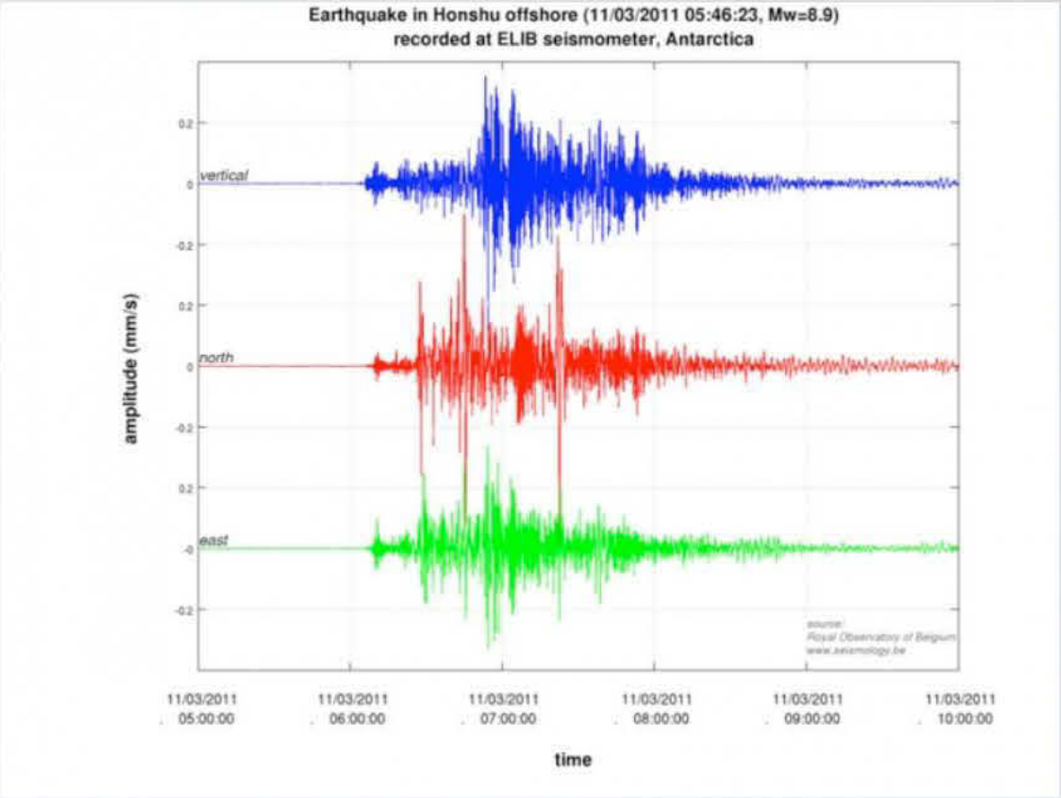
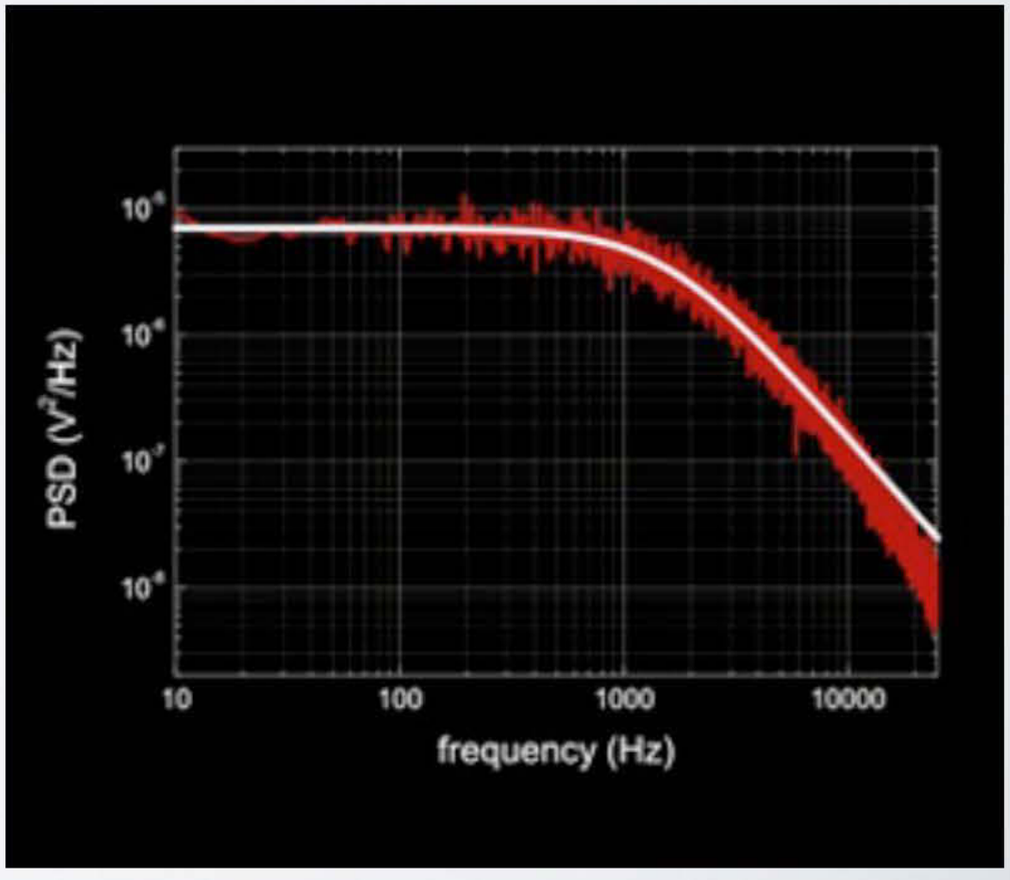
MEASURING GROUND MOTION WITH SEISMOMETERS

