EAE 298 Aeroacoustics
Fall Quarter 2016
Homework #1

John Karasinski

October 17, 2016

Problem 1. [50 pts]

The values in the wav file are in volts. B&K measurement microphones invert the pressure a negative voltage from the microphone corresponds to a positive pressure. When you apply the calibration constant, account for this sign reversal. For this problem, the pre-calculated constant calibration factor is 116 pascals/volt. Convert the time series in voltage to pascals. (Assume that all of the power in the boom waveform is within the range of flat response of the microphone).

Part (a)

[10 pts] Plot the waveform in pascals as a function of time. What is the peak pressure in the time domain? Notice the shape of the first arrival it has the classic N wave shape of a sonic boom. Notice the duration in time from the positive-pressure peak to the negative-pressure peak.

Solution

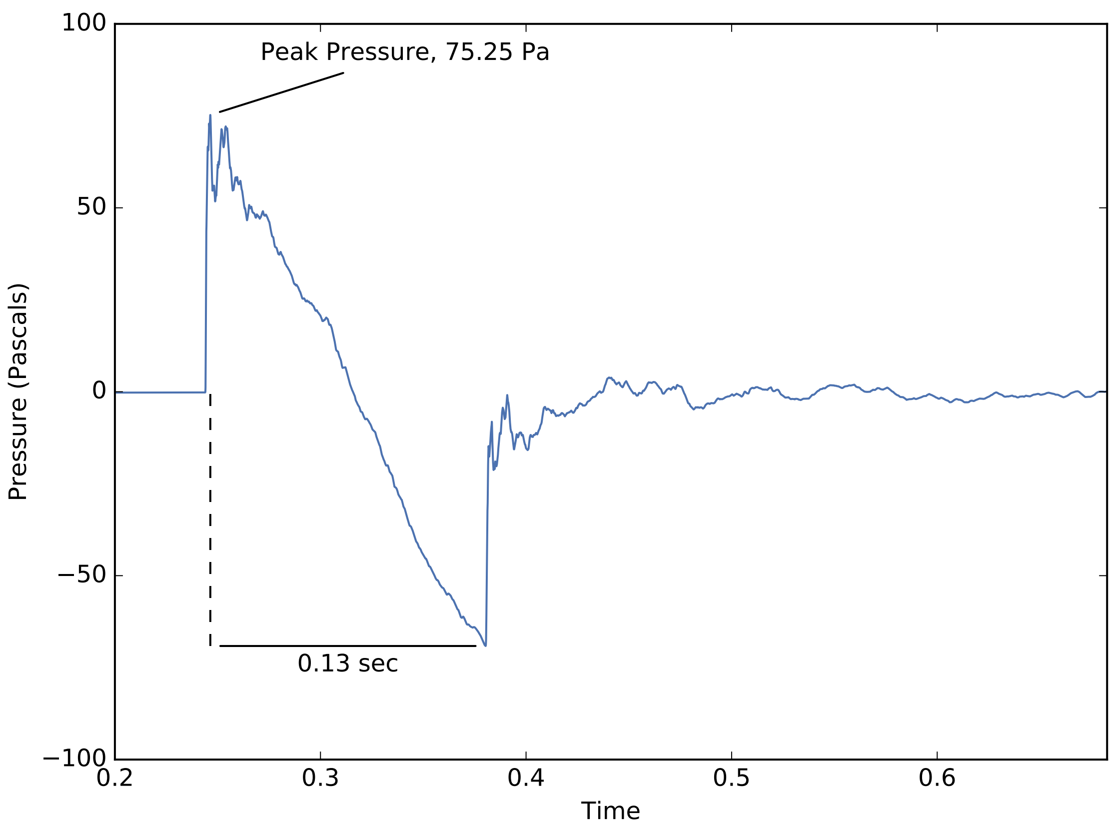
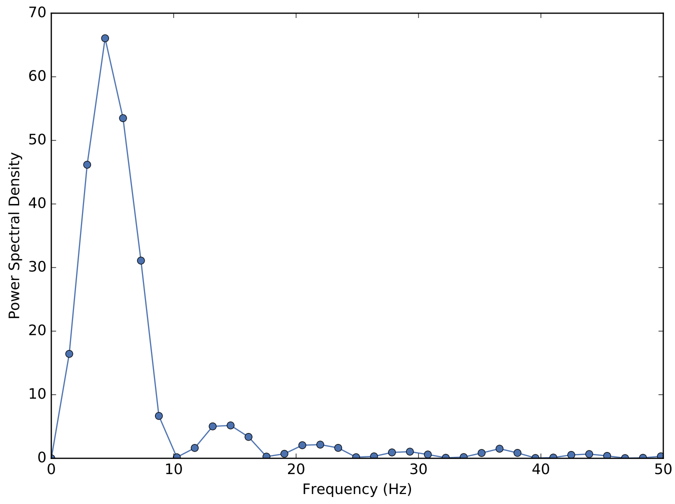


