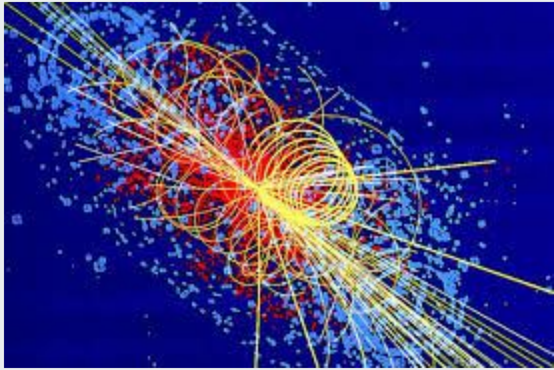
HIGH ENERGY PHYSICS//WORK EXPERIENCE:

Research:

- Analysing Data
- Representing Data
- Collaboration



Teaching:

- Laboratory
- Lectures

Statistics in HEP:

- Using data collected in LHC: CMS-
    - Studying the standard model, extra dimensions and dark matter
    - Detector- around a solenoid(superconducting coil producing 4 teslas)
- LHC collides protons near the velocity of light, has collides at four different point in the lhc
- CMS acts as a high speed camera taking '3D photographs' at all directions 40 million/second
- CMS:
    - 15m*21m containing several concentric rings to help take capture images
    - Very accurately measures muons
    - Contains the most powerful solenoid ever made
- Bending Particles - identifying charged particles and measure momentum of particles
- Identifying Tracks- silicon tracker, interacts electromagnetically with the tracker
- Measuring Energy- ecal:measuring energy of e and p, hcal: hadrons
- Detecting Muons- measured through sub-detectors measuring momentum in the superconducting coils and the muon chambers

https://cms.cern/detector

Data Acquisition in LHC:

- Triggering:
    - When cms is at its peak ~1 billion proton-proton interactions/second,
    - A ‘trigger’ is used to filter through the data to search for potential phenomena
    - Level 1 - a quick and wholly automatic process that which searches for simple signs of interesting physics
    - Higher level triggers will collect data from all parts of the detector to recreate the event
    - With a lower event rate a more detailed analysis is permitted
- Data analysis:
    - Data that has passed the triggering stages and been stored on tape is duplicated using the Grid for various different sites around the world
        - Measurements made on standard model particles
        - Searching for events with missing large transverse energy
        - Studying the kinematics of pais of particles produced by the decah of the parent
        - Looking at the jets of the particles to see how the quarks and gluons have interacted

Example(I)//Without programming:

"In the upper atmosphere protons from space collide with protons in the atmosphere:
$p + p \rightarrow p + n + \pi^+$
The pion then decays into a muon and a muon-neutrino:
$\pi^+ \rightarrow \mu^+\nu_\mu$
Assuming a neutrino mass of 0, can you calculate the momentum of the muon in the pion rest frame ?"

Four Vectors(an object with four components):

_The four components are E, Px, Py, Pz:

- Pion: M_{pi} , 0, 0, 0
- Muon:P_{mu}, $-P_{mu}$, 0 , 0
- Muon-Neutrino: sqrt($P_{mu}^2 + M_{mu}^2$)
- $2P_{mu} = M_{pi}^2 + M_{mu}^2$

The momentum of the muon in the pion rest frame is 30 Mev.

Finding a Higgs//With Programming:

Data recorded: 2012-Jul-26 04:29:42.397495 GMT

- Two photons with their measurements: energy, transverse energy, eta and phi(the spherical coordinates)
- The invariant mass should be 125 MeV, a higgs boson- calculating with four vectors, using python to code it.
- i) Convert the spherical coordinates into Px, Py, Pz:

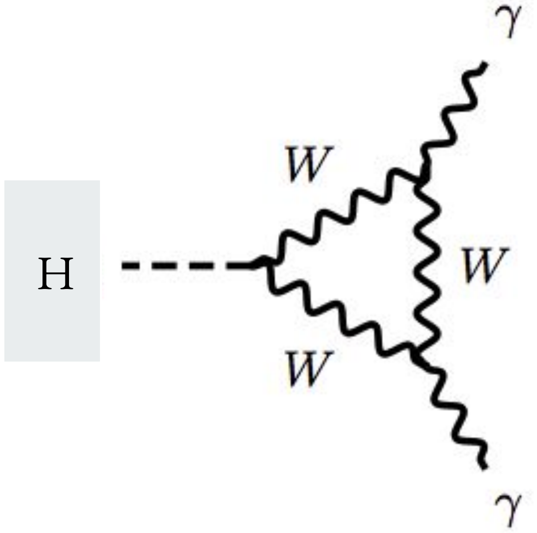
  +Px = pTcos(phi)