BY
AKSHAY IVATURY





Seismometers

-First use dates back to 132 A.D

- Extremely sensitive

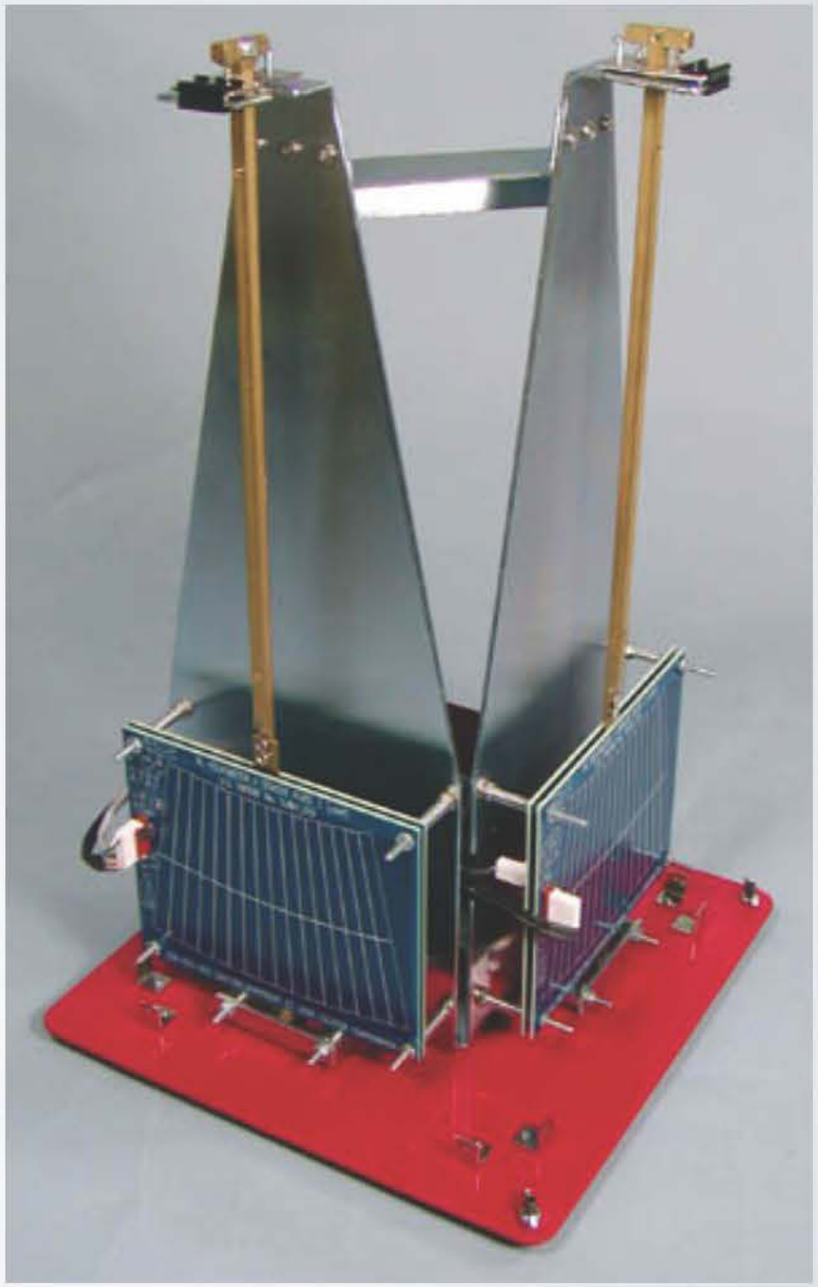
-Instruments used to detect ground motion

VolksMeter II

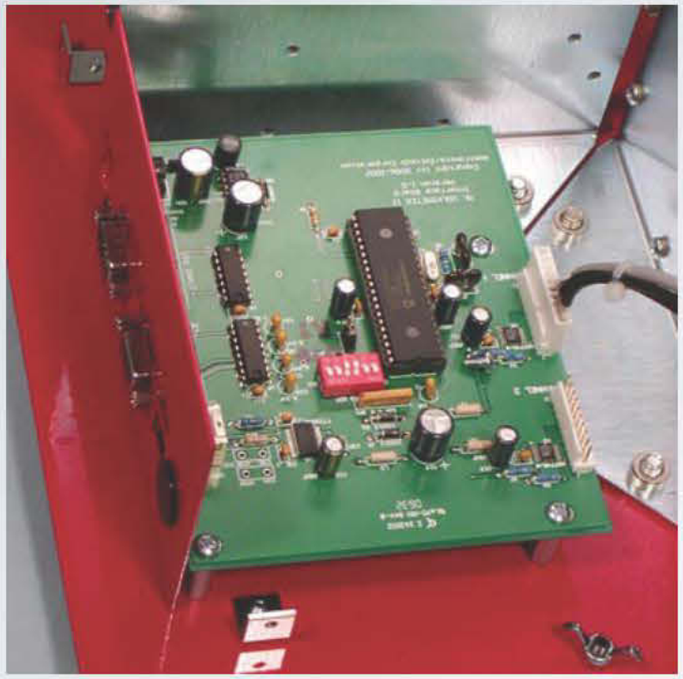
-Developed by RLL Instruments

-Two sensor arrays are mounted at a 90° angle to each other

-Provides useful correlation and two dimensional data



Components



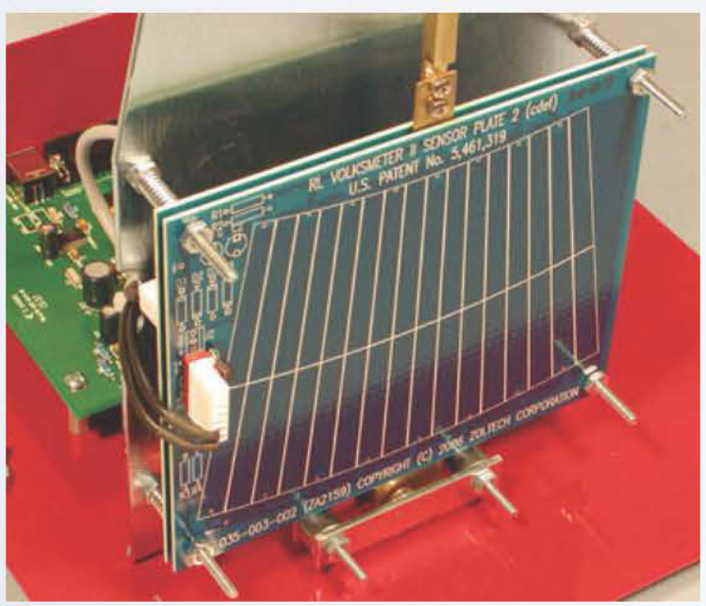
-AD board which reads and transmits data