Figure 1: The waveform in pascals as a function of time. Peak pressure is noted at 75.25 Pa, and the duration in time from the positive-pressure peak to the negative-pressure peak is noted at 0.13 sec.

Part (b)

[30 pts] Calculate and plot the single-sided power spectral density function (G_{xx}).

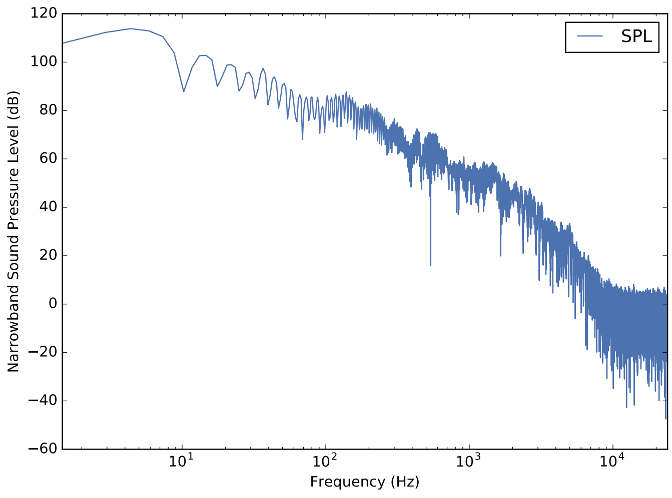
Solution



Part (c)

[10 pts] Convert and plot the standard narrowband sound pressure level with the reference pressure of 20 micro-Pascal.

Solution



Problem 2. [50 Pts]

Write a computer program to convert the narrow band spectra to one-third octave and octave band spectra.

[20 pts] Convert the narrowband spectrum to one-third octave band spectrum and make a plot. [20 pts] Convert the one-third octave band spectrum to octave band spectrum and make a plot. [10 pts] Convert the octave band spectrum to the overall sound pressure level.

