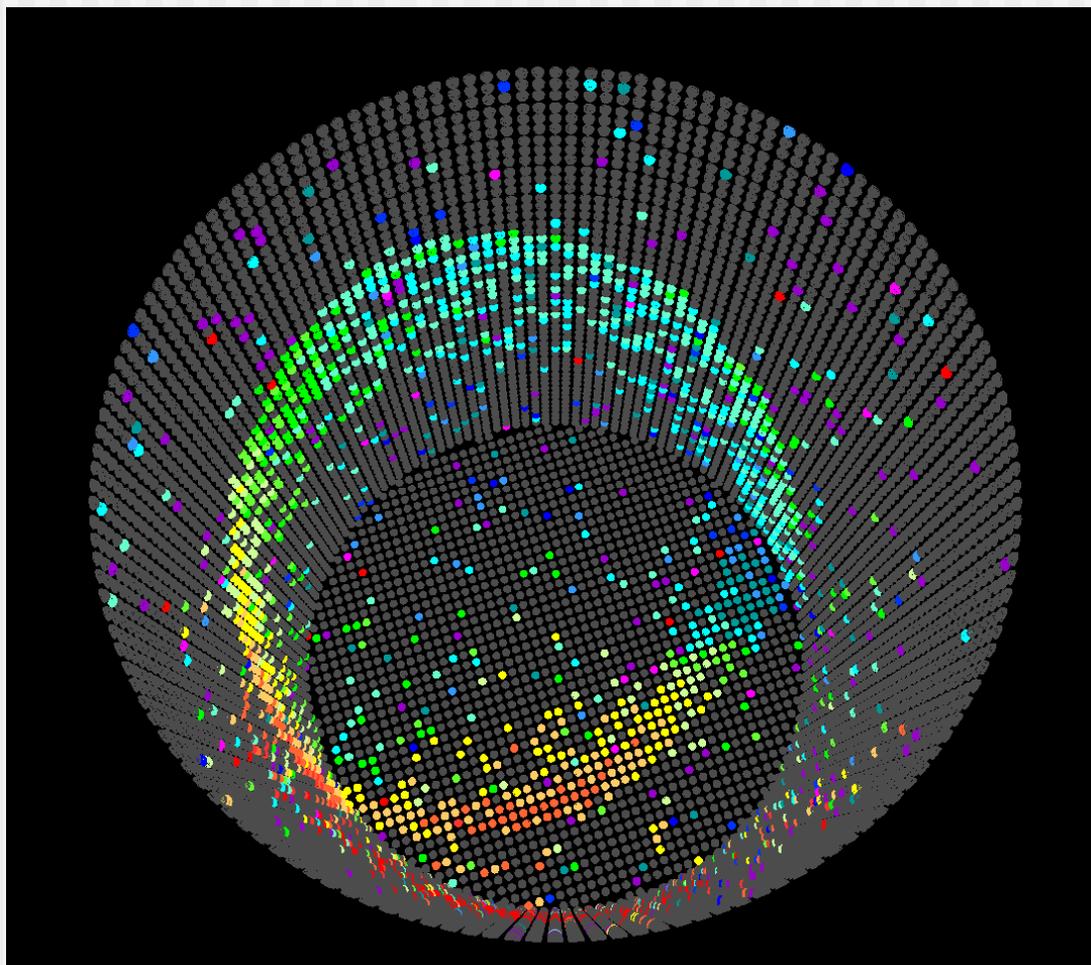
当然，ROOT的功能不只是做图，它不是一个作图工具。  
跟数据分析有关的东西，基本都是ROOT的擅长；  
跟物理有关的很多东西，ROOT基本都可以做得很好：