-GPS to time stamp data

-USB output to connect to computer

Components

-Pendulum is placed between plates



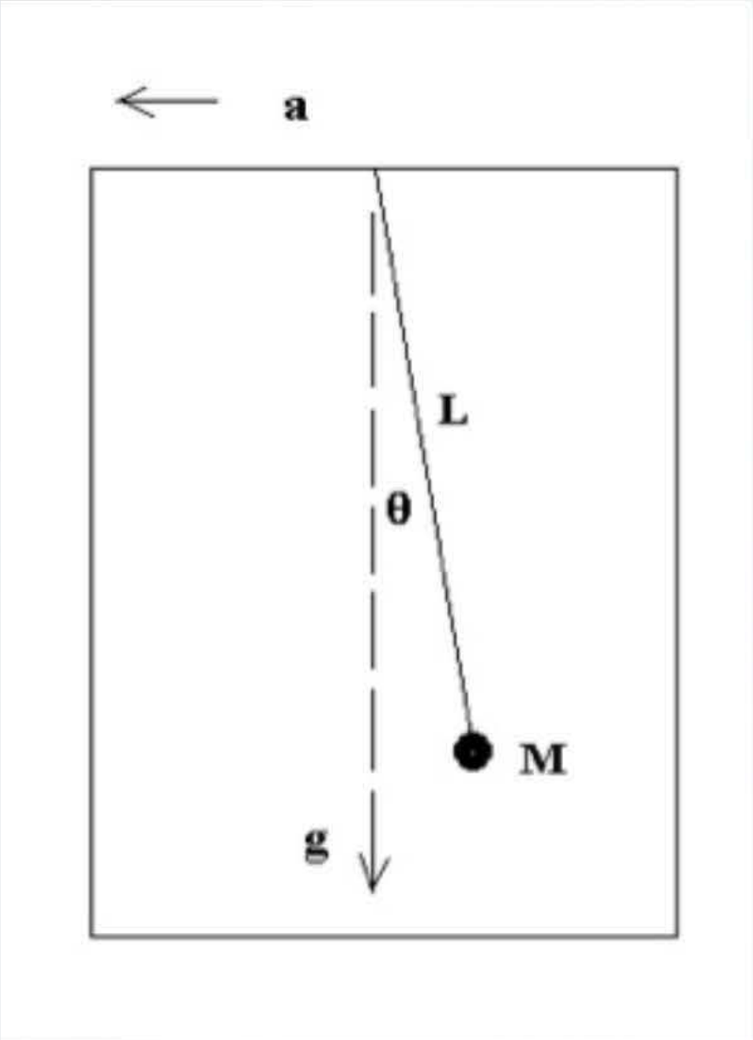
-number readout is proportional to θ

-2.5×10^9 counts/radian

Tools to level

Calibration based on leveling screws

Mathematics of a Pendulum



$g = 9.8\ m/s^2$

$\omega = (2\pi)(\text{frequency})$

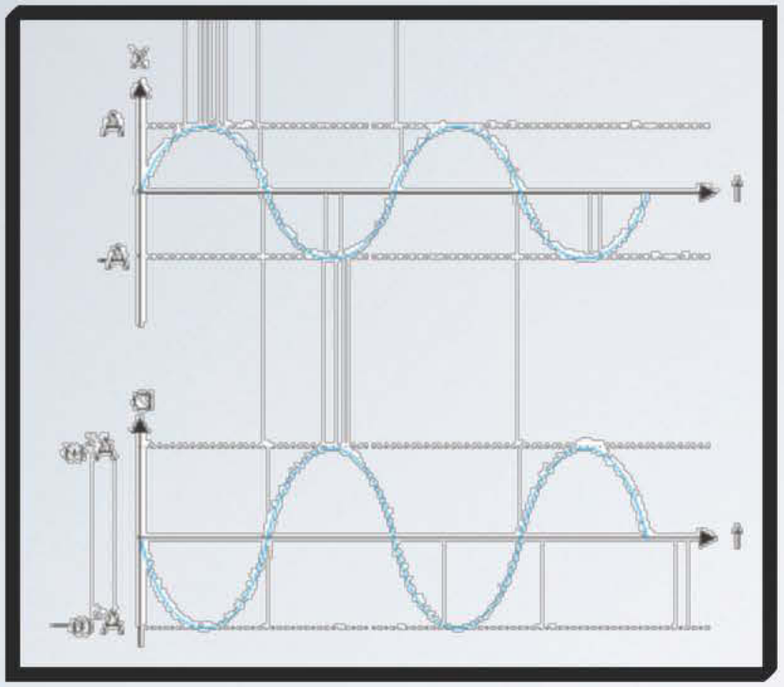
L=length of the pendulum

$\omega^2 = g/L \quad F_0 = .918\ Hz$

$a(t) = g\theta(t)$

Seismometers measure a(t)

The Power Spectrum gives us A_i at all frequencies for ω_i



This seismometer measures a(t)

$$a(t) = \sum(A_i)\cos(\omega_i t) = \sum a_i(t)$$

A_i is the amplitude of acceleration at ω_i in units of m/s^2

We want position with respect to time

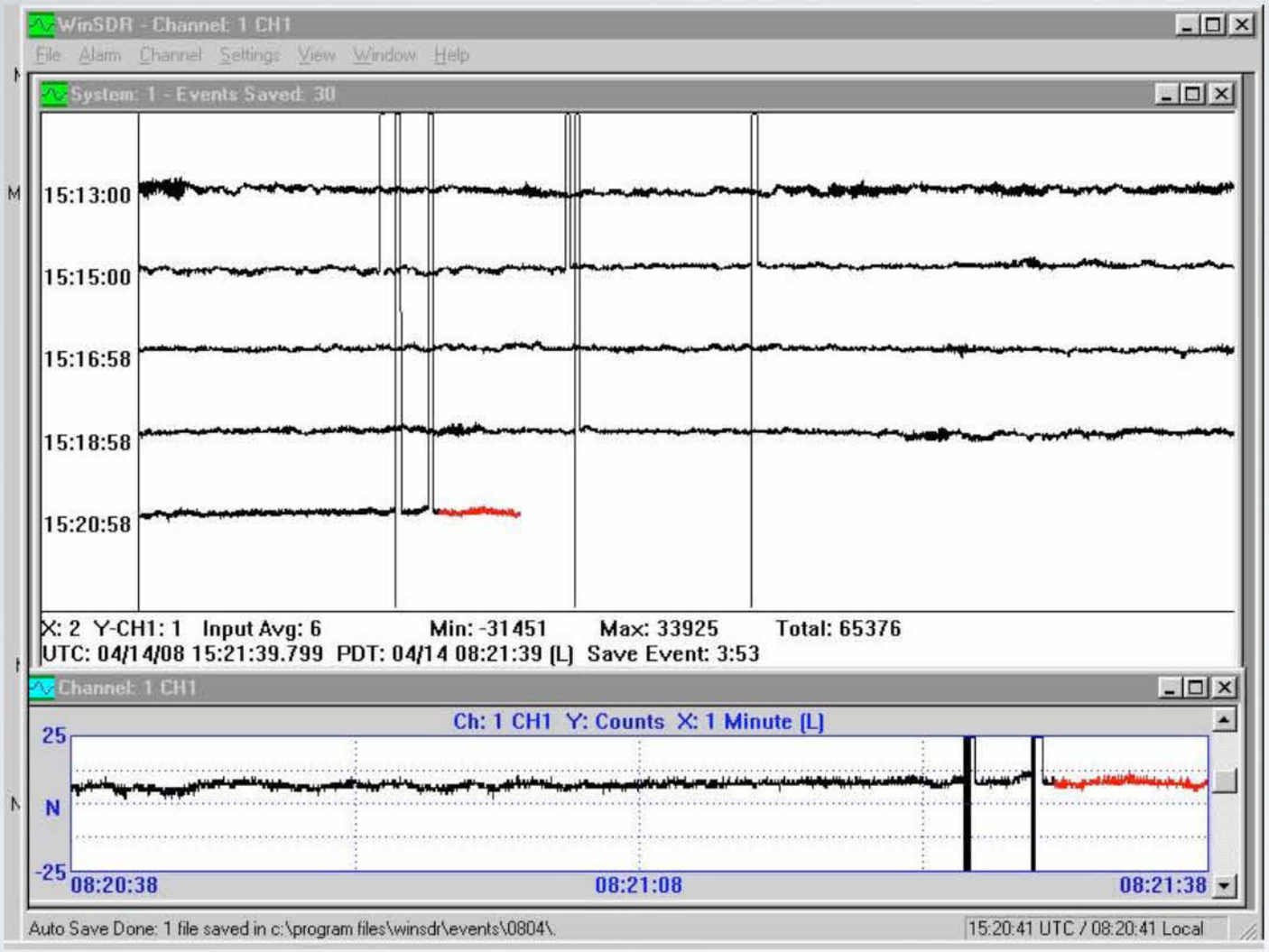
$$x_i(t) = X_i\cos(\omega_i t)$$

we want Xi

$$dx_i/dt = (-X_i)(\omega_i)\sin(\omega_i t) = v_i(t)$$

$$dv_i/dt = (-X_i)(\omega_i^2)\cos(\omega_i t) = a_i(t)$$

$$\therefore X_i = -A_i/(\omega_i^2)$$



-The program we use to collect the data stream is winSDR

-Unable to open this type of file

Extracting Data

```
import numpy
import seisdata

def extract(sdataFile,out):

    s=seisdata.Seisdata(sdataFile)
    ch1,ch2 = numpy.hsplit(s.samples,2)

    f = open(out,'w')

    for i in range(0,len(ch1),1):
        a = str(float(ch1[i]))
        b = str(float(ch2[i]))
        st = a+" "+b+"\n"
        f.write(st)

    f.close()
    return out
```