Solution

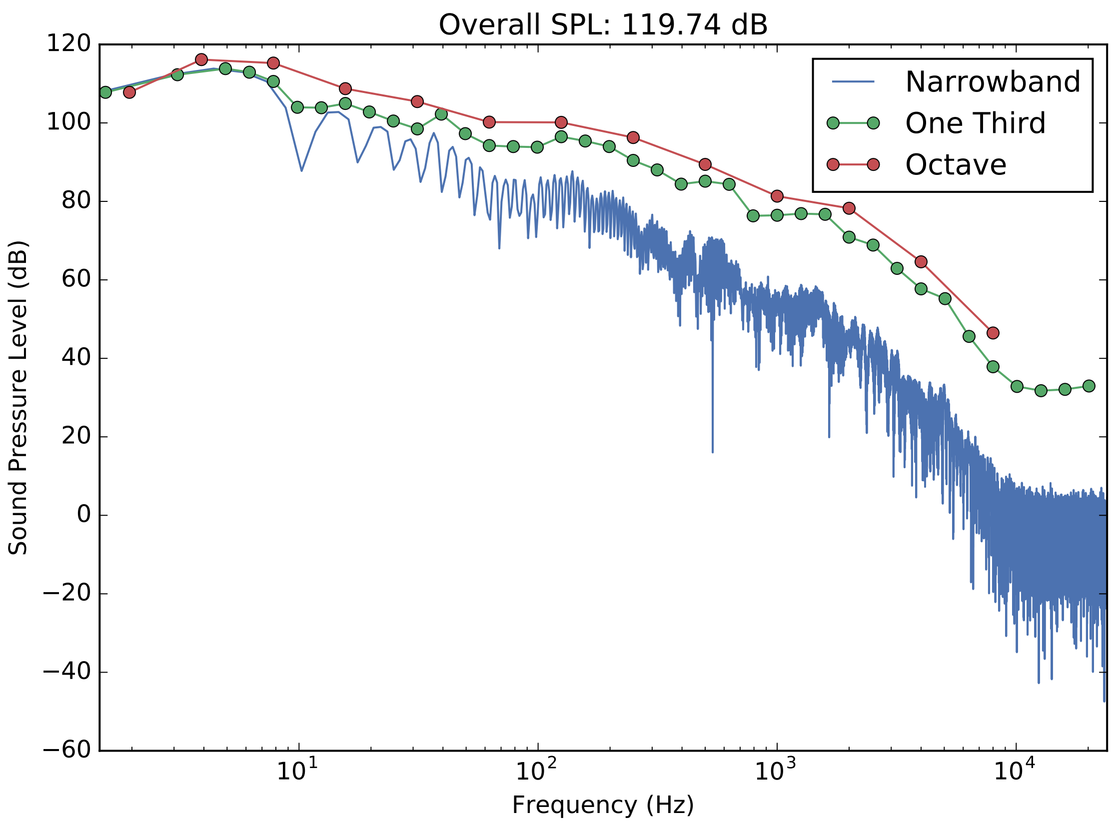


Figure 2: The standard narrowband, 1/3 octave, and octave sound pressure levels with the reference pressure of 20 micro-Pascal. This recording has an overall sound pressure level of 119.74 dB.

1/3 Octave Results

Frequency (Hz)	SPL (dB)
1.55	107.79
3.10	112.28
4.92	113.83
6.20	112.92
7.81	110.56
9.84	103.97
12.40	103.86
15.62	104.95
19.68	102.79
24.80	100.48
31.25	98.47
39.37	102.24
49.60	97.26
62.50	94.23
78.74	93.97
99.21	93.81
125.00	96.48
157.49	95.39
198.42	93.97
250.00	90.44
314.98	88.02
396.85	84.40
500.00	85.15
629.96	84.34
793.70	76.32
1000.00	76.47
1259.92	76.88
1587.40	76.70
2000.00	70.88
2519.84	68.87
3174.80	62.93
4000.00	57.71
5039.68	55.22
6349.60	45.59
8000.00	37.86
10079.36	32.84
12699.20	31.75
16000.00	32.09
20158.73	32.95

Octave Results

Frequency (Hz)	SPL (dB)
1.95	107.79
3.90	116.13
7.81	115.24
15.62	108.73
31.25	105.43
62.50	100.20
125.00	100.13
250.00	96.27
500.00	89.42
1000.00	81.33
2000.00	78.25
4000.00	64.60
8000.00	46.46

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import scipy.signal
import seaborn as sns
import soundfile as sf

sns.reset_orig()
colors = sns.color_palette('deep', 10)

################################################################################
# Problem 1
################################################################################

# Load the audio file
audio, fs = sf.read('Boom_F1B2_6.wav')

df = pd.DataFrame(audio)
df.columns = ['Voltage']
df['Time'] = np.array(df.index) / fs

# Convert to pascals
df['Pascals'] = df.Voltage * -116.

# Find the peak pressure
maxPeak = df.ix[df.Pascals.idxmax()]
minPeak = df.ix[df.Pascals.idxmin()]


def plot1a():
    f, ax = plt.subplots()
    df.plot(x='Time', y='Pascals', legend=False, color=colors[0], ax=ax)

    plt.xlim(xmin=0.2)
    plt.ylim(-100, 100)
    plt.ylabel('Pressure (Pascals)')

    arrowprops = dict(arrowstyle="-",
                      shrinkA=5, shrinkB=5,
                      connectionstyle="arc3")