事例产生、探测器模拟、事例重建、数据采集、**数据分析**



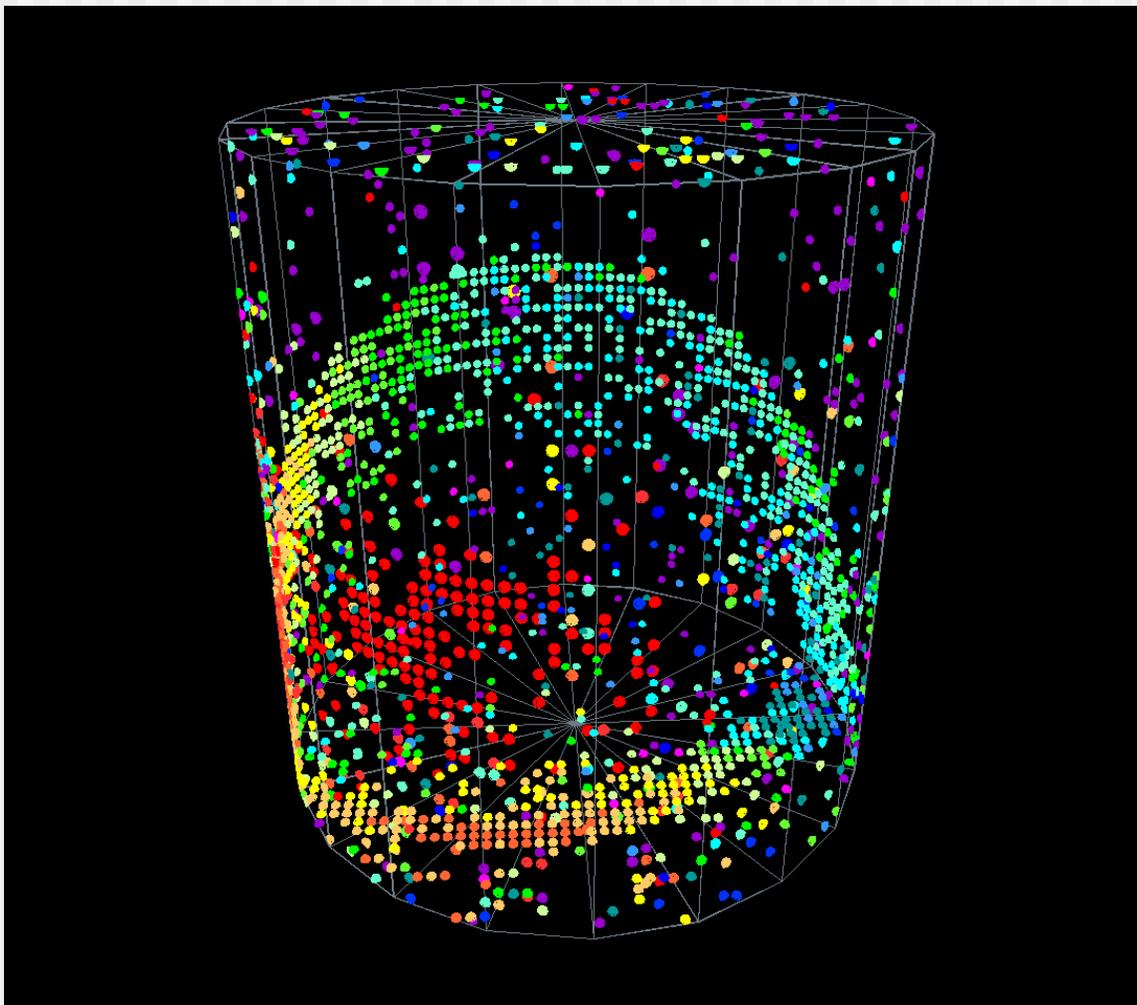
# ROOT体验中心(3-1)

日本超级神冈中微子实验事例显示 (by zhanghb)  
超大的水池，内外装满了光电倍增管，1万多个



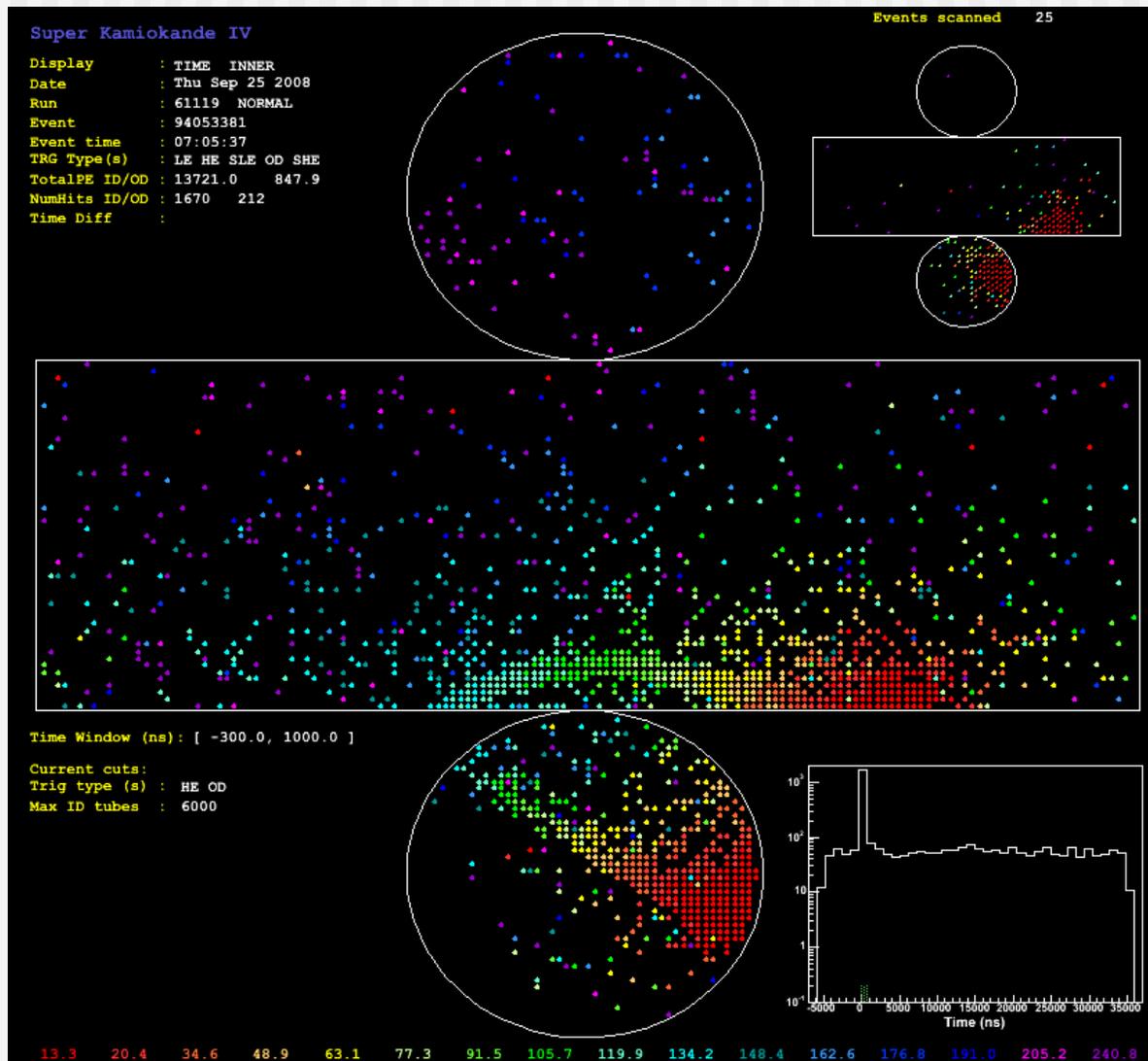
# ROOT体验中心(3-2)

