Звіт до лабораторної роботи з аналізу данних

Дирів Олександр ФВЕ

Мета: З використанням ROOT-класу TMinuit у мові Python написати програму для підгонки даних модельною кривою.

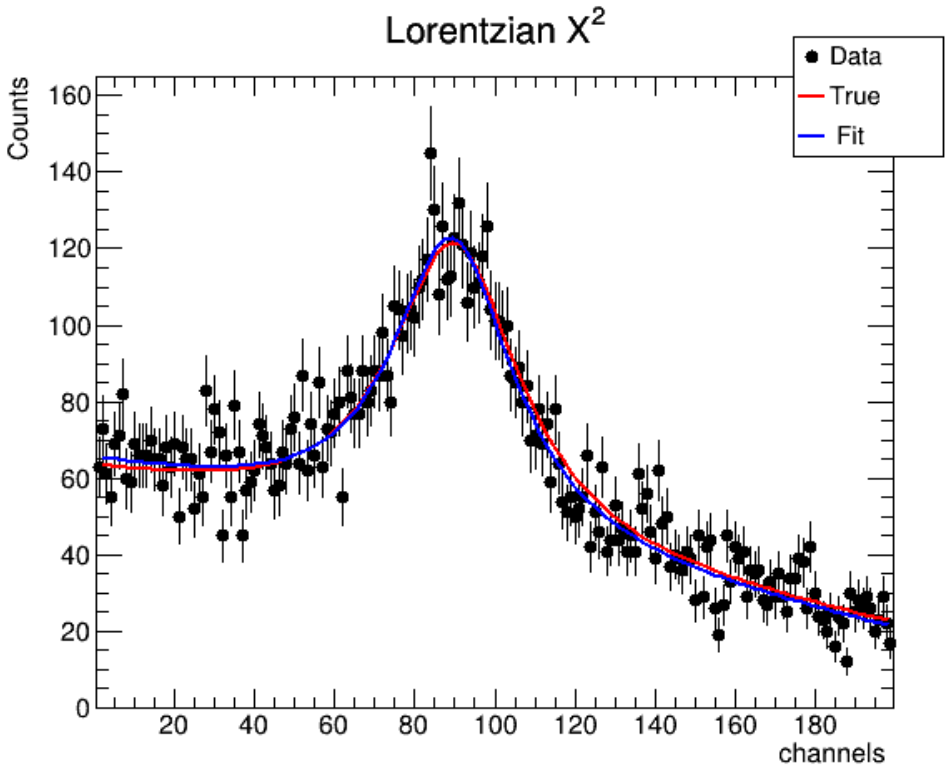
Хід роботи

Для лоренціана з параметрами моделі: *Area*=5000, M_0=90, $\wp$=20, і лінійного фону $y = A_{Bg}x + B_{Bg}$ з параметрами A_{Bg} = -0.2, B_{Bg} = 60, згенерований спектр згідно до Пуасону.

1.У результаті підгонки $|^2$

Отримано наступні параметри:

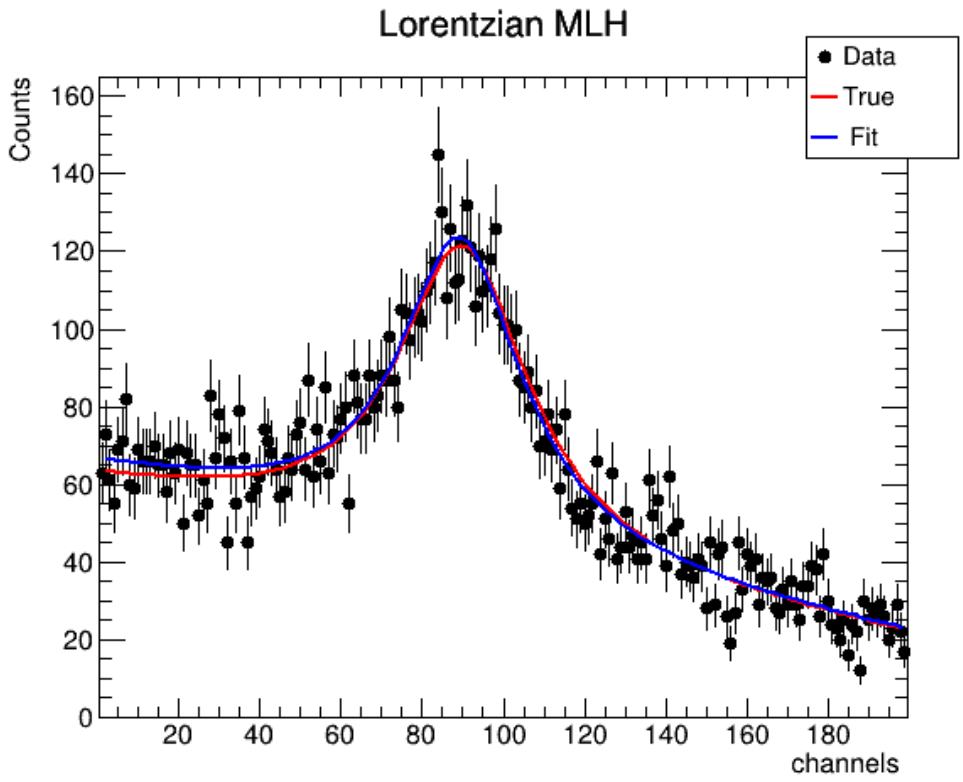
1	Area	4.65687e+03	2.39150e+02	-2.33554e+02	2.45412e+02
2	g	1.85552e+01	1.06719e+00	-1.03522e+00	1.10248e+00
3	x0	8.94039e+01	6.05781e-01	-6.07247e-01	6.04852e-01
4	a_bg	-2.16886e-01	9.30523e-03	-9.28297e-03	9.33012e-03
5	b_bg	6.24585e+01	1.68117e+00	-1.69799e+00	1.66635e+00



1.У результаті підгонки Binned Maximum LikeLihood

Отримано наступні параметри:

1	Area	-4.58096e+03	3.42697e+02	-3.56834e+02	3.30092e+02
2	g	-1.84355e+01	1.55369e+00	-1.62781e+00	1.48573e+00
3	x0	8.93505e+01	8.61608e-01	-8.65661e-01	8.59297e-01
4	a_bg	-2.14979e-01	1.33674e-02	-1.33920e-02	1.33836e-02
5	b_bg	6.36856e+01	2.40507e+00	-2.42680e+00	2.39269e+00



Висновки:

Вдалося непогано підібрати параметри обома способами підгонки.

```
import sys , ROOT, time
from math import *
import numpy as np
import ctypes

def Lorentz_linear(x, params):
    Area, Gamma, x0, A, B = params[0],params[1],params[2],params[3],params[4],
    return Area * ((1/np.pi)*Gamma/(np. square (x-x0)+np.square (Gamma)))+A*x+B

params = [5000, 20., 90., -0.2, 60.]
tArea, tGamma, tx0, tA, tB = 5000, 20., 90., -0.2, 60.0
rnd = ROOT. TRandom3 ()
nChan = 200
nPar = 5
xExpt = np. linspace (1, nChan, nChan)
Tot = [Lorentz_linear(x, params) for x in xExpt]
dx = xExpt [1] - xExpt[0]
yExpt = np.zeros_like(xExpt)
for chan in range(nChan):
    yExpt[chan] = rnd.Poisson(Tot[chan])
indPos = yExpt > 0
nPos = indPos.sum()
print (indPos)
print (nPos)

def FCN(npar, gin, f, par, iflag):
    global valFCN
    yTheor = np.array([Lorentz_linear(x, par) for x in xExpt])
    indPos = yExpt > 0
    arrayFCN = np.square (yExpt [indPos] - yTheor [indPos])/yExpt[indPos]
    valFCN = np.sum(arrayFCN)
    f.value = valFCN


# Now ready for minimization step
ierflg = ctypes.c_int(0)
minuit = ROOT.TMinuit(nPar)
minuit . SetPrintLevel (1)
minuit .SetFCN(FCN)
errordef = 1.

minuit.DefineParameter(0, 'Area', 13, 1e-2, 0., 0.)
minuit.DefineParameter (1, 'g', 10., 1e-4, 0., 0.)
minuit.DefineParameter (2, 'x0', 100, 1e-4, 0., 0.)
minuit.DefineParameter (3, 'a_bg', -0.3, 1e-4, 0., 0.)
minuit.DefineParameter (4, 'b_bg', 50, 1e-4, 0., 0.)

minuit .mncomd("SET ERR "+str(errordef), ierflg);
minuit .mncomd("SET STR 1", ierflg); # strategy (1)
maxIter=10000
tolerance =1.e-8
minuit .mncomd("MIGRAD "+str(maxIter)+' '+str(tolerance), ierflg)

minuit .mncomd("SET STR 2", ierflg); # strategy (2)
minuit .mncomd("MIGRAD "+str(maxIter)+' '+str(tolerance), ierflg)
minuit .mnmnos() # Minos

ndf = nPos-minuit.GetNumFreePars()
print (' chi2/ndf: ' , valFCN, '/' , ndf)
parFit = np.zeros(nPar)
errFit = np.zeros(nPar)
valPar , errPar = ctypes.c_double(0.0), ctypes.c_double(0.0)
for iPar in range(nPar):
    minuit .GetParameter(iPar,valPar , errPar )
    parFit [ iPar ] = valPar.value
    errFit [ iPar ] = errPar.value
Area1, gamma1, x01, a1, b1 = parFit[0], parFit[1], parFit[2], parFit[3], parFit[4]
```

```
# histogram decoration
ROOT.gStyle.SetOptStat(0)
h = ROOT.TH1F('histLorentz','Lorentzian X^{2}',nChan-1,xExpt-dx/2.)
h.SetMinimum(0)
h.SetMarkerColor(ROOT.kBlack)
h. SetLineColor(ROOT.kBlack)
h.SetMarkerStyle(20)
h.SetMarkerSize(1)
h.GetXaxis() . SetTitle ( ' channels ' )
h.GetYaxis() . SetTitle ( 'Counts')

fTrue = ROOT.TF1('fTrue','[0]*(([1]/3.14159265359)/((x - [2])*(x - [2]) + [1]*[1])) + [3] * x + [4]',xExpt[0],xExpt[-1])
fTrue . SetLineColor(ROOT.kRed)
fTrue .SetLineWidth(2)
fTrue .SetParameters(tArea, tGamma, tx0, tA, tB)


fFit = ROOT.TF1('fFit','[0]*(([1]/3.14159265359)/((x - [2])*(x - [2]) + [1]*[1])) + [3] * x + [4]',xExpt[0],xExpt[-1])
fFit . SetLineColor(ROOT.kBlue)
fFit .SetParameters(Area1, gamma1, x01, a1, b1)

hLeg = ROOT.TLegend(0.8,0.8,0.95,0.95)
hLeg.AddEntry(h,'Data', 'p')
hLeg.AddEntry(fTrue,'True' , ' l ' )
hLeg.AddEntry(fFit, ' Fit ' , ' l ' )
hLeg. SetFillColor (0)
for chan in range(nChan):
    h.SetBinContent(chan+1,yExpt[chan])
cMnt = ROOT.TCanvas('cMnt','chi2',600,500)
h.Draw('e')
fTrue.Draw('same&l')
ROOT.gPad.SetTicks(1,1)
ROOT.gPad.RedrawAxis()
fFit.Draw('same&l')

hLeg.Draw()
ROOT.gPad.Update()
```