    ax.annotate('Peak Pressure, {:2.2f} Pa'.format(maxPeak.Pascals),
                xy=(maxPeak.Time, maxPeak.Pascals),
                xytext=(25, 25), textcoords='offset points',
                arrowprops=arrowprops,
                horizontalalignment='left', verticalalignment='bottom')

    ax.annotate('',
                xy=(minPeak.Time, minPeak.Pascals),
                xytext=(maxPeak.Time, minPeak.Pascals),
                arrowprops=arrowprops)

    plt.text((maxPeak.Time + minPeak.Time) / 2, 1.1 * minPeak.Pascals,
```

```
            '{:2.2f} sec'.format(minPeak.Time - maxPeak.Time),
            horizontalalignment='center')

    plt.vlines(x=maxPeak.Time, ymax=0, ymin=minPeak.Pascals, linestyles='--')

    plt.tight_layout()
    plt.savefig('tex/figs/Pascals_vs_Time.pdf')
    plt.clf()
#plot1a()

################################################################################
# Calculate the single sided power spectral density function
f, Gxx = scipy.signal.periodogram(df.Pascals, fs=fs, return_onesided=True)


def plot1b():
    plt.plot(f, Gxx, marker='o', color=colors[0])

    plt.xlim(0, 50)
    plt.xlabel('Frequency (Hz)')
    plt.ylabel('Power Spectral Density')

    plt.tight_layout()
    plt.savefig('tex/figs/G_xx.pdf')
    plt.clf()
#plot1b()

###############################################################################
# Calculate and plot standard narrowband sound pressure level
Pref = 20E-6

T = len(df) / fs
SPL = 10 * np.log10((Gxx / T) / Pref ** 2)
df = pd.DataFrame([f, SPL]).T
df.drop(0, inplace=True)
df.columns = ['Frequency', 'SPL']


def plot1c():
    df.plot(x='Frequency', y='SPL', logx=True, color=colors[0])

    plt.xlabel('Frequency (Hz)')
    plt.ylabel('Narrowband Sound Pressure Level (dB)')

    plt.tight_layout()
    plt.savefig('tex/figs/narrowband.pdf')
    plt.clf()
#plot1c()

###############################################################################
# Problem 2
###############################################################################


```

```python
def generate_octaves(octave=1., upper_frequency=10E3):
    fc30 = 1000
    m = np.arange(1, 80)

    center = fc30 * 2 ** (-10 + (m * octave))
    upper = center * 2 ** (octave / 2)
    lower = center / 2 ** (octave / 2)

    freqs = pd.DataFrame([lower, center, upper]).T
    freqs.columns = ['lower', 'center', 'upper']

    return freqs.query('upper < @upper_frequency')


def sum_octaves(df, octave=1.):
    octave = generate_octaves(octave, upper_frequency=df.Frequency.max())
    res = []
    for i, band in octave.iterrows():
        b = df.query('@band.lower < Frequency < @band.upper')
        Lp = 10 * np.log10((10 ** (b.SPL / 10)).sum())
        if Lp > 0:
            res.append([band.center, Lp])

    res = pd.DataFrame(res)
    res.columns = ['Frequency', 'SPL']
    return res


third = sum_octaves(df, octave=1 / 3)
full = sum_octaves(third, octave=1.)
overall = 10 * np.log10((10 ** (full.SPL / 10)).sum())


def plot2():
    f, ax = plt.subplots()
    df.plot(x='Frequency', y='SPL', logx=True,
            color=colors[0], legend=False, ax=ax)
    third.plot(x='Frequency', y='SPL', logx=True,
               marker='o', color=colors[1], legend=False, ax=ax)
    full.plot(x='Frequency', y='SPL', logx=True,
              marker='o', color=colors[2], legend=False, ax=ax)

    plt.xlabel('Frequency (Hz)')
    plt.ylabel('Sound Pressure Level (dB)')
    plt.legend(['Narrowband', 'One Third', 'Octave'])
    plt.title('Overall SPL: {:2.2f} dB'.format(overall))

    plt.tight_layout()
    plt.savefig('tex/figs/octaves.pdf')
    plt.clf()
#plot2()
```