CS 237 Lab Ten: Correlation and Autocorrelation of Signals

Due date: PDF file due Thursday November 29th @ 11:59PM (10% off if up to 24 hours late) in GradeScope

General Instructions

Please complete this notebook by filling in solutions where indicated. Be sure to “Run All” from the Cell menu before submitting.

You may use ordinary ASCII text to write your solutions, or (preferably) Latex. A nice introduction to Latex in Jupyter notebooks may be found here: http://data-blog.udacity.com/posts/2016/10/latex-primer/ (http://data-blog.udacity.com/posts/2016/10/latex-primer/)

As with previous homeworks, just upload a PDF file of this notebook. Instructions for converting to PDF may be found on the class web page right under the link for homework 1.
In [2]: # General useful imports
import numpy as np
from numpy import arange, linspace, mean, var, std, sin, cos
from numpy.random import random, randint, uniform, choice, binomial, geometric, poisson
import math
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import matplotlib.mlab as mlab
from matplotlib.ticker import LinearLocator, FormatStrFormatter
from collections import Counter

# Basic Numpy statistical functions
X = [1, 2, 3]

# mean of a list
mean(X)  # might need to use np.mean, np.var, and np.std

# population variance
var(X)

# sample variance  ddof = delta degrees of freedom, df = len(X) - ddof
var(X, ddof=1)

# population standard deviation
std(X)

# sample standard deviation
std(X, ddof=1)

# Scipy statistical functions

from scipy.stats import norm, t, binom, geom, expon, poisson, uniform, bernoulli, pearsonr

# Calculate the correlation coefficient \rho(X,Y)

# Utility functions

# Round to 4 decimal places
def round4(x):
    return round(float(x) + 0.00000000001, 4)

def round4List(lst):
    return [round4(x) for x in lst]
Correlation of Signals

This lab is about the use of correlation to determine the frequency of waveforms with a repetitive structure, such as musical signals. We will first understand what signals are, and then how correlation can be used to quantify how similar two signals are.

A signal \( X \) is just a function from non-negative integers into the reals:

\[
X : [0,1,\ldots,(N-1)] \rightarrow [-1..1]
\]

where \( t = 0, 1, 2, \ldots, N - 1 \) represents time (in some units, such as milliseconds) and \([-1..1]\) represents the amplitude of the signal. For instance here is an example adapted from Lab 9 which is just a sine wave, where \( N = 315 \):

\[
X(k) = \sin(x/10), \text{ for } k = 0, 1, \ldots, 314.
\]

\[
X = [0.0885, 0.0932, 0.0936, 0.0915, 0.089, \ldots, -0.0009, -0.001, -0.001]
\]

A periodic signal \( X \) has a pattern that repeats every \( P \) time units, where \( P \) is called the period. In this lab we will be dealing with signals that are periodic, or approximately periodic.

For the rest of this lab, we will be dealing with sequences that are signals, but all you need to know is that they behave like random variables which, when you "poke" them, give the values \( X[0], X[1], X[2], \ldots \). Therefore, it is not surprising that we can take the correlation of two signals.

The correlation of two signals of length \( N \) is exactly the same as if we consider them to be equiprobable, finite random variables:

\[
\text{Cov}(X, Y) = E(X \cdot Y) - \mu_X \cdot \mu_Y = \frac{\sum_{i=0}^{N-1} X[i] \cdot Y[i]}{N} - \mu_X \cdot \mu_Y
\]

\[
\rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sigma_X \cdot \sigma_Y}
\]

For the rest of this lab, we will be dealing with sequences that are signals, but all you need to know is that they behave like random variables which, when you "poke" them, give the values \( X[0], X[1], X[2], \ldots \). Therefore, it is not surprising that we can take the correlation of two signals.

