Table of Contents

Graficas en ROOT

Archivos ascii

Esta parte del tutorial, esta inspirada en un codigo que tuve que hacer para un colega que no tenia ni idea de como empezar con ROOT. Basicamente, el necesitaba combinar en un solo grafico, datos guardados en formato ascii. Estos datos, provenian de distintas fuentes y por ende, ademas de las coordenadas X-Y de cada punto, se tenia algunas veces errores simetricos en X-Y, errores asimetricos, errores solo en Y, o sin errores.

Aqui doy una descripción del código que escribi y que ustedes pueden bajar haciendo click aqui. Primero dividire en problema en las funciones que realizan la lectura de los datos y que devuelven la grafica correspondiente.

Formato de entrada : 6 columnas

Los datos vienen en 6 columnas (x, y, xerr1, xerr2, yerr1, yerr2) donde los errores son asimetricos. El correspondiente tipo de grafico es TGraphAsymmErrors:

```
TGraphAsymmErrors * readTypeOne( const char * fname)
{

  ifstream in;
  in.open(fname); //

  double en, flux, enerr1, enerr2, fluxerr1, fluxerr2; //we read 6 columns of data

  TGraphAsymmErrors * graphT1 = new TGraphAsymmErrors();

  int i = 0;

  while ( in.good() )
  {

    in >> en >> flux >> enerr1 >> enerr2 >> fluxerr1 >> fluxerr2;

    graphT1->SetPoint( i , en, flux);
    graphT1->SetPointError( i, enerr1, enerr2, fluxerr1, fluxerr2);

    ++i;

  }

  std::cout << "Total points read: " << i << std::endl;

  in.close();

  return graphT1;

}
```

Formato de entrada : 4 columnas

Los datos vienen en 4 columnas (x, y, xerr, yerr) donde los errores son simetricos. El correspondiente tipo de grafico es TGraphErrors:

```
TGraphErrors * readTypeTwo( const char * fname)
{

  ifstream in;
```

```
 5  in.open(fname); //
 6
 7  double en, flux, enerr, fluxerr;
 8
 9  TGraphErrors * graphT2 = new TGraphErrors();
10
11  int i = 0;
12
13  while ( in.good() )
14  {
15
16    in >> en >> flux >> enerr >> fluxerr; //we read 4 columns of data
17
18    graphT2->SetPoint( i , en, flux);
19    graphT2->SetPointError( i, enerr, fluxerr);
20
21    ++i;
22
23  }
24
25  std::cout << "Total points read: " << i << std::endl;
26
27  in.close();
28
29  return graphT2;
30
31}
```

Formato de entrada : 3 columnas

Los datos vienen en 3 columnas (x, y, yerr) donde los errores son simetricos. Es decir, no tenemos informacion del error en x o simplemente no existe. Manejaremos el mismo tipo que en el caso anterior, es decir TGraphErrors pero introduciremos un Xerror = 0.0:

```
1TGraphErrors * readTypeThree( const char * fname)
2{
3
4  ifstream in;
5  in.open(fname); //
6
7  double en, flux, fluxerr;
8
9  TGraphErrors * graphT3 = new TGraphErrors();
10
11  int i = 0;
12
13  while ( in.good() )
14  {
15
16    in >> en >> flux >> fluxerr; //we read 3 columns of data
17
18    graphT3->SetPoint( i , en, flux);
19    graphT3->SetPointError( i,  0.0, fluxerr);
20
21    ++i;
22
23  }
24
25  std::cout << "Total points read: " << i << std::endl;
26
27  in.close();
28
29  return graphT3;
30
31}
```

Formato de entrada : 2 columnas

Los datos vienen en 2 columnas (x, y) no hay ningun tipo de error. El correspondiente tipo de grafico es el mas sencillo de todos, un TGraph:

```
TGraph * readTypeFour( const char * fname)
{

  ifstream in;
  in.open(fname); //

  double en, flux;

  TGraph * graphT4 = new TGraph();

  int i = 0;

  while ( in.good() )
  {

    in >> en >> flux; //we read 2 columns of data

    graphT4->SetPoint( i , en, flux);

    ++i;

  }

  std::cout << "Total points read: " << i << std::endl;

  in.close();

  return graphT4;

}
```

Script final

Finalmente ponemos todo esta teconologia para leer los archivos y producir la grafica deseada. Algunas notas:

- Nota 1: Usameremos un contenedor propio de ROOT, una [[http://root.cern.ch/root/html/TList.html][TList]. Este nos permitira guardar todos los distintos graficos, sin importar de que tipo son. En ROOT todos estos objetos heredan de una clase comun a todo objeto en ROOT, un TObject. En este caso en particular, crearemos un apuntador a una TList llamado **allgraphs**. La logica es la siguiente:
  - Declaramos un apuntador al grafico que requerimos. La funcion correspondiente --segun el formato de los datos de entrada-- abrira el archivo y devolvera la grafica.
  - Usando el metodo Add(XX) adicionamos la grafica al contenedor.

- Nota 2: Todas las funciones descritas arriba, reciben como argumento, el camino y el nombre del archivo que contiene los datos.

- Nota 3: Necesitamos de un TCanvas para pintar nuestras multiples graficas

- Nota 4: Adicionamos todas las opciones necesarias: ejes en escala logaritmica. Tambien, necesitamos que cada grafica se dibuje con alguna caracteristica que la distinga: color, linea, etc . Por ejemplo, las opciones que podemos dar a la linea que une los puntos de la grafica son TAttLine

- Nota 5: Despues de asignar todas las opciones, realizamos un ciclo sobre los contenidos del

contenedor **allgraphs** y dibujar cada uno de ellos en el Canvas.

```
void makePlots() {

 //. Plots of separate data and different data formats
 //.
 //. The available data formats and their matching Graph objects are:
 //. 6 columns data (x,y, x1, x2, y1, y2) == TGraphAsymmErrors
 //. 4 columns data (x,y, xerr, yerr) == TGraphErrors
 //. 3 columns data (x,y, yerr) == TGraphErros ( xerr = 0.0)
 //. 2 columns data (x,y) == TGraph
 //.
 //....


 int ndataset = 0;
 TList * allgraphs = new TList(); //this is a ROOT container. It will store all of your Gra

 // 1. 6 columns data

 TGraphAsymmErrors * graphT1 =  readTypeOne("1959-650-1ES-data/data-set-n6-MAGIC-uncorrecte
 allgraphs->Add( graphT1 );
 ndataset++;

 graphT1 =  readTypeOne("1959-650-1ES-data/data-set-n7-MAGIC-corrected-absorption-by-IR.txt
 allgraphs->Add( graphT1 );
 ndataset++;

 graphT1 =  readTypeOne("1959-650-1ES-data/data-set-n8-historical-TeV-region.txt");
 allgraphs->Add( graphT1 );
 ndataset++;

 // 1. 3 columns data

 TGraphErrors * graphT3 = readTypeThree("1959-650-1ES-data/data-set-n2-optical-UV-swift.txt
 allgraphs->Add( graphT3 ); //Append to your List this Graph
 ndataset++;

 graphT3 = readTypeThree("1959-650-1ES-data/data-set-n3-may24-max.txt");
 allgraphs->Add( graphT3 ); //Append to your List this Graph
 ndataset++;

 graphT3 = readTypeThree("1959-650-1ES-data/data-set-n4-may19-min.txt");
 allgraphs->Add( graphT3 ); //Append to your List this Graph
 ndataset++;

 graphT3 = readTypeThree("1959-650-1ES-data/data-set-n5-avg-suzaku-swift-may24-29.txt");
 allgraphs->Add( graphT3 ); //Append to your List this Graph
 ndataset++;

 // 1. 2 columns data

 TGraph * graphT4 = readTypeFour("1959-650-1ES-data/data-set-n1-historical-low-region.txt")
 allgraphs->Add( graphT4 ); //Append to your List this Graph
 ndataset++;

 graphT4 = readTypeFour("1959-650-1ES-data/data-set-n9-optical-UV-ground.txt");
 allgraphs->Add( graphT4 ); //Append to your List this Graph
 ndataset++;

 graphT4 = readTypeFour("1959-650-1ES-data/log-parabolas-fit.txt");
 allgraphs->Add( graphT4 ); //Append to your List this Graph
 ndataset++;

 std::cout << "Total dataset read " << ndataset << " " << std::endl;