仅显示被击中的光电倍增管



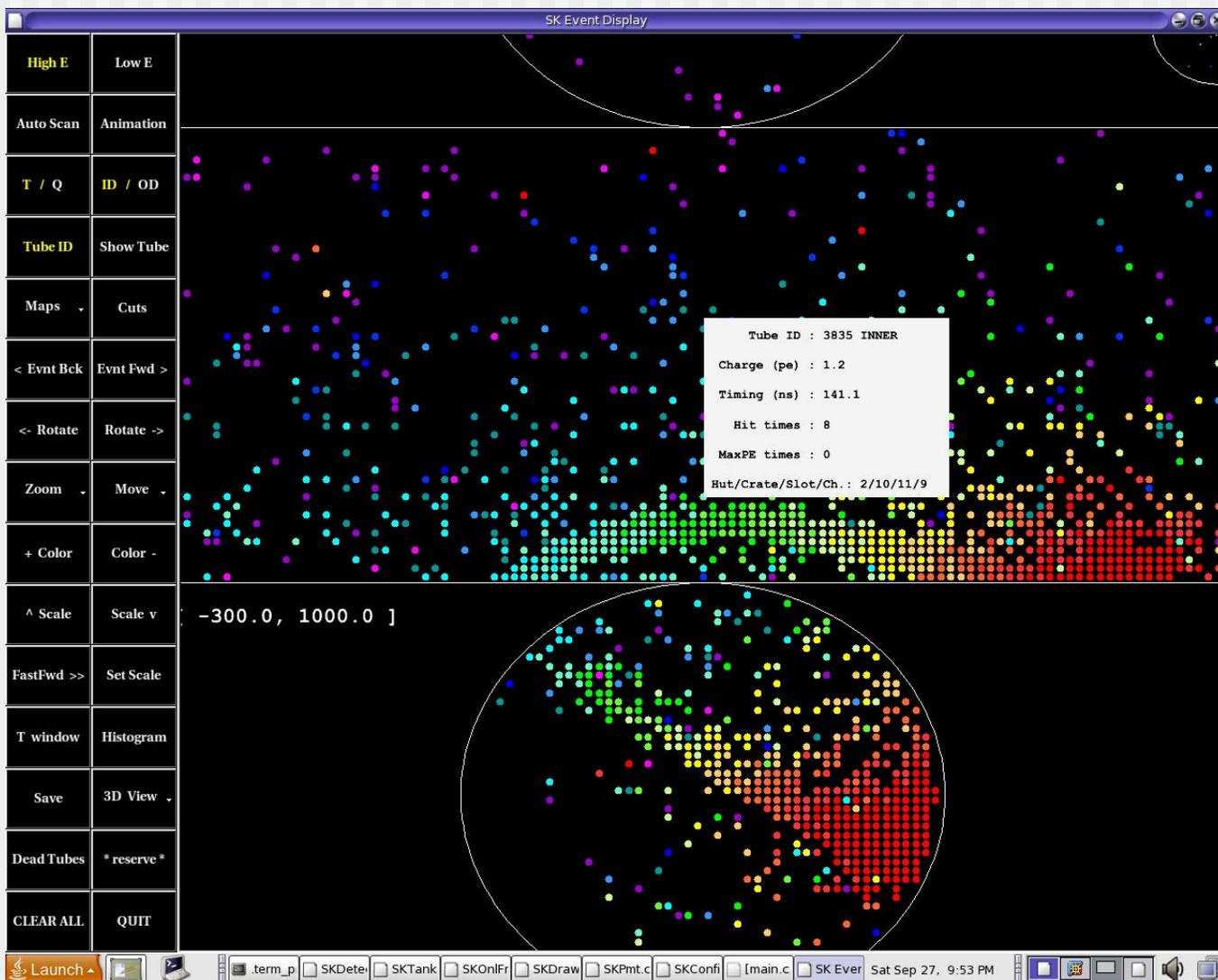
# ROOT体验中心(3-3)

## 平面展开显示



# ROOT体验中心(3-4)

平面展开，鼠标缩放，显示鼠标位置光电倍增管信息



# ROOT语法(1)—基本信息

- ROOT使用C++语法

一段C++程序可以直接在ROOT环境运行

- 数据类型重定义

int → Int\_t

float → Float\_t

double → Double\_t

.....