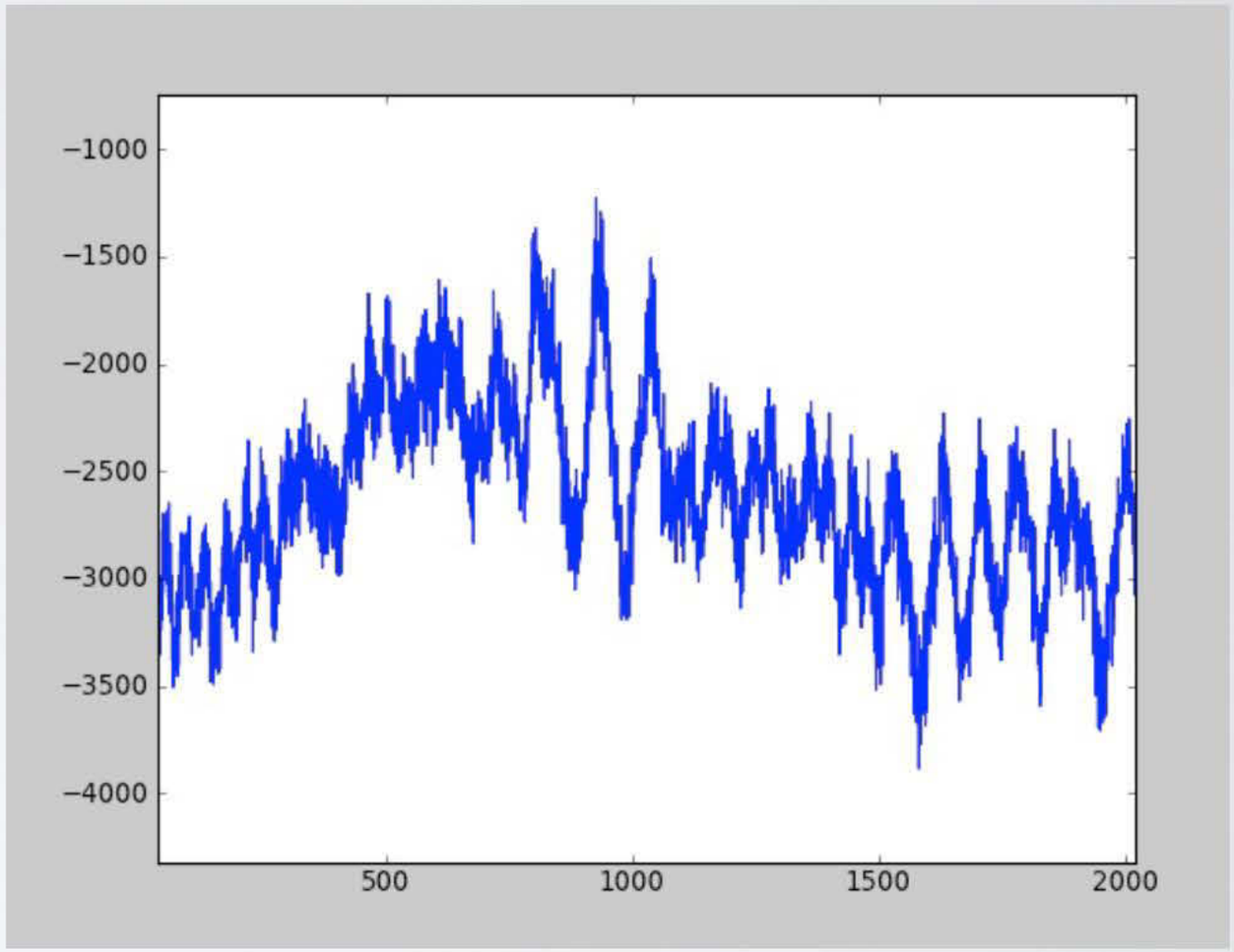
```
-2811.0
-3305.0
-2912.0
-3102.0
-2849.0
-3229.0
-2950.0
-2988.0
-2976.0
-3039.0
-3292.0
-3166.0
-3242.0
-3064.0
-3356.0
-3064.0
-3394.0
-2950.0
-3394.0
-3115.0
-3254.0
-3064.0
-3406.0
-3178.0
-3495.0
-3356.0
-3178.0
-3520.0
-3280.0
-3761.0
```

Sample Data

-This is a stream of data which comes from unpacking information from the seismometers

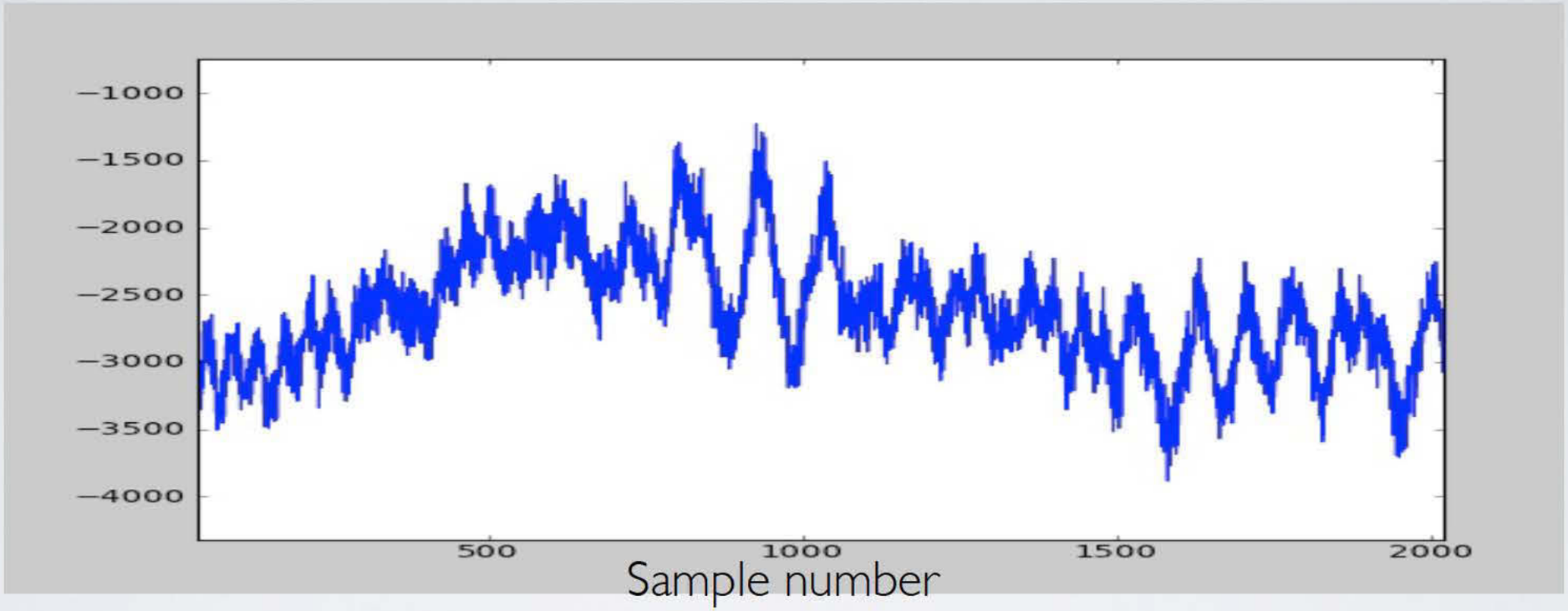
-Python is the programming language used to plot the overnight data stream

```
-2811.0
-3305.0
-2912.0
-3102.0
-2849.0
-3229.0
-2950.0
-2988.0
-2976.0
-3039.0
-3292.0
-3166.0
-3242.0
-3064.0
-3356.0
-3064.0
-3394.0
-2950.0
-3394.0
-3115.0
-3254.0
-3064.0
-3406.0
-3178.0
-3495.0
-3356.0
-3178.0
-3520.0
-3280.0
-3761.0
```