The correlation of two signals of length \( N \) is exactly the same as if we consider them to be equiprobable, finite random variables:

\[
\text{Cov}(X, Y) = E(X \cdot Y) - \mu_X \cdot \mu_Y = \frac{\sum_{i=0}^{N-1} X[i] \cdot Y[i]}{N} - \mu_X \cdot \mu_Y
\]

\[
\rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sigma_X \cdot \sigma_Y}
\]
Problem One

(A) Complete the following code template to calculate $\rho(X, Y)$, where we assume that $X$ and $Y$ are two signals of the same length. You may use $\text{mean}(\ldots)$ and $\text{std}(\ldots)$ from the numpy library, but otherwise write it yourself.

In [4]:

1.0
-1.0
1.0
0.8

The following utility function simply displays three different signals lined up with respect to their x-axes.
In [5]:

```python
# Just plotting three signals lined up along the x axis

def plot3Signals(X,Y,Z,title1="Signal 1", title2="Signal 2", title3="Signal 3"):  
    N = len(X)  
    fig = plt.figure(figsize=(12,10))  
    fig.subplots_adjust(hspace=.5)  
    ax1 = fig.add_subplot(311)  
        plt.title(title1)  
        plt.xlabel('Time (ms)')  
        plt.ylabel('Amplitude')  
        plt.plot([0,N-1],[0,0],color='black')  
        plt.xlim([0,N-1])  
        plt.ylim([-1.2,1.2])  
        plt.grid()  
        plt.plot(X)

    fig.add_subplot(312,sharex=ax1)

    plt.title(title2)  
    plt.xlabel('Time (ms)')  
    plt.ylabel('Amplitude')  
    plt.plot([0,N-1],[0,0],color='black')  
    plt.xlim([0,N-1])  
    plt.ylim([-1.2,1.2])  
    plt.grid()  
    plt.plot(Y)

    fig.add_subplot(313,sharex=ax1)

    plt.title(title3)  
    plt.xlabel('Time (ms)')  
    plt.ylabel('Amplitude')  
    plt.plot([0,N-1],[0,0],color='black')  
    plt.xlim([0,N-1])  
    plt.ylim([-1.2,1.2])  
    plt.grid()  
    plt.plot(Z)

# example

T = list(range(314))
X = [math.sin(x/10) for x in T]
Y = [math.cos(x/10) for x in T]
Z = [math.sin(x/5) for x in T]

plot3Signals(X,Y,Z)
```
(B) Using the function in the previous code cell, print out the graphs for

Signal 1: \( X(k) = \sin(x/10) \) for \( 0 \leq k < 315 \).

Signal 2: \( Y = X \)

Signal 3: \( Z = X \cdot Y \)

and then print out \( \rho(X, Y) \). This example was shown in lecture on the board.
In [6]: # (B) solution

\[
\rho(X,Y) = 1.0
\]

(C) Now print out the graphs for these three collections of signals (which should be familiar to you from lecture):

**First:**

Signal 1: \( X(k) = \sin(x/10) \) for \( 0 \leq k < 315 \).

Signal 2: \( Y = -X \)

Signal 3: \( Z = X \cdot Y \)

and then print out \( \rho(X, Y) \).

**Second:**

Signal 1: \( X(k) = \sin(x/10) \) for \( 0 \leq k < 315 \).

Signal 2: \( X(k) = \cos(x/10) \) for \( 0 \leq k < 315 \).

Signal 3: \( Z = X \cdot Y \)

and then print out \( \rho(X, Y) \). (In this case, do not round to 4 decimal place, since it will be very close to 0.)

**Third:**

Signal 1: \( X(k) = \sin(x/10) \) for \( 0 \leq k < 315 \).

Signal 2: \( X(k) = \sin(x/5) \) for \( 0 \leq k < 315 \).

Signal 3: \( Z = X \cdot Y \)

and then print out \( \rho(X, Y) \). (In this case, do not round to 4 decimal place, since it will be very close to 0.)
In [7]: # (C) solution
\[ \rho(X,Y) = -1.0 \]
\[
\rho(X,Y) = -4.24138709996e-05
\]
(D) Your turn! Now I would like you to experiment with signals in the following form:

**Signal 1:** \( X(k) = \sin(x/10) \) for \( 0 \leq k < 315 \).

**Signal 2:** \( X(k) = \sin(x/scale - lag) \) for \( 0 \leq k < 315 \).

**Signal 3:** \( Z = X \cdot Y \)

where \( scale \) and \( lag \) have various values; \( scale \) will expand or contract \( X \) along the \( x \)-axis, and \( lag \) will shift it to the left or right along the \( x \)-axis.