  +Py = pTsin(phi)

  +Pz = pTsinh(eta)

  ii) Add both four vectors

  iii)mass^2 = energy^2 - momentum^2
- The invariant mass we calculated was 124.59 GeV (125 at 3s.f.)



```
def px(_Pt, _phi):
    return _Pt*(math.cos(_phi))
def py(Pt, phi):
    return Pt*(math.sin(phi))
def pz(Pt, eta):
    return Pt*(math.sinh(eta))
def pt(px,py):
    return(math.sqrt(px**2 + py**2))

def gamma_momentum(_E, _Et, _eta, _phi):
    _px = px(_Et, _phi)
    _py = py(_Et, _phi)
    _pz = pz(_Et, _eta)

    return (math.sqrt(_px**2 + _py**2 + _pz**2))

def p(px,py,pz):
    return (math.sqrt(px**2 + py**2 + pz**2))



def main():
    #photon 1
    E  = 71.4363
    Et= 64.1091
    eta = 0.473667
    phi = -0.815133
    Pt = Et

    _px = px(Pt, phi)
    _py = py(Pt, phi)
    _pz = pz(Pt, eta)
    gamma1 = gamma_momentum(E, Et, eta, phi)
    print(E,_px,_py,_pz)

    #photon 2
    E1 = 58.6338
    _Et = 58.6038
    eta1 = 0.0320185
    phi1 = 2.58568
    Pt1 = _Et
    __px = px(Pt1, phi1)
    __py = py(Pt1, phi1)
    __pz = pz(Pt1, eta1)

    print(E1,__px,__py,__pz)
    E_higgs = E + E1
    px_higgs = _px + __px
    py_higgs = _py + __py
    pz_higgs = _pz + __pz

    p_higgs = p(px_higgs,py_higgs,pz_higgs)

    m = math.sqrt(E_higgs**2 - p_higgs**2)

    print(m)

    print("I am done here.")


if __name__ == '__main__':
    main()
```

Displaying Data - Distributions:

- With the data received, the diphoton mass, it is necessary to display in such a way to make it more easily eligible.
- Displaying the data was far more difficult than calculating the higgs mass as a result of trying to find an appropriate function to fit the histogram
- Functions including: gaussian, crystal ball,which had been our most precise fit yet is too narrow, a breit-wigner and a double gaussian
- Despite many attempts an appropriate fit has not yet been found

Crystal Ball:

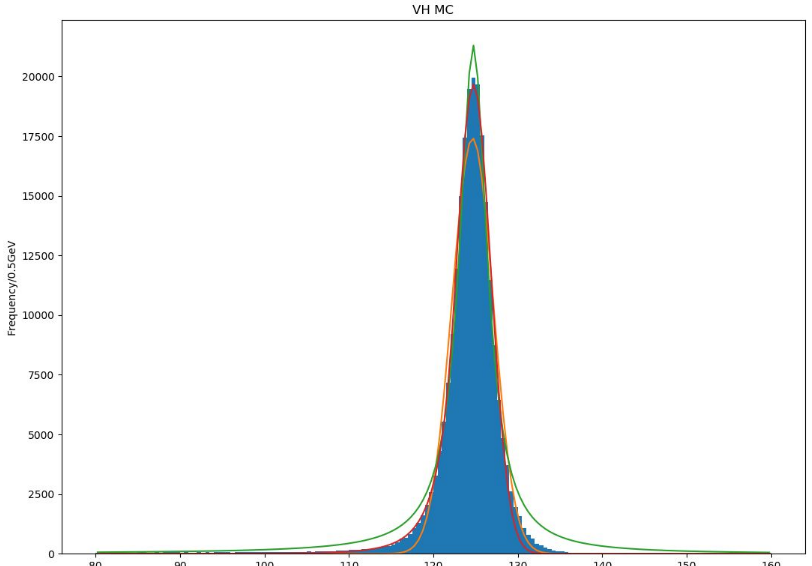
$$f(x;\alpha,n,\bar{x},\sigma) = N \cdot \begin{cases} \exp\left(-\frac{(x-\bar{x})^2}{2\sigma^2}\right), & \text{for } \frac{x-\bar{x}}{\sigma} > -\alpha \\ A \cdot \left(B - \frac{x-\bar{x}}{\sigma}\right)^{-n}, & \text{for } \frac{x-\bar{x}}{\sigma} \leqslant -\alpha \end{cases}$$