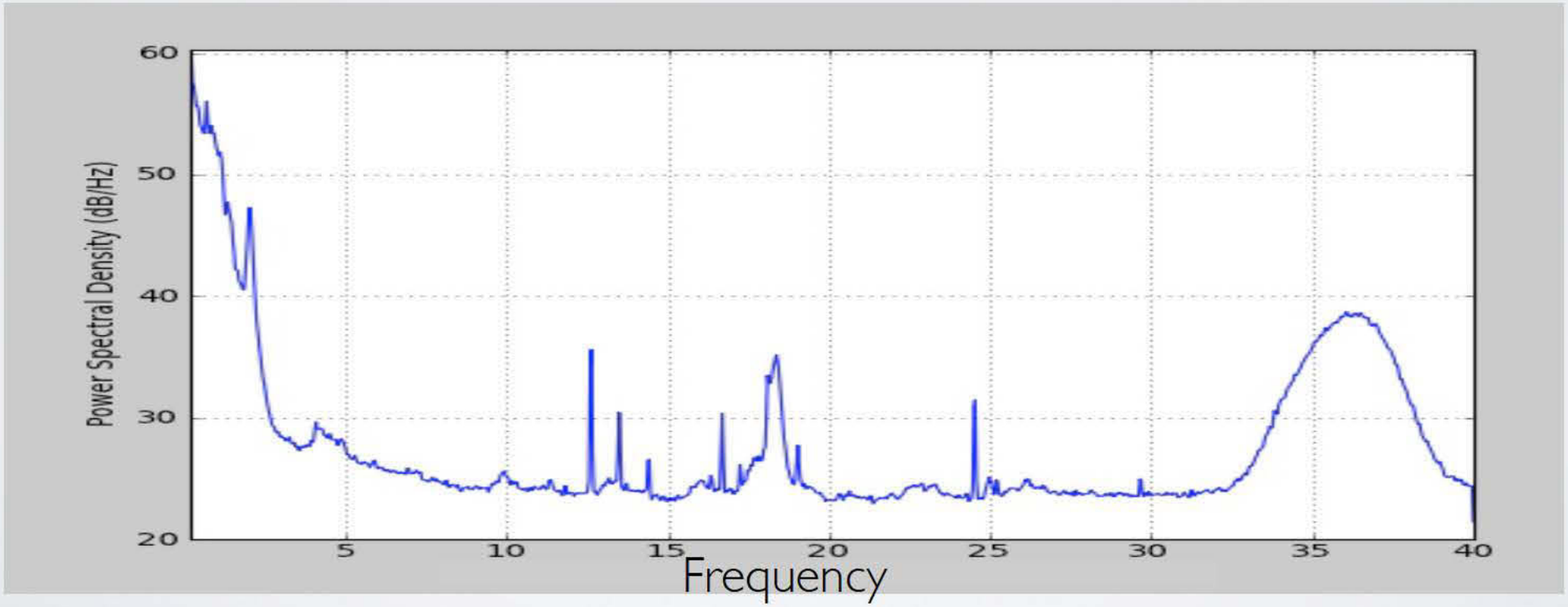


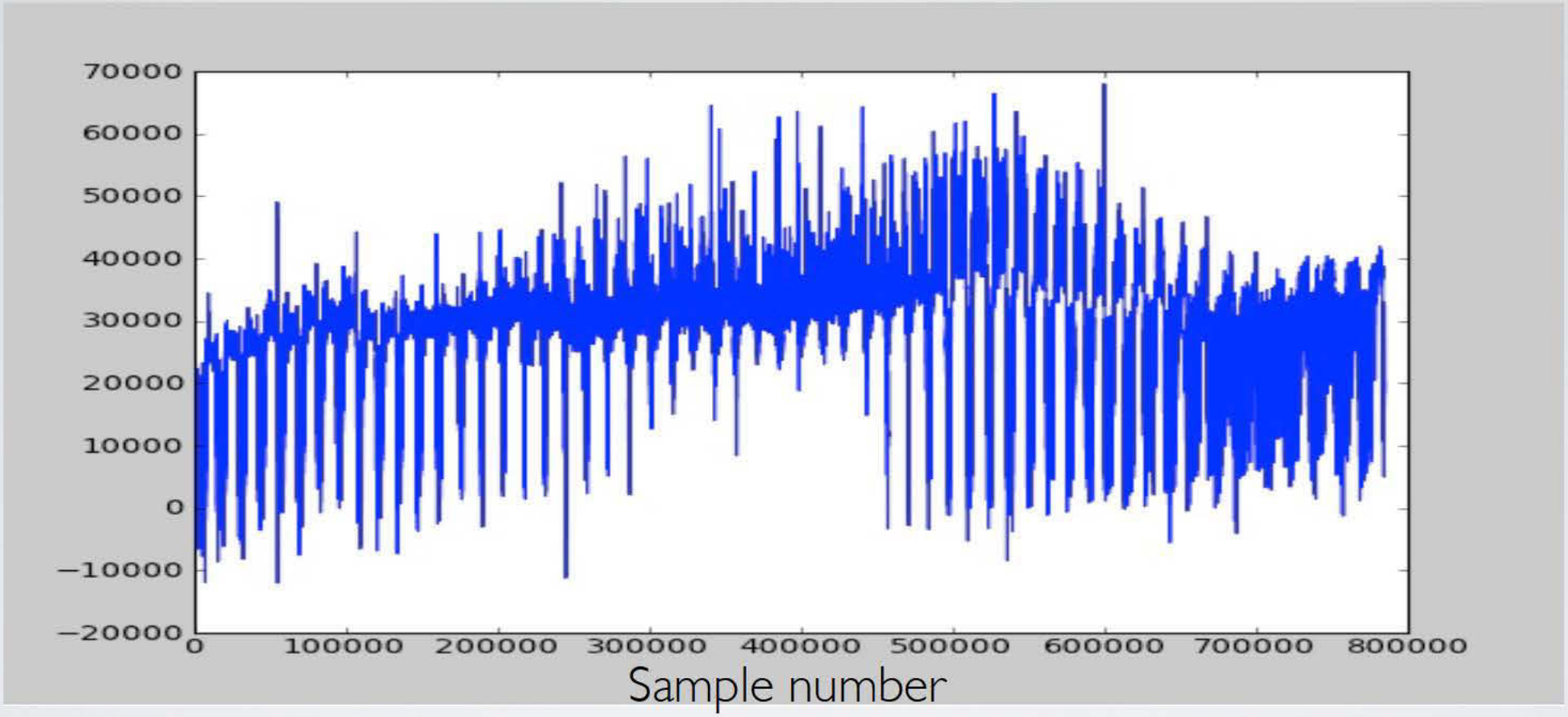
Sample number
(80 Hz)

