Event Reconstruction

Dr. Donglian Xu
(taking all the credit 😮)
Kevin Ghorbani
Conor Kirby
Ben Hokanson-Fasig
William Luszczak
Bunheng Ty

Outline

- What was our Project?
- The Physics Behind our Project
- Nuts and Bolts (description of the process)
- Results of Analysis
- What we Learned

What was our project?

- Goal: Construct an energy estimator for any event (optimised for single cascade) so that we have a function we can use to find an estimated value for visible energy, given the total charge (Qtot) collected in the IceCube detector.
- To find this function we used simulated NuE events (100 GeV - 100 PeV) for which we could find both the energy deposited and the charge collected. We then plotted the energy vs the charge for each event in a 2d histogram.
- Using the histogram we took the mean value of the energy for each value of charge collected, and fitted a polynomial function to those data points.

*Used simulation data from file:

/data/sim/IceCube/2011/filtered/level2/neutrino-generator/10648/00000-00999/
Level2_IC86.2011_nugen_NuE.010648.000316.i3.bz2

The Physics

Why do we see a track for a muon but a cascade for an electron?

The Physics

A reason: The mass

- **Maximum energy transfer** occurs in a "head-on" collision between two particles of masses m and M: and can be expressed as

$$Q_{\max} = \frac{4mME}{(M+m)^2}$$

where E is the kinetic energy of the incident particle.

With light charged particles, $\boldsymbol{m = M}$ and so $\boldsymbol{Q_{max} = E}$.

- The electron collides with a particle of identical mass and thus **large scattering angles are possible**.

- This results in a track that is very tortuous instead of the straight path of a heavy charged particle.

Source: http://ocw.mit.edu/courses/nuclear-engineering/22-01-introduction-to-ionizing-radiation-fall-2006/lecture-notes/energy_dep_elect.pdf

The Physics

Neutral Current(NC) vs. Charge Current(CC) events

CC Charged Current Reaction	$\nu_e + d \rightarrow p + p + e^-$
NC Neutral Current Reaction	$\nu_x + d \rightarrow \nu_x + p + n$

For both types of events, we had to approach data interpretation differently in order to ensure we used the correct visible energy. For the CC interaction we collected data on the lepton and hadron produced in the interaction, but for the NC interaction only the hadron energy was visible

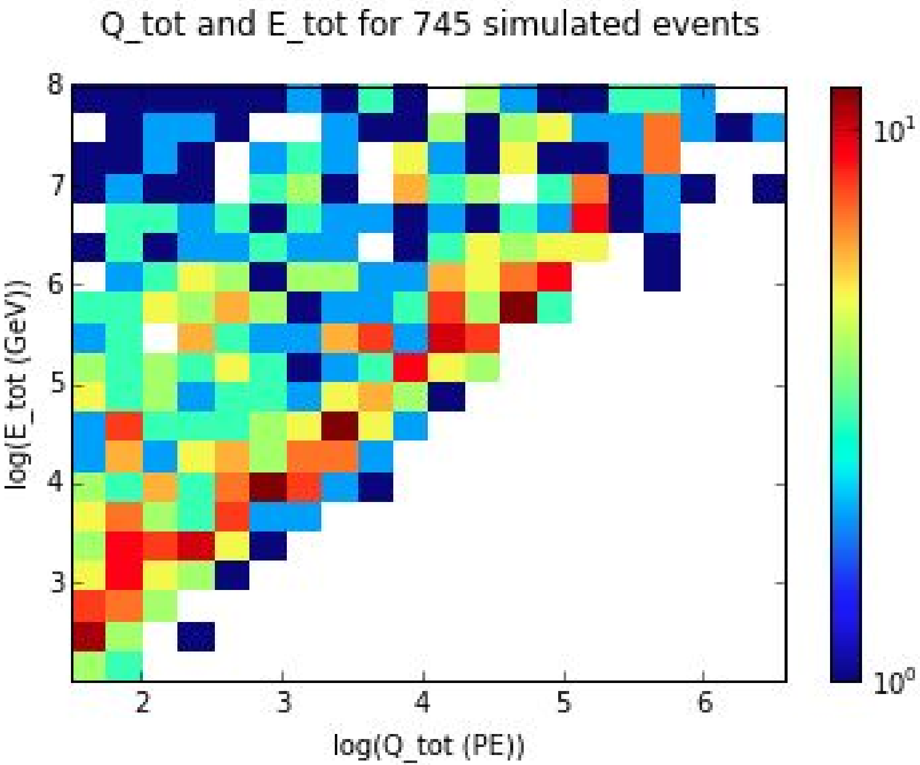
How did we find charge and energy?