```

```
65  //. End of reading and creating Graphs from data files
66  //.
67  ////////////////////////////////////////////////////////////////////////////////
68  //.
69  //. Plot each Dataset on a sigle Canvas (all combined)
70
71  TCanvas * canvas = new TCanvas("Plot1", "Canvas for plot 1", 94, 262,700, 502 );
72  canvas->SetFillColor(10);
73
74  //If you need log scale on both axis - uncomment the following lines
75  //canvas->SetLogx();
76  //canvas->SetLogy();
77
78  //Set the minimum and maximum for the whole graph
79  //Min a max for Axis
80  float xmin =  10.0;
81  float xmax =  28.0;
82  float ymin = -13.0;
83  float ymax =  -9.0;
84
85  //Datasets options (Markers: style, color, size) : You can also do it by hand using the in
86  int style[15];
87  int color[15];
88  float size[15];
89
90  //For dataset No1
91  style[0] = 25;
92  color[0] = 2;
93  size[0]  = 1.0;
94
95  //For dataset No2
96  style[1] = 26;
97  color[1] = 4;
98  size[1]  = 1.0;
99
100 //For dataset No3
101 style[2] = 24;
102 color[2] = 38;
103 size[2]  = 1.0;
104
105 //For dataset No4
106 style[3] = 20;
107 color[3] = 4;
108 size[3]  = 0.5;
109
110 //For dataset No5
111 style[4] = 24;
112 color[4] = 6;
113 size[4]  = 0.8;
114
115 //For dataset No6
116 style[5] = 26;
117 color[5] = 2;
118 size[5]  = 0.8;
119
120 //For dataset No7
121 style[6] = 7;
122 color[6] = 2;
123 size[6]  = 1.0;
124
125 //For dataset No8
126 style[7] = 25;
127 color[7] = 38;
128 size[7]  = 1.0;
129
130 //For dataset No9
131 style[8] = 26;
```

```
132  color[8] = 7;
133  size[8]  = 1.0;
134
135  //For dataset No10
136  style[9] = 1;
137  color[9] = 1;
138  size[9]  = 1.0;
139
140  // Loop over all Graphs and draw them in the Canvas
141  // Min,Max for X and Y axis are set on the first Graph that is plotted
142
143  ndataset = allgraphs->GetSize(); //Get the ndatasets from the size of the List
144
145  // Loop now over the List using the index k
146
147  for(int k=0; k < ndataset; ++k) {
148
149    if( k == 0 ) {
150      //this is our first graph and it is special (to define axis min,max)
151      ((TGraph*)allgraphs->At(k))->SetMinimum(ymin);
152      ((TGraph*)allgraphs->At(k))->SetMaximum(ymax);
153      ((TGraph*)allgraphs->At(k))->Draw("AP");
154      ((TGraph*)allgraphs->At(k))->GetXaxis()->SetLimits(xmin, xmax);
155      //set the color options
156      ((TGraph*)allgraphs->At(k))->SetMarkerStyle( style[k] );
157      ((TGraph*)allgraphs->At(k))->SetMarkerSize( size[k] );
158      ((TGraph*)allgraphs->At(k))->SetMarkerColor( color[k] );
159      ((TGraph*)allgraphs->At(k))->SetLineColor( color[k] );
160      ((TGraph*)allgraphs->At(k))->Draw("AP"); // Draw option AP A=draw axis P=draw a marker
161
162    }
163    else {
164      ((TGraph*)allgraphs->At(k))->SetMarkerStyle( style[k] );
165      ((TGraph*)allgraphs->At(k))->SetMarkerSize( size[k] );
166      ((TGraph*)allgraphs->At(k))->SetMarkerColor( color[k] );
167      ((TGraph*)allgraphs->At(k))->SetLineColor( color[k] );
168
169      ((TGraph*)allgraphs->At(k))->Draw("P"); // since we have already plotted the axis on t
170      if( k == 9 ){
171        ((TGraph*)allgraphs->At(k))->SetLineWidth(2);
172        ((TGraph*)allgraphs->At(k))->Draw("C");
173      }
174
175    }
176
177  }
178
179  /// All done!
180
181}
```

-- AndresOsorio - 27-Jun-2012

This topic: Main > RootGraphicsAO
Topic revision: r1 - 2012-06-27 - AndresOsorio

Copyright &© 2008-2019 by the contributing authors. All material on this collaboration platform is the property of the contributing authors.
Ideas, requests, problems regarding TWiki? Send feedback