- ROOT的类都以T开头

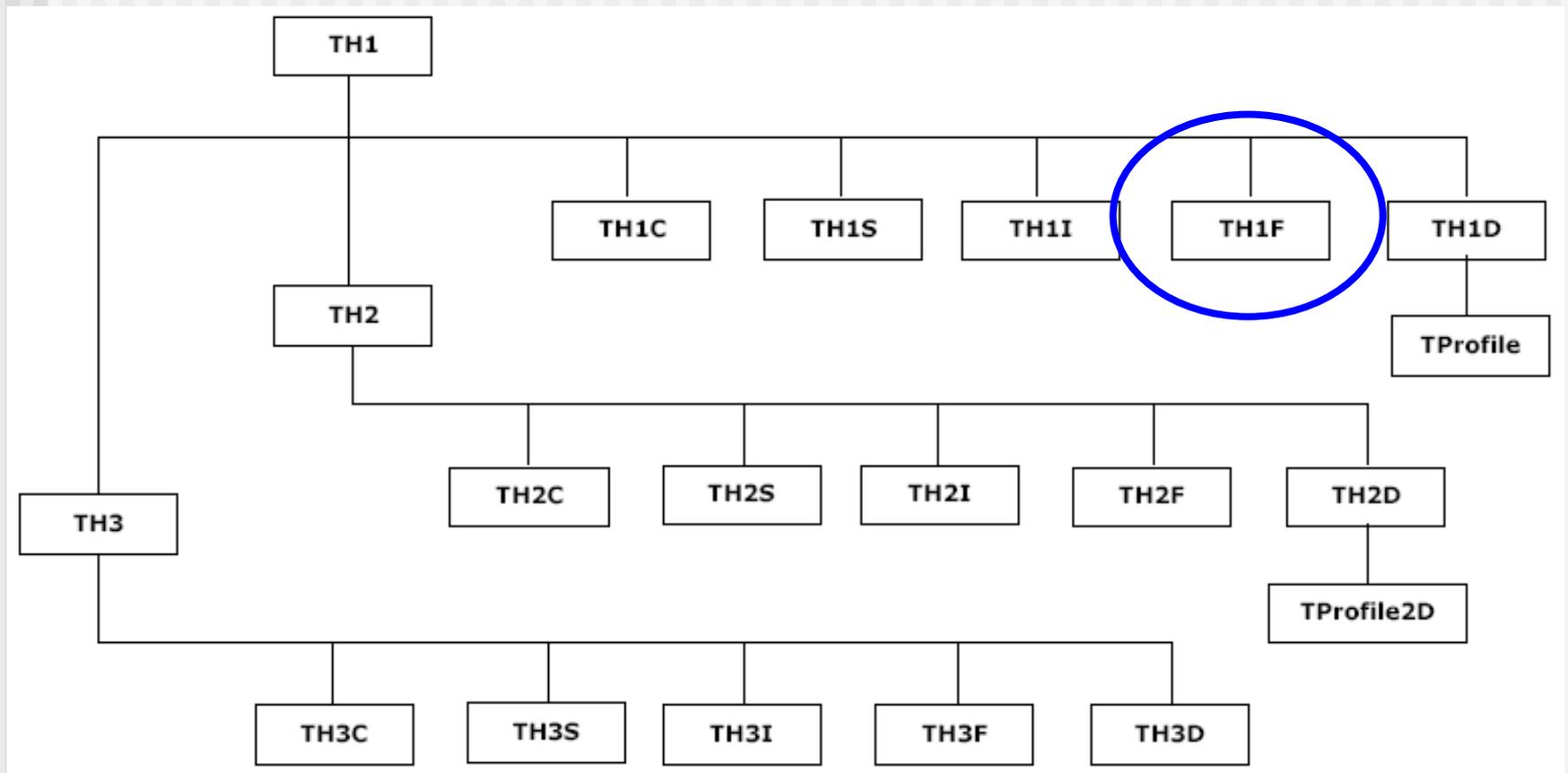
如TFile, TH1F, TTree, ...

- 详细规定参阅ROOT手册(5.21版)第18-20页, 关于Convention和Global Variables部分。

- 可以直接在ROOT环境中运行macro文件(自动调用cint编译器), 也可以在makefile中设置好相关参数用g++编译得到可执行文件运行。

# ROOT语法(2)—直方图类

ROOT中有众多已经定义好的类可供使用，比如直方图家族



# ROOT语法(2)—其它类

其它常用类

数学函数: TF1, TF2, TF3...

图形: TGraph, TGraphErrors, TGraph2D, ...

文件: TFile

画布: TCanvas, TPad, ...

随机数: TRandom, TRandom1, TRandom2, TRandom3

周期	$10^9$	$10^{171}$	$10^{26}$	$10^{6000}$
速度(ns/call)	34	242	37	45

比如跟数据结构和分析有关的:

速度与CPU和编译器有

TTree, TChain, ...

参见 <http://root.cern.ch/root/html522/ClassIndex.html>

(谨代表\*\*\*邀请各位光临敝舍。注: \*\*\*=yangzw)

还有很多全局函数, 多数以g开头, 如:

gRandom, gROOT, gStyle, gPad, gEnv, gFile...