Be sure to try setting \( lag \) to 0 and try various values of \( scale \), and also setting \( scale \) to 1.0 and trying various values of \( lag \).

In each case, examine the signals and also print out \( \rho(X, Y) \), and then answer the following questions:

1. What happens when you fix \( scale \) at 1 and move \( lag \)?
2. What happens when you fix \( lag \) at 0 and change \( scale \)? Be sure to try integer values and also floats (e.g., 3.45, 1.83, etc.).
3. Supposing we set \( scale \) at 1 and vary \( lag \). For which values of \( lag \) does \( \rho(X, Y) \) most closely approach 1.0?

**Solution:**

1. In this case, the signal shifts back and forth along the \( x \)-axis, and \( \rho \) seems to vary quite a lot.
2. In this case, the signal expands and contracts, and it seems that \( \rho \) is close to 1 when \( scale \) is an integer, and close to 0 otherwise.
3. It seems to be at multiples of 63.
Problem Two (Correlation of Signals with Lags)

If we advance a signal, say by moving it lag time units to the left along the x-axis, then we have a lagged signal. If we have a formula for a signal, we can calculate this by simply subtracting the lag from the time before calculating the signal, as we did in the previous problem.

Sometimes this happens accidentally, say in YouTube, when the audio is out of sync with the video. In this problem we will do it deliberately, and investigate what happens to the correlation between a signal and a lagged version of itself.

Unfortunately, we are not always given a formula for a signal; sometimes we have a recording of an actual sound or some other real phenomenon, and we have to manipulate the original signal.

In order to compare such signals with lags, will do something a bit drastic: we will make a copy of the signal, and delete the first lag values from the front of the signal (thereby starting it at the value X[lag]):

\[ X_{\text{lagged}} = X[\text{lag}:] \]

Now if we do this, then it is shorter than the original signal, so we will chop the same number of values from the end of X so we can compare them:

\[ X_{\text{chopped}} = X[\cdot:\text{-lag}] \]

as illustrated here:

(A) Complete the following code template to take a time series, a signal, and a lag value, and return the chopped and lagged versions of the signal.

Hint: You need to be able to input a lag of 0, but this won’t work with the Python code shown above, because X[::0] will give you the empty list! So make a special case for the case of lag = 0.
In [9]:

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 2, 3, 4, 5, 6]
[3, 4, 5, 6, 7, 8, 9]

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

(B) Print out the graphs comparing \( X \) with a lagged version, where the lag = 10:

Signal 1: \( X' = X \) with 10 values chopped from the end

Signal 2: \( Y = X \) with 10 values chopped from the beginning

Signal 3: \( Z = X \cdot Y \)

and then print out \( \rho(X', Y) \).

In [10]:

\[
\rho(X, Y) = 0.55908215311
\]

(B) Now do the same thing, but experiment and find a lag value that produces a \( \rho \) value as close to 0.0 as possible. You may ONLY use integer values for the lag, so you won’t get exactly 0, but come as close as you can.
In [11]:

\[
\rho(X,Y) = 0.00644448046226
\]

(C) Now do the same thing, but experiment and find a lag value that produces a $\rho$ value as close to 1.0 as possible.
Problem Three (Autocorrelation of Signals)

The autocorrelation of a signal is the correlation of the signal with a lagged version of itself. This autocorrelation can be used to determine whether the signal is periodic, i.e., repeats at fixed intervals, such as with a sine wave.

(A) Now we would like you to graph the correlation between a signal and a lagged version of itself, for lags from 0 to N/3. Do not draw the graphs for each signal, but just complete the template below and plot the ρ against the lag for the given signal X for lags = 0, 1, 2, ..., N/3.