- To calculate the charge collected by the DOMs, we pulled the charge data from each PMT pulse in units of PE and summed them for the duration of the event.
- To calculate the energy deposited in the detector by an event, we took any events in a frame that started from neutrinos, summed the energies of any in-ice neutrinos produced by the event, and subtracted off the energies of neutrinos leaving the detector. This allowed us to collect the visible energies from both CC and NC events.

Nuts and Bolts

```
Q_tot = []
E_tot = []
i = 0
for frame in q_frames:
    i += 1
    pulse_frame = dataclasses.I3RecoPulseSeriesMap.from_frame(frame,'OfflinePulses')
    mc_frame = frame['I3MCTree']
    frame_Q = 0
    frame_E = 0
    for pulses in pulse_frame.values():
        for pulse in pulses:
            frame_Q += pulse.charge
    Q_tot.append(frame_Q)
    nu_primary = False
    for particle in mc_frame:
        if particle in mc_frame.primaries and particle.is_neutrino:
            nu_primary = True
        elif particle in mc_frame.primaries:
            nu_primary = False
        if nu_primary and particle.is_neutrino and particle.location_type_string=='InIce':
            if frame_E==0:
                frame_E += particle.energy
            else:
                frame_E -= particle.energy
    E_tot.append(frame_E)
```

Nuts and Bolts



Nuts and Bolts

```
binnum = 100.

binsizey = (np.log10(max(E_tot)))/binnum
binsizex = (np.log10(max(Q_tot)))/binnum

print "binsizex =", binsizex

logenergies =np.nan_to_num([np.log10(i) for i in E_tot])
logcharges = np.nan_to_num([np.log10(i) for i in Q_tot])

print "max logcharges = ",max(logcharges)

xvals = []
yvals = []
for j in range(0,int(binnum+1.)):
        points = []
#       print "bin %s"%(j), "log(Qtot) = ", binsizex*j
        xvals.append((binsizex/2)+binsizex*j))
        bins = [i for i, x in enumerate(logcharges) if binsizex*j<=x<binsizex*(j+1.)]
        for n in bins:
                points.append(logenergies[n])
#       print "log(Energy) = ", np.mean(points)
        yvals.append(np.mean(points))
```

Nuts and Bolts

```
coords = []
for i in range(0,len(xvals)):
        pairs = [xvals[i],np.nan_to_num(yvals)[i]]
        if pairs[1]!=0.:
                coords.append(pairs)

newxvals = [coords[i][0] for i in range(0,len(coords))]
newyvals = [coords[i][1] for i in range(0,len(coords))]

print "max newxvals = ", max(newxvals)


fit = np.polyfit(newxvals,newyvals,3)
p = np.poly1d(fit)
xn = np.linspace(1,7.5,100)

print fit
```