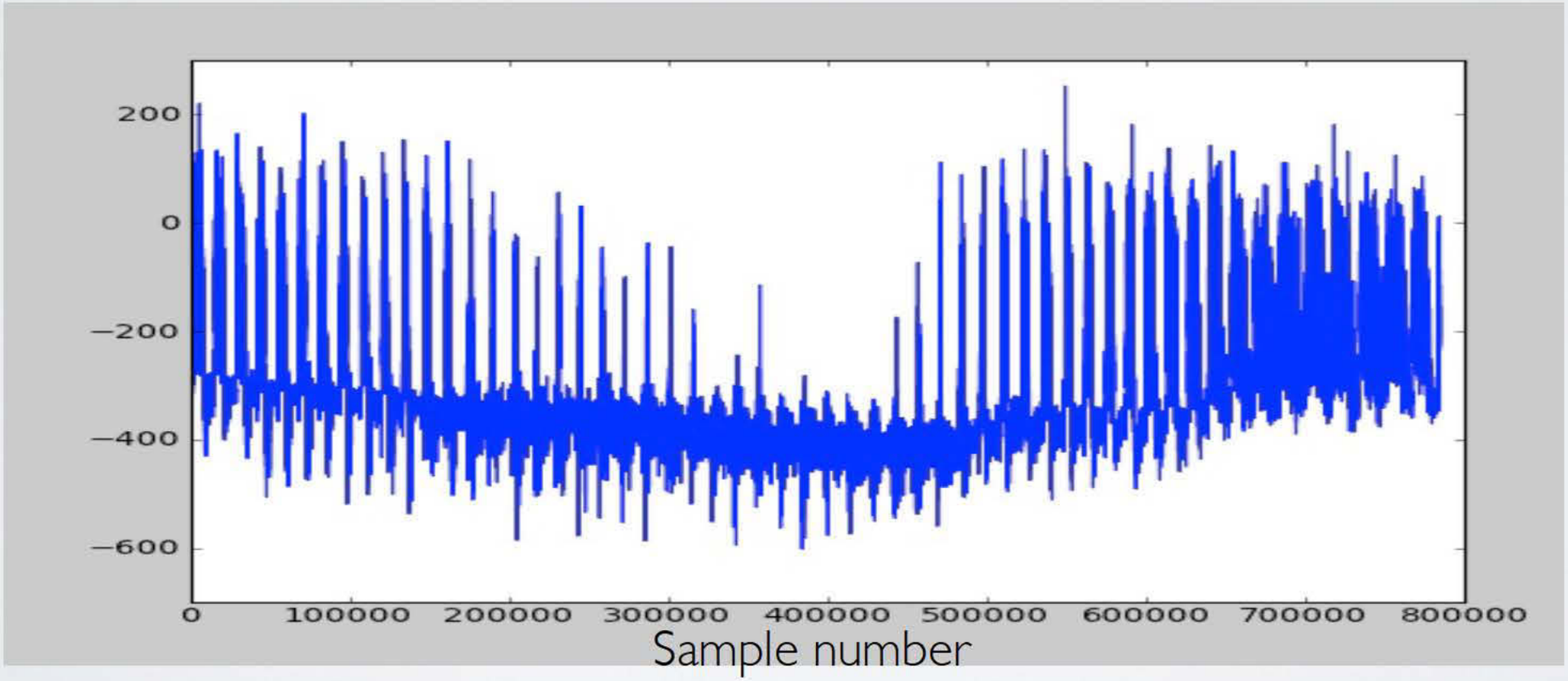
Power Spectral Density



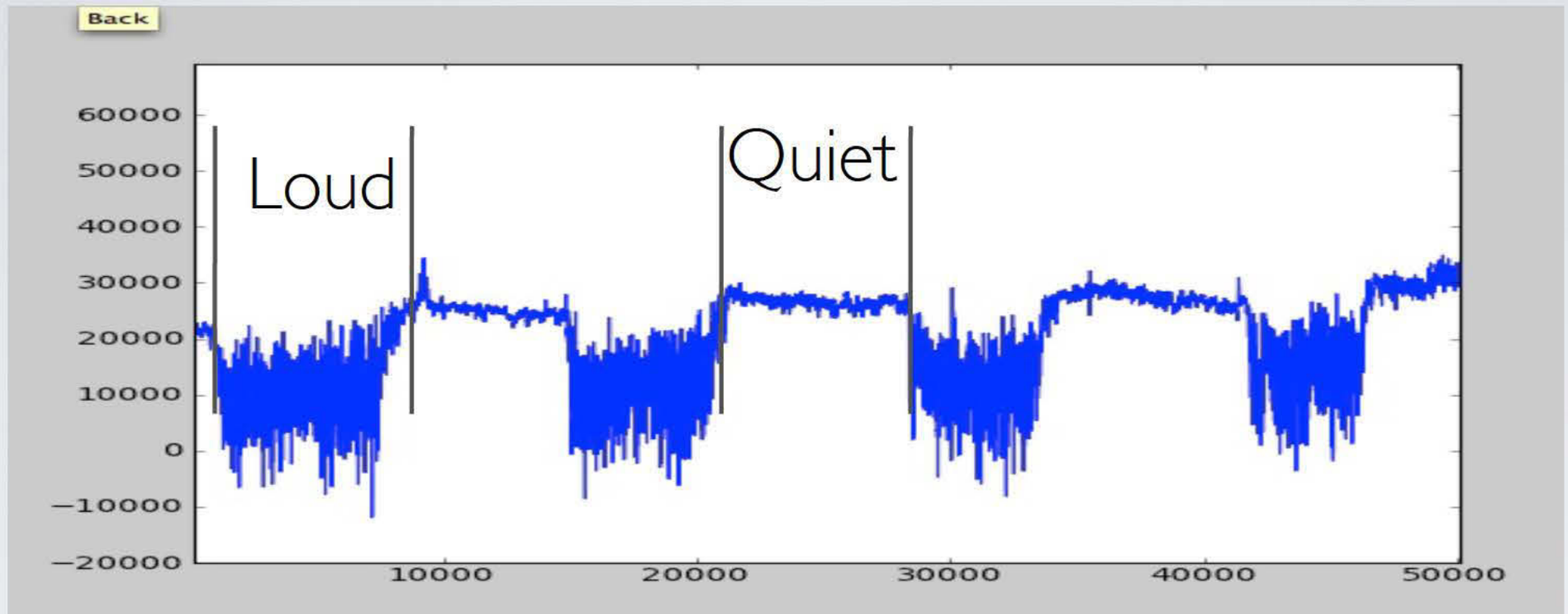

```
matplotlib.pyplot.psd(xaxis,2**6,80)
```