# ROOT语法(3)—随机数

`gRandom`是指向当前随机数产生器的指针，该产生器默认为 `TRandom3` 对象。

<http://root.cern.ch/root/html522/TRandom.html>

(为什么看 `TRandom`? 因为 `TRandom1/2/3` 都继承自 `TRandom`)

<code>gRandom-&gt;Binomial(ntot, p):</code>	二项分布
<code>gRandom-&gt;BreitWigner(mean, gamma):</code>	Breit-Wigner分布
<code>gRandom-&gt;Exp(tau)</code>	指数分布
<code>gRandom-&gt;Gaus(mean, sigma)</code>	高斯分布
<code>gRandom-&gt;Integer(imax)</code>	(0, imax-1) 随机整数
<code>gRandom-&gt;Landau(mean, sigma)</code>	Landau分布
<code>gRandom-&gt;Poisson(mean)</code>	泊松分布(返回int)
<code>gRandom-&gt;PoissonD(mean)</code>	泊松分布(返回double)
<code>gRandom-&gt;Rndm()</code>	(0, 1] 均匀分布
<code>gRandom-&gt;Uniform(x1, x2)</code>	(x1, x2] 均匀分布
....	

使用前可根据需要改变随机数种子和机制

思考：什么情况下需要 `PoissonD(mean)`?

# ROOT脚本文件示例(1): Macro文件

/home/yangzw/examples/Lec3/ex31.C

用花括号括起来，后缀名一般用“.C”

```
{  
    cout << "Hello ROOT" << endl;  
    int Num=5;  
    for (int i=0;i<Num;i++) {  
        cout << "i=" << i << endl;  
    }  
}
```

纯粹C++语法，执行的时候只需要在命令提示行：

```
cd /projects/$USER
```

```
cp -r ~yangzw/examples/Lec3 . (注意最后有个“.”)
```

```
cd Lec3
```

```
root -l ex31.C
```

# ROOT中的数学函数

## □制作一维函数曲线图

```
TF1 *fun_name = new TF1("fun_name","expression",  
x_low,x_high);
```

```
root[0]TF1 *f1 = new TF1("f1","x*sin(x)",-5,5);
```

## □制作二维函数曲线图

```
TF2 *fun_name = new TF2("fun_name","expression",  
x_low,x_high, y_low,y_high);
```

```
root[0]TF2 *f2 = new TF2("f2","x*sin(x)+y*cos(y)",  
-5,5,-10,10);
```

## □制作三维函数曲线图

```
TF3 *fun_name = new TF3("fun_name","expression",  
x_low,x_high,y_low,y_high,z_low,z_high);
```

```
root[0]TF3 *f3 = new TF2("f3","x*sin(x)+y*cos(y)  
+z*exp(z)",-5,5,-10,10,-20,20);
```

# 数学函数的定义方式(1)

## ROOT中定义数学函数的方式多种多样

### □ 利用C++数学表达式

```
TF1* f1 = new TF1("f1","sin(x)/x",0,10);
```

### □ 利用TMath定义的函数

```
TF1 *f1 = new TF1("f1","TMath::DiLog(x)",0,10);
```

### □ 利用自定义C++数学函数

```
Double_t myFun(x) {  
    return x+sqrt(x);  
}  
TF1* f1 = new TF1("f1","myFun",0,10);
```

以上函数都不含参数，但在数据拟合时，我们往往需要定义含未知参数的函数

## 数学函数的定义方式 (2)

### ROOT中定义含未知参数的数学函数

□ ROOT已经预定义了几种常用的含参函数

**gaus**: 3个参数

$$f(x) = p_0 * \exp(-0.5 * ((x - p_1) / p_2)^2)$$

**expo**: 2个参数

$$f(x) = \exp(p_0 + p_1 * x)$$

**polN**: N+1个参数

$$f(x) = p_0 + p_1 * x + p_2 * x^2 + \dots$$

其中  $N = 0, 1, 2, \dots$ , 使用时根据需要用  $pol_0, pol_1, pol_2 \dots$

**landau**: 3个参数

朗道分布, 没有解析表达式

这些预定义函数可直接使用, 比如

```
histogram->Fit("gaus"); //对直方图进行高斯拟
```

```
TF1 *f1=new TF1("f1","gaus",-5,5);
```

# 数学函数的定义方式 (3)

## ROOT中自定义含未知参数的数学函数

### □ 利用C++数学表达式

```
TF1* f1 = new TF1("f1", "[0]*sin([1]*x)/x", 0, 10);
```

### □ 利用C++数学表达式以及ROOT预定义函数

```
TF1* f1 = new TF1("f1", "gaus(0)+[3]*x", 0, 3);
```

### □ 利用自定义的C++数学函数

```
Double_t myFun(Double_t *x, Double_t *par) {  
    Double_t xx=x[0];  
    Double_t f=par[0]*exp(-xx/par[1]);  
    return f;  
}
```

```
TF1* f1 = new TF1("f1", "myFun", 0, 10, 2);
```

指定参数数目



定义了含参的TF1对象f1之后，可以设定参数初值，比如  
`f1->SetParameter(0,value);` //为第0个参数设初值为value

# ROOT中统计直方图

## ❑ 定制一维直方图

```
TH1F *hist_name = new TH1F("hist_name","hist_title",  
num_bins,x_low,x_high);
```

## ❑ 定制二维图

```
TH2F *hist_name = new TH2F("hist_name","hist_title",  
num_bins_x,x_low,x_high,num_bins_y,y_low,y_high);
```