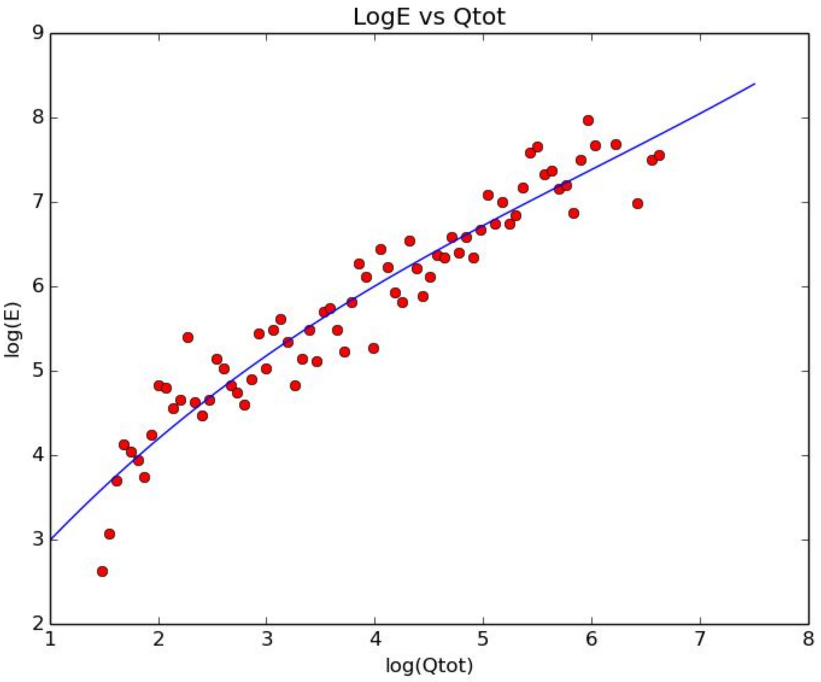
```
#######How good is my fit?
arr = []
for i in newxvals:
        calc = p(i)
        true = i
        arr.append(((calc-true)**2)/true)
chi = sum(arr)
print chi
```

Results

Fit a cubic with 100 bins

fit = [0.00902597 -0.16231309 1.62779889 1.50695517]

$\chi^2 = 91.368931806$

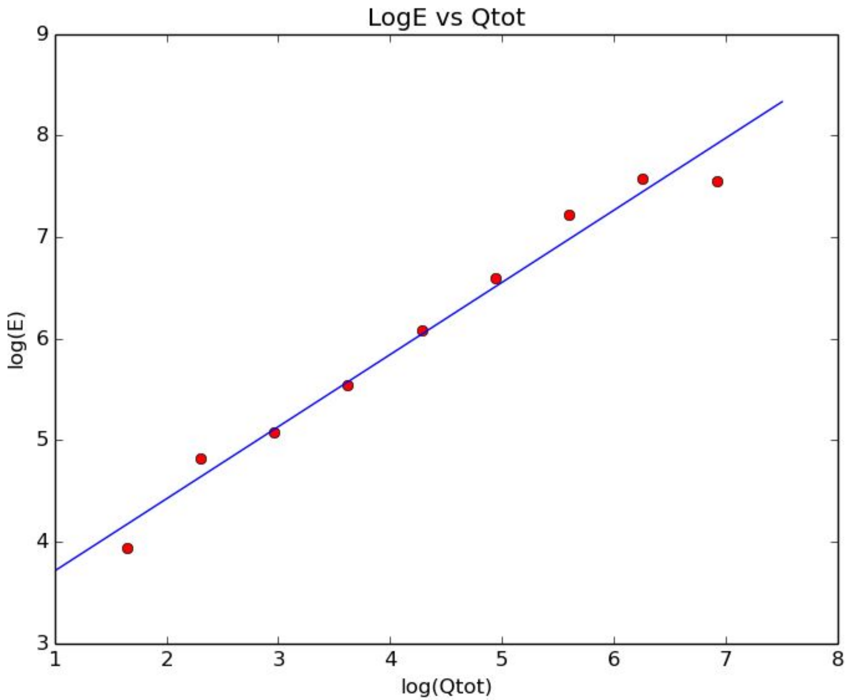


Results

Fit a line with 10 bins

fit =[0.71113522 2.99988121]

