

Computational Physics Lab

Data Analysis Frameworks: Project 9

03/04/2008

Project 9

1 Part 1

- Creating a ROOT macro
- Executing a macro with ROOT
(Fitting a distribution with Breit-Wigner and Polynomial)
- Modifying a macro and improving graph formatting
- Interacting formatting and fitting
- Save image of graph

2 Part 2

- Repeat Part 1 using a 2-Gaussian fit function

3 Part 3

- Convert macro to stand-alone program

4 Part 4

- Document project on your Computational Physics web site

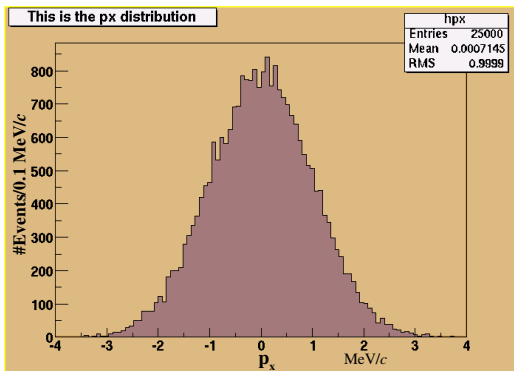
Histograms

Project 9

Part 1
Part 2
Part 3
Part 4

Histograms are graphs representing statistical distributions

- Projections from large volumes of data
- Heights of the bars represent observed frequencies

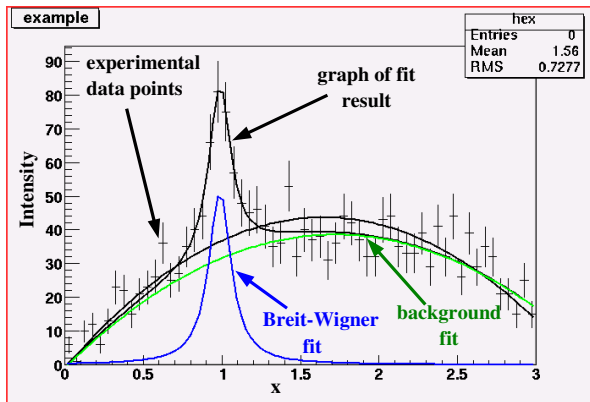


Fitting Histogram Data

Project 9

- Part 1
- Part 2
- Part 3
- Part 4

Breit-Wigner Resonance plus a polynomial background



$$\text{BreitWigner}(x) = \frac{1}{2\pi} \frac{I_o \Gamma}{(x - M_o)^2 + 0.24 \Gamma_o^2}$$

$$\text{Background}(x) = a_o + b_o x + c_o x^2$$

parameters of the fit: $a_o, b_o, c_o, I_o, M_o, \Gamma_o$

Part 1: ROOT Macro Creation

Project 9

Part 1

Part 2

Part 3

Part 4

Create a text file called “**histoFit.C**” with editor

Add the following user-defined functions:

```
double background(double *x, double *par) {  
    return par[0] + par[1]*x[0] + par[2]*x[0]*x[0];  
}
```

```
double breitWigner(double *x, double *par) {  
    return ( 0.5*par[0]*par[1] / TMath::Pi() ) /  
        TMath::Max( 1e-10, (x[0] - par[2]) * (x[0] - par[2]) +  
        0.24*par[1]*par[1] ); }  

```

```
double fitFunction(double *x, double *par) {  
    return background(x, par) + breitWigner(x, &par[3]);  
}
```

Macro: Add a “doFit()” Function

Project 9

- Part 1
- Part 2
- Part 3
- Part 4

```
void doFit() { // Get the text for doFit() at ~/crede/root/doFit.txt
    const int nBins = 60;
    double data[60] = { 6, 1, 10, 12, 6, 13, 23, 22, 15, 21, 23, 26, 36,
                       25, 27, 35, 40, 44, 66, 81, 75, 57, 48, 45, 46, 41, 35, 36,
                       53, 32, 40, 37, 38, 31, 36, 44, 42, 37, 32, 32, 43, 44, 35,
                       33, 33, 39, 29, 41, 32, 44, 26, 39, 29, 35, 32, 21, 21, 15,
                       25, 15 };

    // Look up TH1F at http://root.cern.ch/root/Reference.html
    TH1F *histogram = new TH1F("histo","title", 60, 0, 3);

    for (int i = 0; i < nBins; i++) {
        histogram->SetBinContent(i+1, data[i]);
        histogram->SetBinError(i+1, TMath::Sqrt(data[i]));
    }

    // Look up TF1 at ROOT Refrence page
    TF1 *fitFcn = new TF1("fitFcn", fitFunction, 0, 3, 6);

    // Try fitting wo/ starting values for the parameters
    histogram->Fit("fitFcn");
}
```

Execute Macro w/ CINT/ROOT

Project 9

- Part 1
- Part 2
- Part 3
- Part 4

gluon 25% root

```
*****
*
*   WELCOME to ROOT
*
*   Version  5.14/00   14 Dec 2006
*
*   You are welcome to visit our Web site
*   http://root.cern.ch
*
*****
```

Compiled for linux with thread support.