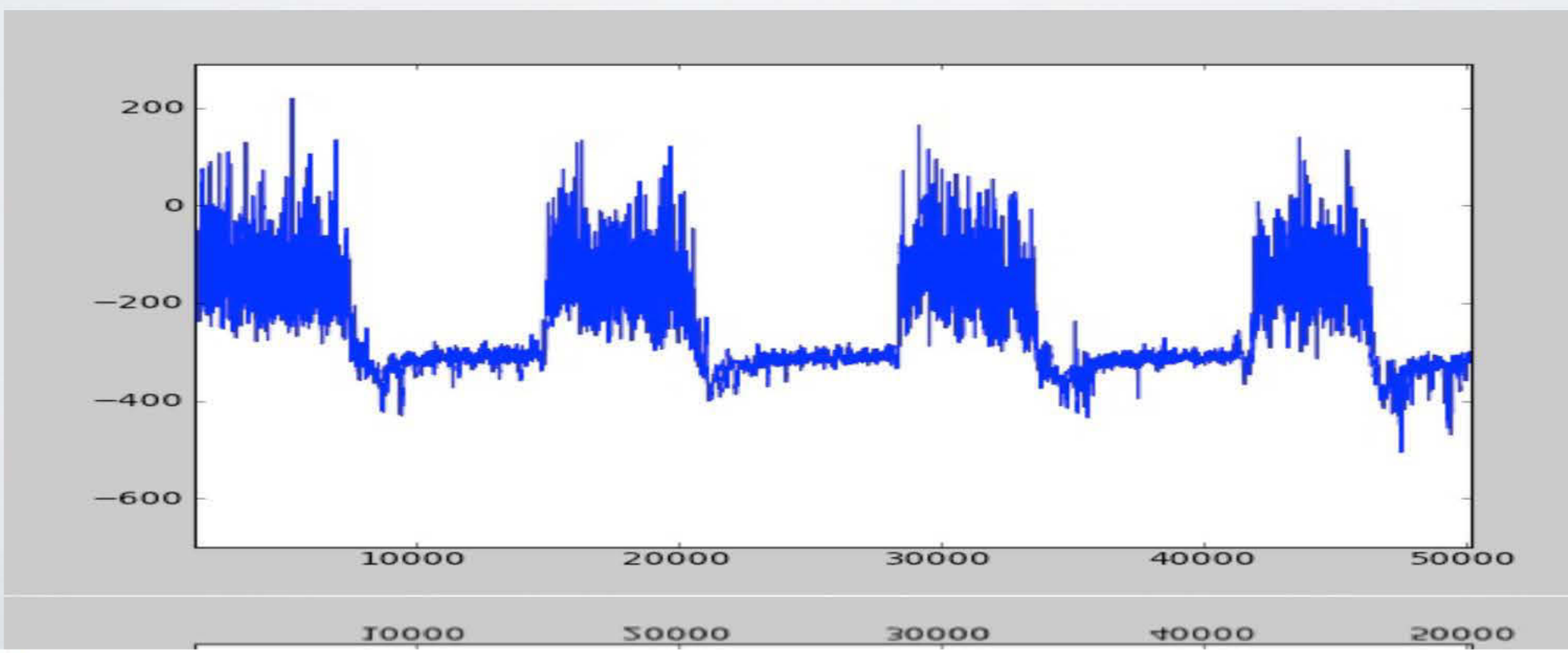



70000
60000
50000
40000
30000
20000
10000
0
−10000
−20000
0
100000
200000
300000
400000
500000
600000
700000
800000
Sample number

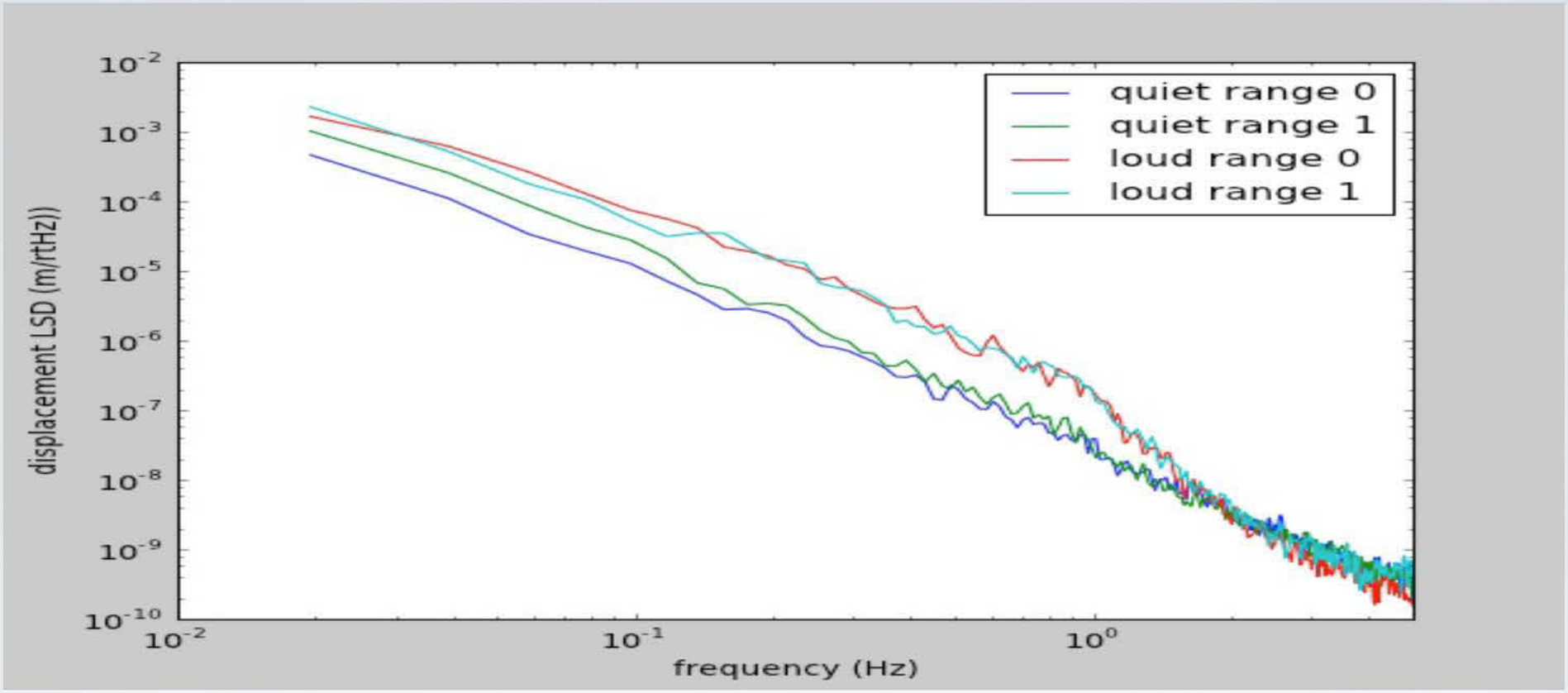

200
0
−200
−400
−600
0
100000
200000
300000
400000
500000
600000
700000
800000
Sample number