χ^2 = 10.780107699

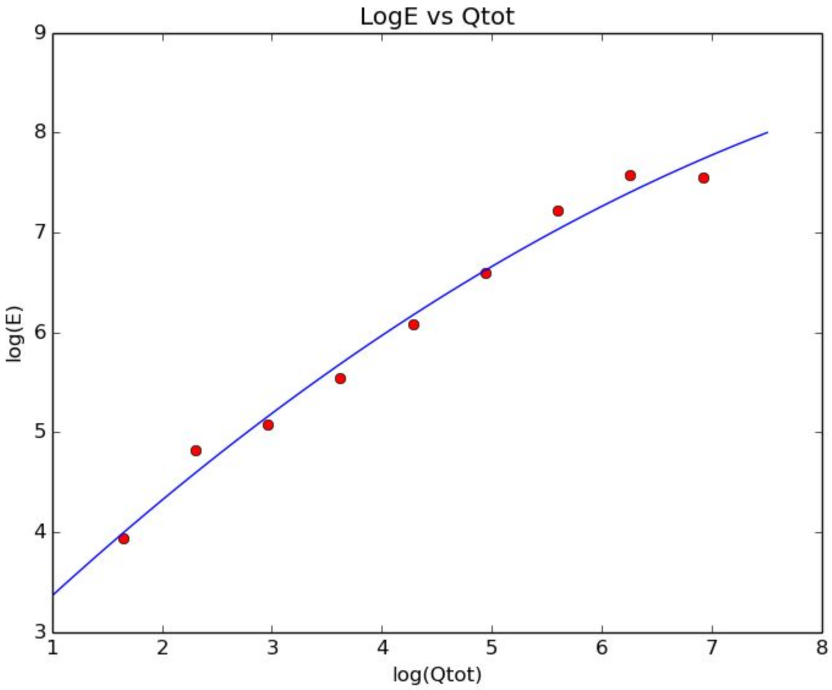


Results

Fit a parabola with 10 bins

fit = [-0.04450927 1.09219117 2.31299158]

χ^2 = 10.4971936556

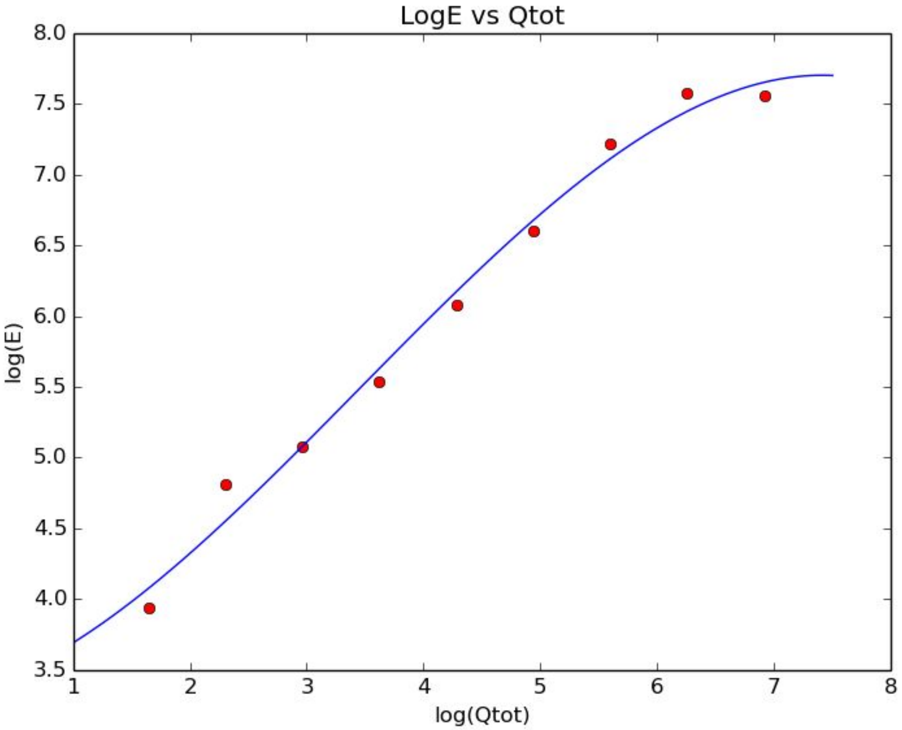


Results

Fit a cubic with 100 bins

fit = [-0.01782318 0.1843744 0.20363661 3.32055109]

χ^2 = 10.5603018162



What we Learned / Re-Learned

- How to use python to read in and find appropriate data in an i3 file
- How to make nice histograms
- How to differentiate CC and NC events
- What this “qtotal” thing is all about
- Ways to fit functions to binned data