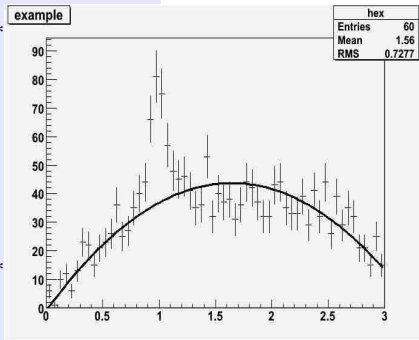
CINT/ROOT C/C++ Interpreter version 5.16.16, Nov 2004

Type ? for help. Commands must be C++ statements.

Enclose multiple statements between { }.

root [0] **.L histoFit.C**

root [1] **doFit()**



Macro: Improving the Fitting

Project 9

Part 1

Part 2

Part 3

Part 4

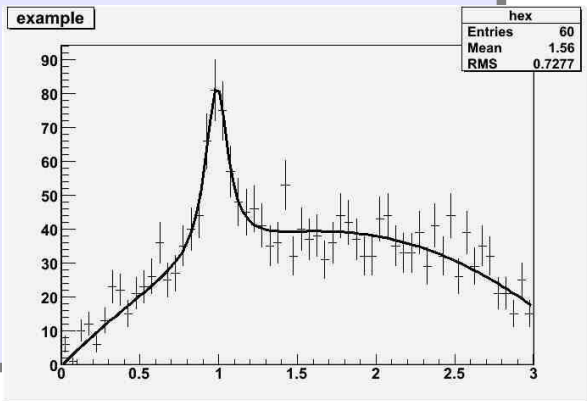
```
// Add these commands to the doFit() function
```

```
// now set start values
```

```
fitFcn->SetParameter(4,0.2); //width
```

```
fitFcn->SetParameter(5,1); // peak
```

```
histogram->Fit("fitFcn","V+");
```



Macro: Improving the Graph

Project 9

- Part 1
- Part 2
- Part 3
- Part 4

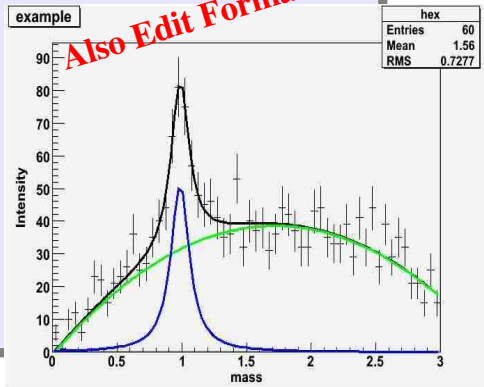
```
// Add these commands to the end of the doFit() function
// improving the figure
```

```
TF1* backFcn = new TF1("backFcn",
                        background,0,3,3);
backFcn->SetLineColor(3);
TF1* bwFcn = new TF1("bwFcn", brietWigner,0,3,3);
bwFcn->SetLineColor(4);
```

```
double par[6];
fitFcn->GetParameters(par);

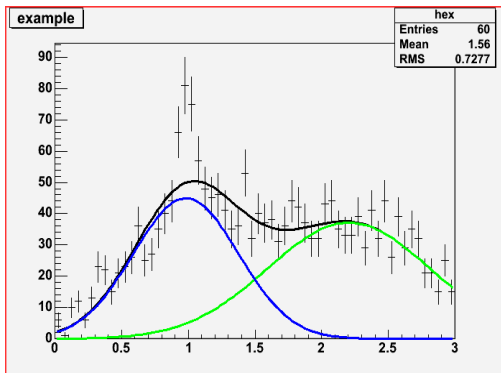
backFcn->SetParameters(par);
backFcn->Draw("same");

bwFcn->SetParameters(&par[3]);
bwFcn->Draw("same");
```



Part 2: Fit Histogram to Gaussian plus Gaussian

$$G(A, \sigma, \bar{x}) = \frac{A}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\bar{x})^2}{2\sigma^2}}$$



Part 3: Making a Stand-Alone Application

Get the Makefile

→ Type: `cp /export/home/crede/root/Makefile .`

Create fit.cc source

- 1 Type: `cp histoFit.C fit.cc`
- 2 Add includes & main() function
- 3 Modify source code to adhere to coding conventions

Add at Beginning:

```
#include "TApplication.h"  
#include "TH1.h"  
#include "TF1.h"  
#include "TMath.h"  
#include <iostream>  
  
using namespace std;
```

Add at End:

```
int main(int argc, char **argv) {  
    TApplication theApp("App", &argc, argv);  
    doFit();  
    theApp.Run();  
}
```

Part 4: Document Project 9

Project 9

Part 1

Part 2

Part 3

Part 4

Post Project 9 to your computational physics web site. Create a `html` page for this project. Create a link from your main project web page to this `html` page. It should include the following heading information: project title, project number, your name, & today's date.

The main content of this page should include the following:

- A short description of the project
- A link to the macro code for part 2
- A text region which contains the actual macro code for part 2
- An image for a formatted plot for part 3
- A link to the stand-alone code for part 3
- A text region which contains the course code for part 3

For text regions use the `html` object tag; example:

```
<object width="600" height="400" type="text/plain"  
data="yourProgram.cc" border="0"></object>
```