## ❑ 定制三维图

```
TH3F *hist_name = new TH3F("hist_name","hist_title",  
num_bins_x,x_low,x_high,num_bins_y,y_low,y_high,  
num_bins_z,z_low,z_high);
```

## ❑ 填充统计图

```
hist_name.Fill(x);  
hist_name.Fill(x,y);  
Hist_name.Fill(x,y,z);
```

**绘图：**

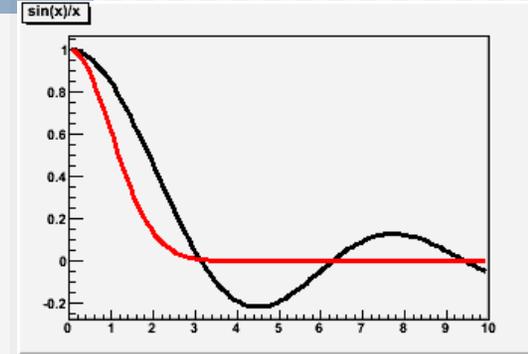
```
root[0]hist_name.Draw();
```

# ROOT脚本文件示例(2): 数学函数定义

/home/yangzw/examples/Lec3/ex32.C

```
//a simple ROOT macro, ex32.C  
//说明ROOT中数学函数的使用, 如TF1
```

```
void ex32() {  
    //定义函数  
    TF1 *f1 = new TF1("func1", "sin(x)/x", 0, 10);  
    f1->Draw(); //画出函数图像  
    TF1 *f2 = new TF1("func1", "TMath::Gaus(x,0,1)", 0, 10);  
    f2->SetLineColor(2); //设置颜色为红色  
    f2->Draw("same"); //用参数"same", 把f1,f2画在同一个画布上  
}
```



**运行:** 在命令提示行下 `root -l ex32.C`  
或在ROOT环境下 `.x ex32.C`

**提示:** 1)脚本中void函数的名字必须与文件名相同(如ex32)  
2)ROOT环境中定义类指针之后, 如TF1 \*f1, 之后输入“f1->”, 然后按一下Tab键, 可以自动列出该类对象的成员函数和成员变量



# ROOT脚本文件示例(3): 画布, 保存图片

/home/yangzw/examples/Lec3/ex33.C

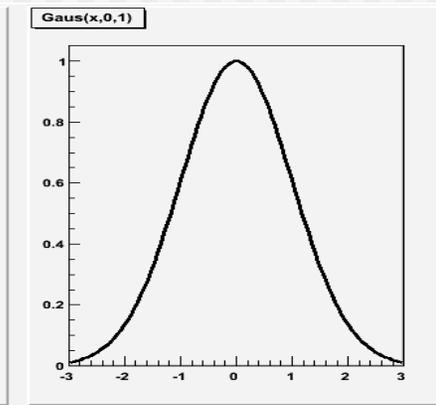
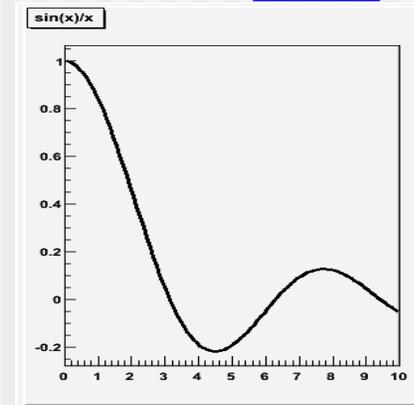
//说明ROOT画布的使用, TCanvas, 保存图形

```
void ex33() {  
    //define a function sin(x)/x  
    TF1 *f1 = new TF1("func1","sin(x)/x",0,10);  
    //define a Gaussian function, mean=0, sigma=1  
    TF1 *f2 = new TF1("func2","Gaus(x,0,1)",-3,3);  
    //定义一个画布, TCanvas  
    TCanvas *myC1 = new TCanvas("myC1","A Canvas",10,10,800,600);  
    //将画布分成两部分  
    myC1->Divide(2,1);  
    myC1->cd(1); //进入第一部分  
    f1->Draw();  
    myC1->cd(2); //进入第二部分  
    f2->Draw();  
    myC1->SaveAs("myex33.gif");  
    myC1->SaveAs("myex33.eps");  
}
```

像素坐标 (10,10):左上角  
(800,600):右下角

名称

描述



运行: 在命令提示行下 > root -l ex33.C  
或在ROOT环境下 root[0] .x ex33.C

# ROOT脚本文件示例(4a):直方图, 随机数

/home/yangzw/examples/Lec3/ex34a.C

//说明ROOT直方图、随机数的使用, 如TH1F, gRandom

```
void ex34a() {
```

```
    const Int_t NEntry = 10000;
```

```
    //创建一个root文件
```

```
    TFile *file = new TFile("hist1.root", "RECREATE");
```

```
    TH1F *h1 = new TH1F("h1", "A simple histo", 100, 0, 1);
```

```
    //填充直方图10000次, 用(0,1)均匀分布
```

```
    for (int i=0; i<NEntry; i++) h1->Fill( gRandom->Uniform() );
```

```
    h1->Draw();
```

```
    h1->GetYaxis()->SetRangeUser(0,150);
```

```
    h1->GetXaxis()->SetTitle("x");
```

```
    h1->GetXaxis()->CenterTitle();
```

```
    file->cd(); //进入文件file
```

```
    h1->Write(); //将h1写入文件
```

```
}
```