$$A = \left(\frac{n}{|\alpha|}\right)^n \cdot \exp\left(-\frac{|\alpha|^2}{2}\right),$$

$$B = \frac{n}{|\alpha|} - |\alpha|,$$

$$N = \frac{1}{\sigma(C+D)},$$

Red - Crystal Ball
Green - Breit-Wigner
Orange - Gaussian/Normal



```python
import uproot as upr
import matplotlib.pyplot as plt
import getpass
import numpy as np
import scipy as sp
from scipy.optimize import curve_fit
import scipy.optimize as spo
from scipy import optimize
import scipy.stats as sps
from scipy.stats import chisquare

def make_pandas_from_file(inputfile, interesting_bits, show_input=False):

    file_dict = {"Data.root" :"Data",
                 "VH.root" : "vh",
                 "VBF.root" : "vbf",
                 "ggH.root" : "ggH"}

    username = getpass.getuser()
    input_file = upr.open("/home/%s/higgs/%s" %(username, inputfile))

    file_data = input_file["vbfTagDumper/trees/%s_13TeV_VBFDiJet" %(file_dict[inputfile])]
    if show_input:
        print(file_data.show())

    print("There are %s events in the %s file" %(file_data.numentries,inputfile))
    file_data = file_data.pandas.df(interesting_bits)

    return file_data

def gaussian(x, a, mean, sigma):
    return a* sps.norm.pdf(x,mean,sigma)

def breit_wigner(x, a, m,gamma):
    g = sp.sqrt((m**2 + gamma**2) * m**2)
    k = 2 * sp.sqrt(2) * m * gamma * g / (sp.pi * sp.sqrt(m**2 + g))
    return a * k / ((x**2 - m**2)**2 + m**2 * gamma**2)

def crystal_ball(x, a, b, m,  mean, sigma):
    return a * sps.crystalball.pdf(x, b, m, mean, sigma)

def main():
    interesting_bits = ['dipho_mass','dipho_leadPt','dipho_leadEta','dipho_leadPhi']

    vh_mc_data = make_pandas_from_file("VH.root",interesting_bits)


    plt.figure("Fit to MC")
```

```
    ydata, xdata, patches = plt.hist(x=vh_mc_data.dipho_mass, bins=160, range=(80,160))
    xmid = np.float64(0.5*(xdata[1:] + xdata[:-1]))
    ydata = np.float64(ydata)

    a=np.array(xdata, dtype="float64")
    print(a.std())
    starting_mean = 125.0
    starting_sigma = 23.0

    #Gaussian
    popt, pcov = optimize.curve_fit(gaussian, xmid,ydata,sigma=np.sqrt(ydata), p0=[2000, starting_mean,starting_sigma])
    plt.plot(xmid, gaussian(xmid,*popt), label='fit')
    print(popt)
    print(pcov)
    print(gaussian(125.25, 106977, 124.6, 2.45))
    print(gaussian(xmid,*popt))
    #Breit_Weigner
    starting_gamma = 23.0
    starting_mean  = 125.0
    popt, pcov = optimize.curve_fit(breit_wigner, xmid,ydata, p0=[2000, starting_mean,starting_gamma])
    plt.plot(xmid, breit_wigner(xmid,*popt), label='fit')
    print(popt)
    print(pcov)
    #Crystal Ball
    po,pvov =spo.curve_fit(crystal_ball,xmid, ydata,[2000,2,3,125,3])
    plt.plot(xmid, crystal_ball(xmid,*po), label='fit')
    print(crystal_ball(xmid,*po))
    print(xmid)
    print(ydata)
    print(len(crystal_ball(xmid,*po)))

    chisquare,p = sps.chisquare(ydata,crystal_ball(xmid,*po))
    print("Here",chisquare,p)

    plt.xlabel('di photon mass')
    plt.ylabel('Frequency/0.5GeV')
    plt.title('VH MC')
    plt.show(block=False)
    plt.pause(1)
    plt.savefig("diphoton_mc_mass.png")

    input("\n<Hit Enter to Exit Programme>")
    print("I am done here")

if __name__=='__main__':
    main()
```

Radiation Experiment

- Measuring the count rate of a beta source while varying the distance.
- As distance increased , count rate decreased.
- We increased time to account for this and thus give a more valid count rate.
- We accounted for the error on the displacement as we didn't know the exact location of the source within the box.
- As you can see on the y-axis we used count rate*displacement^2. This is done to create a straight line , thus making it easier to see the rate of decay.