Zoomed in view



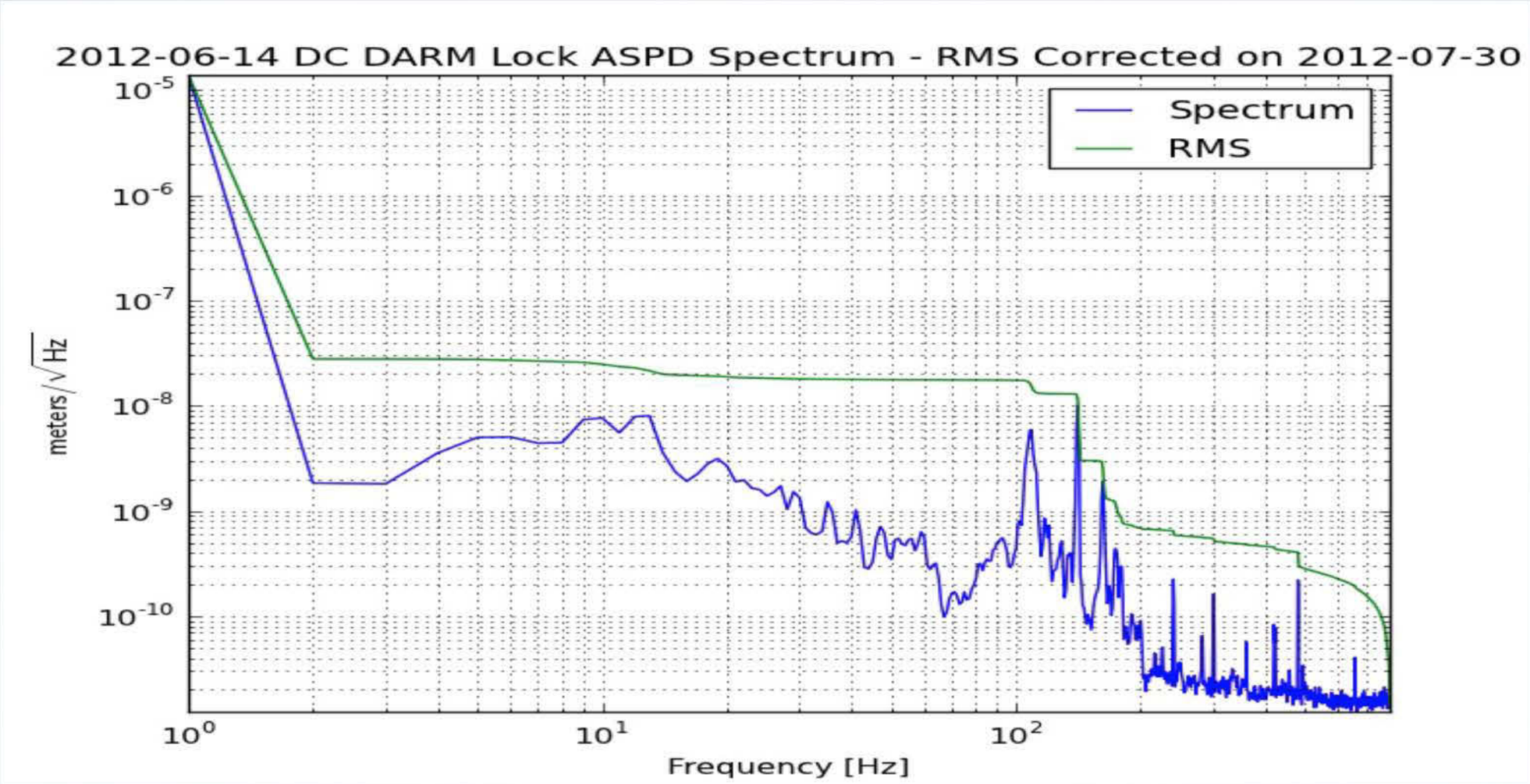


Seismometer Results



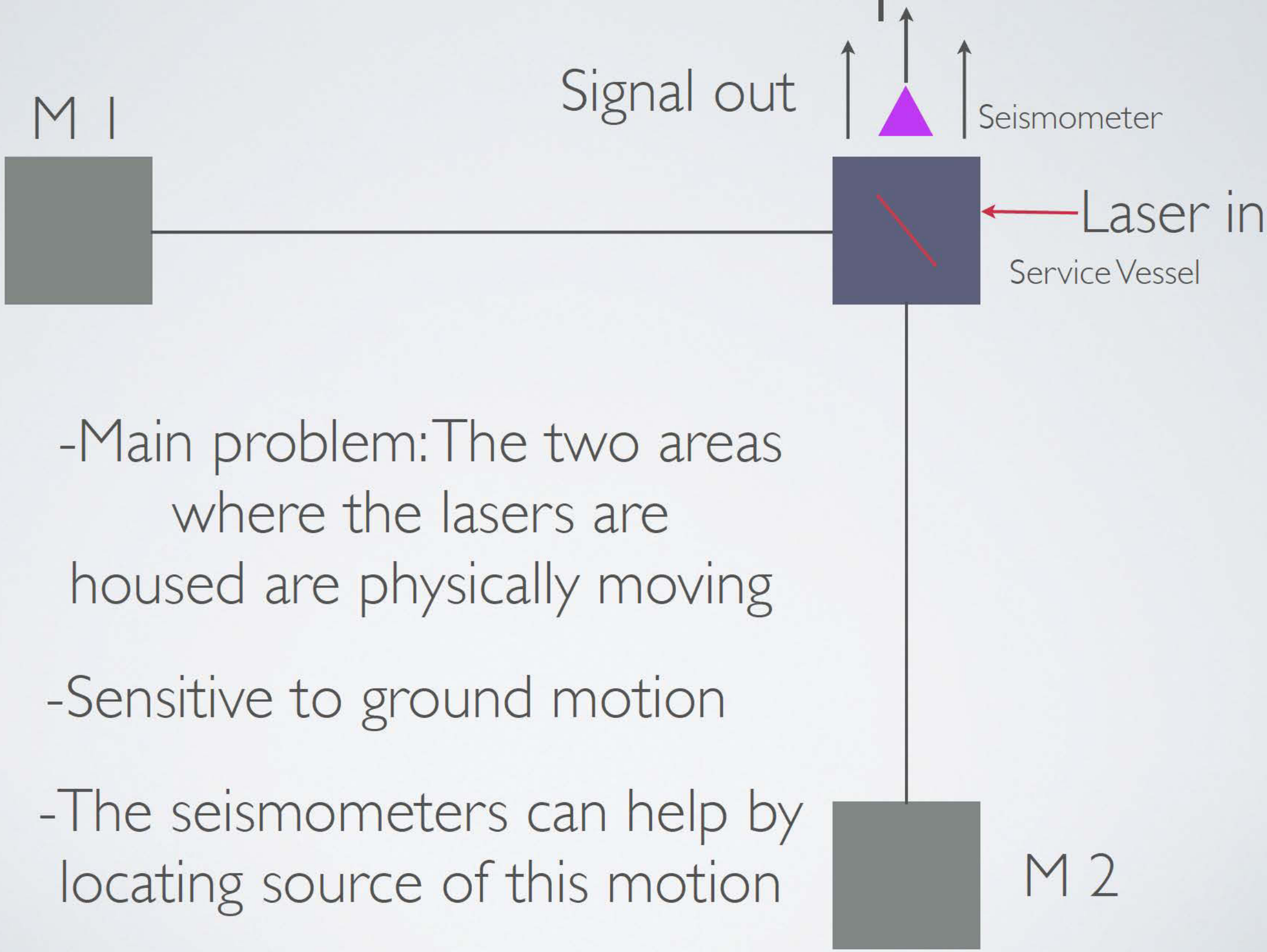
At 1Hz we're are observing a displacement of about 10^{-8} m/$\sqrt{\text{Hz}}$

Interferometer Results



Here we can see that at 1 Hz, we are getting something at about 10^{-5} m/$\sqrt{\text{Hz}}$

Interferometer setup



-Main problem: The two areas where the lasers are housed are physically moving

-Sensitive to ground motion

-The seismometers can help by locating source of this motion

How this helps the Holometer

-We can see that something here at Fermilab is turning off and on every couple minutes

-Can compensate for that noise in the Holometer

Questions, Comments?

```
import numpy
import pylab
import matplotlib
from numpy import *
from pylab import *

f = raw_input("Enter Seisdata Here:  ")

f = open(f)
lines = f.readlines()
xaxis = []
yaxis = []
for i in lines:
    parts=i.split(" ")
    xaxis.append(float(parts[0]))
    yaxis.append(float(parts[1]))

print " xaxis plot "
pylab.plot(xaxis)
pylab.show()

print " yaxis plot "
pylab.plot(xaxis)
pylab.show()
```

```
import numpy
import pylab
import matplotlib
from numpy import *
from pylab import *

f = raw_input("Enter Seisdata Here: ")
nfft = input("Enter nfft Here: ")
sampleRate = input("Enter sampleRate Here: ")
f = open(f)

lines = f.readlines()
xaxis = []
yaxis = []
for i in lines:
    parts=i.split(" ")
    xaxis.append(float(parts[0]))
    yaxis.append(float(parts[1]))

print " xaxis psd "
matplotlib.pyplot.psd(xaxis,nfft,sampleRate)
pylab.show()

print " yaxis psd "
matplotlib.pyplot.psd(yaxis,nfft,sampleRate)
pylab.show()
```