ρ(X,Y) = 0.999857881243
(B) Repeat for the following signal
In [14]:

(B) The period of a periodic (i.e., repeating) signal is the time period between successive repeats of the pattern.

- What is the period (approximately) of the signal X in (B); and
- What do the maxima of the autocorrelation curve correspond to?

Solution:

- The period seems to be at 50, since at each lag of 50, 100, 150, etc. the autocorrelation peaks at 1.0
- The peaks or maxima of the autocorrelation curve correspond to lags in at which the signal repeats.

Problem Four (Fundamental Frequency and Peak Picking)

As we have just seen, the autocorrelation of a signal can be used to determine whether the signal is periodic, i.e., repeats at fixed intervals, such as with a sine wave.

The fundamental frequency of a periodic signal is the smallest lag at which the signal repeats, which corresponds to the smallest lag at which there is a maximum in the autocorrelation curve. This represents the smallest possible period for the signal. Any other period will be a multiple of this shortest period, hence the name fundamental.

The maxima or peaks in any signal A are locations t at which


as shown in red in the next figure:
These peaks can be easily identified with a scan through the signal. Note that we are calling the autocorrelation curve \( \hat{A} \) a *signal* because that is what it is!

(A) Rewrite the function from the previous problem, filling in the following template, and adding code to determine the peaks of the form \((t, A[t])\) (where \(A\) is the autocorrelation signal) and graph them in red together with the autocorrelation curve. Return the list peaks as the result of the function.

In [15]:

![](signal.png)

In [16]:

![](autocorr.png)
(B) Now a common heuristic is to use the first peak that exceeds a given threshold, say 0.9. For the next example, scan through the peaks returned by the function, and output the first \( t \) value for a peak where \( A[t] > 0.9 \). Express this as “Period found at \( t \) time units” for the \( t \) you found.

In [17]:

![Signal X](image1.png)

![Autocorrelation of X](image2.png)

Period found at 50 time units.

(C) Repeat the same experiment, but using the following signal, and checking for lags from 0 to 400:

```python
In [17]:
```
Problem Five: Fundamental Frequency of Some Real Signals

In this problem we will try to determine the fundamental frequency of some actual real-world signals. Apply the same technique just shown to determine the fundamental frequency of each of the signals below, and then translate it into Hertz (cycles/second) by the following formula, if the period is found at $P$ time units:

$$\text{fundamental frequency in Hertz} = \frac{(\text{time units} / \text{second})}{P}$$

In all the sound files we will experiment with, there are 44100 time units / second, therefore the frequency will be $\frac{44100}{P}$ Hz.

(A) This signal is an actual sound file, but of a single sine wave. Use a threshold of 0.9 and print out the fundamental frequency. Also, LISTEN to the file using iTunes or other audio player!
In [20]: # (A)

Signal X

![Signal X plot](image1)

Period found at 490 time units.
Fundamental Frequency = 90.0 Hz

Autocorrelation of X

![Autocorrelation plot](image2)

Period found at 490 time units.
Fundamental Frequency = 90.0 Hz

(B) The next file is of my 1959 Martin steel-string guitar, where I'm playing the low A string. Determine the fundamental frequency of the signal. Use a threshold of 0.9. Listen to the file as well!
In [21]:

(C) This is a recording of a bell; unfortunately, we can not always use such a high threshold as 0.9; for this one, you need to set the threshold to around 0.7.

Period found at 200 time units.
Fundamental Frequency = 220.5 Hz
In [22]:

(D) This is a recording of the radio signal of the pulsar PSR B1937+21, the dead neutron star left after a supernova, and which has the mass of about our sun but the size of Boston, rotating rapidly and emitting pulses of radiation. How fast is it rotating? Fast enough that the surface of the pulsar is moving at 1/7th the speed of light! And this is only the second fastest pulsar that has been observed!

Run the code and find out its rotation speed in Hz. Use a threshold of 0.9.
In [23]:

Period found at 64 time units.
Fundamental Frequency = 689.0625 Hz