名称

描述

No. of Bin

区间

调用均匀分布Uniform(), 其它:  
[Landau](#)(mean,sigma);  
[Binomial](#)(ntot,prob);  
[Poisson](#)(mean);  
[Exp](#)(tau);  
[BreitWigner](#) (mean,sigma);

执行的时候只需要在命令提示行 `root -l ex34a.C`  
或者进入ROOT环境之后, 运行 `.x ex34a.C`

# ROOT脚本文件示例(4b): 随机数-舍选法

/home/yangzw/examples/Lec3/ex34b.C

```
void ex34b() {
  //gDirectory->Delete("*;*");
  Float_t xMin = 0.0 ;
  Float_t xMax = 1.0 ;
  TH1F *hX = new TH1F("hX","sawtooth p.d.f.",100,xMin,xMax);
  gRandom->SetSeed();
  for (int i=0;i<10000;i++) {
    float x=mypdf(xMin,xMax); //舍选法产生随机分布
    hX->Fill(x);
  }
  hX->Draw("e");
}

float mypdf( float xMin, float xMax ){
  float fmax = 2.; //寻找分布函数最大值
  while (1) {
    float r = gRandom->Uniform(xMin,xMax); //1st随机数(xMin,xMax)
    float z = 2.*r/xMax/xMax; //期待的分布函数
    float u = gRandom->Uniform(0.,fmax); //2nd随机数(0,fmax)
    if(u<=z) return r;
  }
}
```

执行的时候只需要在命令提示行 `root -l ex34b.C`  
或者进入ROOT环境之后, 运行 `.x ex34b.C`

# ROOT脚本文件示例(4c): 随机数

/home/yangzw/examples/Lec3/ex34c.C

也可以利用类TF1、TF2或TF3自定义函数，通过调用GetRandom()函数获得服从自定义函数分布的随机数：

```
TF1 *f1 = new TF1("f1","abs(sin(x)/x)*sqrt(x)",0,10);  
double r = f1->GetRandom();
```

```
void ex34c() {  
    //定义直方图  
    TH1F *h1 = new TH1F("h1","histogram from TF1",100,0,10);  
    //定义TF1函数  
    TF1 *f1 = new TF1("f1","abs(sin(x)/x)*sqrt(x)",0,10);  
    for (int i=0;i<10000;i++) {  
        double r = f1->GetRandom(); //按照f1分布产生随机数  
        h1->Fill(r);  
    }  
    h1->Draw();  
}
```

执行时只需要在命令提示行 `root -l ex34c.C`  
或进入ROOT环境后，运行 `.x ex34c.C`

感兴趣者可以看看TF1的GetRandom()函数是如何实现的。实际上，是把SDA(3.5)-(3.6)进行数值积分得到 $x(r)$ 。

当函数f1有陡峰时，要小心！这时可能需要改变一些参数。

# 直方图、打开root文件

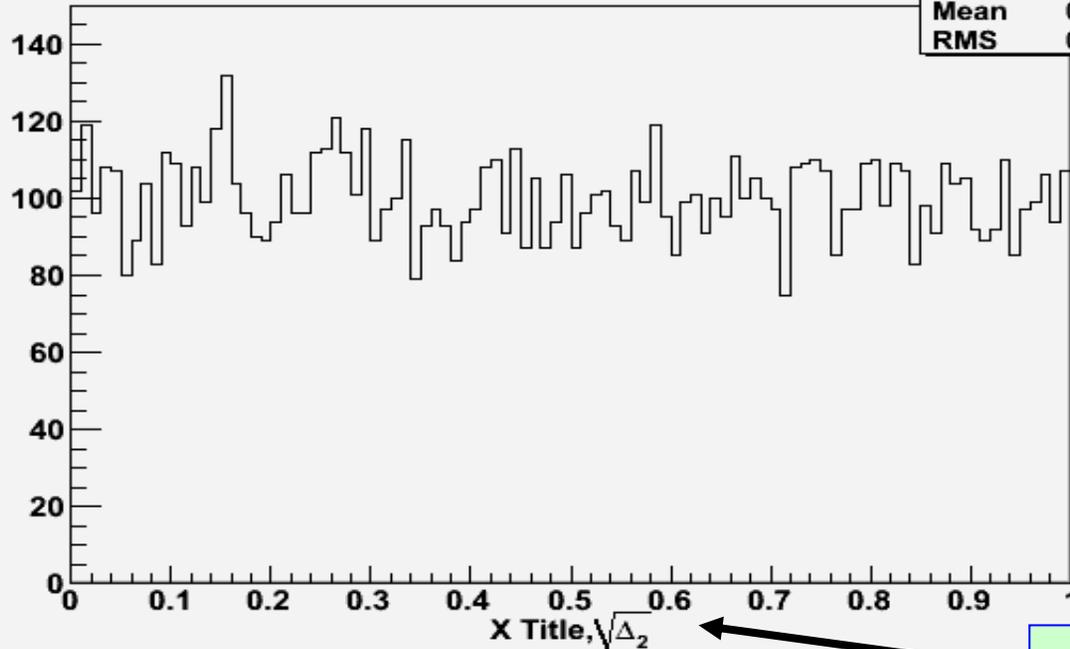
A simple histo

直方图的描述

h1	
Entries	10000
Mean	0.4969
RMS	0.2897

直方图统计信息  
事例数: Entries  
均 值: Mean  
方 差: RMS

参见ROOT手册37页  
"Statistics Display"



X轴的名称

打开已有的root文件, 如hist1.root:

终端提示行下:

```
root -l hist1.root
```

ROOT环境下:

```
TFile f1("hist1.root");
```

```
.ls
```

```
h1->Draw();
```

```
[training] /home/yangzw/workdir/examples/Lec3 > root -l
root [0] TFile f1("hist1.root");
root [1] .ls
TFile**      hist1.root
TFile*       hist1.root
KEY: TH1F    h1:1    A simple histo
root [2] h1->Draw("e");
```

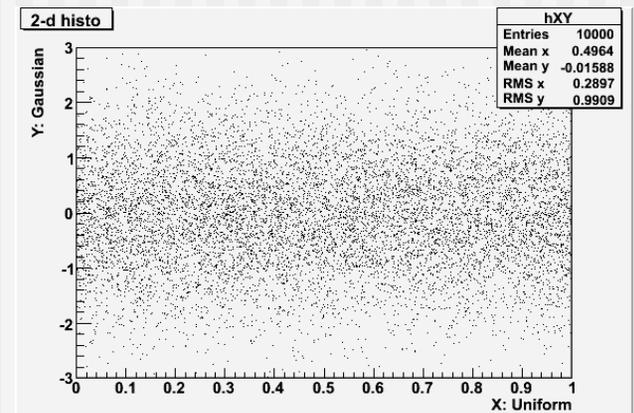
# ROOT脚本文件示例(5): 散点图

/home/yangzw/examples/Lec3/ex35.C

//2维直方图TH2F, 散点图, 散点图的协方差

```
void ex35() {  
    const Int_t NEntry = 10000 ;  
    TH2F *hXY = new TH2F("hXY","2d histo",100,0,1,100,-3,3);  
    for (int i=0;i<NEntry;i++) {  
        float x = gRandom->Rndm() ;  
        float y = gRandom->Gaus(0,1) ;  
        hXY->Fill(x,y) ; //填充2维直方图  
    }  
    hXY->Draw(); //2维直方图的散点图  
    hXY->GetXaxis()->SetTitle("X: Uniform" );  
    hXY->GetYaxis()->SetTitle("Y: Gaussian");  
    Float_t covar = hXY->GetCovariance(); //协方差  
    cout << "Covariance = " << covar << endl;  
}
```

二维直方图的Draw()函数  
有很多选项, 请自行选择



运行: 在命令提示行下 root -l ex35.C  
或在ROOT环境下 .x ex35.C

# 小结

---

- ROOT简介
  - C++，面向对象，实验数据处理的强大工具
- 安装与登录以及体验
- 运行ROOT脚本
- 数学函数，画布，直方图，随机数，散点图，舍选法等等
  - TF1, TCanvas, TH1F, gRandom, TH2F
- 新建root文件，查看root文件
  - TFile

# 练习

1. 写一个ROOT脚本, `ex3_gaus.C`, 产生高斯分布, 区间 $(-6,6)$ , 分30个bin, 比较不同的参数的分布。  
参数组合为:  $(\text{mean}, \text{sigma}) = (0,1), (0,2), (1,1), (1,2)$ , 把这4个分布画在同一个图中进行比较。  
**hint: 高斯分布用 `gRandom->Gaus(mean,sigma)` 产生。**
2. 写一个ROOT脚本, `ex3_pdf.C`, 作4个直方图, 分别产生10000事例的 `Gauss, Poisson, Binomial, Landau` 分布。创建画布, 分成 $2*2$ 块, 将4个直方图画在画布的1-4部分。**注意不同分布的参数选择合理性, 比如 `Binomial(ntot,p)`,  $ntot > 0, 0 < p < 1$ 。**  
定义一个二维直方图(`TH2F`), 将随机产生的1000个坐标 $(x,y)$ 填充到直方图中, 其中 $x$ 和 $y$ 都是 $(0,1)$ 之间的均匀分布。画出散点图, 查看 $x$ 和 $y$ 的关联。  
用**hint: 用 `gRandom->Rndm()` 产生均匀分布。**
3. 将练习2中产生的直方图储存到`mypdf.root`文件中。  
将所画直方图的 $x/y$ 轴添加上名称, 不同分布用不同颜色。  
将画布存成`eps`文件和`gif`文件
4. 将例题`ex35.C`中的事例数改为1000, 屏幕打印出关联系数。
5. `cp -r $ROOTSYS/tutorials /projects/$USER`  
运行以下几个文件, 查看ROOT直方图的常用功能如何实现  
`twoscales.C, transpad.C, multicolor.C, logscale.C, hstack.C`
6. 阅读ROOT手册第二章以及第三章(直方图)  
熟悉ROOT语法惯例, 直方图制作的各种参数, 随机数的使用

# 参考资料

---

- ROOT手册第2章，第3章
- <http://root.cern.ch>
- <http://root.cern.ch/root/Reference.html>
- <http://root.cern.ch/root/Tutorials.html>
- <http://root.cern.ch/root/HowTo.html>
- \$ROOTSYS/tutorials中的各个例子