Running the OSOP 'noiseestimate.py"

Note: This program is proprietary to OSOP and should not be shared without permission.

This is a Python program which, under Windows, must be run in the interpreted mode. After it is completely debugged it may be possible to make it into a stand-alone .exe.

It makes use of the library stacks 'numpy', 'matplotlib' and possibly 'scipy'. Also it uses the python seismic library 'obspy'. It works with Python version 2.7x, (but probably not 3).

I found that the easiest way to install Python and the libraries is to use the 'Anaconda' free Python distribution that installs many useful libraries including all those needed by noiseestimate, excepting for 'obspy'.
See: https://store.continuum.io/cshop/anaconda/
Download the appropriate Windows installer and run it. I would recommend using the 32bit version even on a 64 bit machine, as there are a few Python functons (probably not used here) which only work with 32bit Python. It should preferably be installed in a directory of the root C:\ such as C:\Anaconda or C:\Python27 rather than in C:\Program Files or C:\Program Files (x86).

You then need to make sure that the Windows environment variable 'path' includes your Python root directory and its '\Scripts' subdirectory. Open a command window and enter 'path' to see if those two directories have been included. If not, add them by entering something like: **`path %path%;C:\Python27;C:\Python27\Scripts`**
where, in this example, Python27 was the Python root directory.

Obspy may be found at
https://github.com/obspy/obspy/wiki
Installing obspy is not quite as automatic as Anaconda, but I followed the directions, including those for installing some of its modules from the Command line.

After everything was properly installed, I made a subdirectory in the Python root called 'Noiseestimate' to contain everything associated with noise plotting. Unzip the noiseestimate files into that directory. In it will be subdirectories 'data' which contain the .mseed recorded data files, along with the files containing the instrument response poles and zeros and normalization CONSTANTs. The subdirectory 'models' contains data for plotting the various low and high earth noise models as well as for plotting the approximate ground motions from earthquakes of various magnitudes and distances. I created a subdirectory, 'Results', where I could save the output plot images as .png files.

The noiseestimate program consists of 'noiseestimate.py', the main program, and 'setup.py'. setup.py is used like an 'inf' file to specify the filenames of the data and response files to be processed, the plot range, as well as True/False switches to define what information will be plotted. This is a text file which may be edited with a text

editor, or better, with the 'idle' Python text editing program which should have gotten installed along with Python.

In addition, ascii2mseed.exe converts ascii data files having header lines with a defined format into the .mseed data files required by noiseestimate.py. SDRmanip is first used to extract data from the WinSDR 'sys' daily record files into the SEIFE ascii format. Then, for now, their headers must be manually edited into the form required for ascii2mseed. I am hoping to soon be able to somewhat automate the entire process of getting from WinSDR 'sys' files to the mseeds. SDRmanip may also be used to decimate the data to a lower sample rate, which only needs to be greater than ~2.1x the highest frequency you want to evaluate. The latest versions of WinSDR now provide satisfactory anti-alias filtering when decimating.

I have not tried to use bdf2mseed.py.

When editing 'setup.py', note:
The three SAC_PZs_Data(n), instrument response files must be listed in the same order, as their corresponding .mseed data files. You can check when running noiseestimate that the "Matched file pairs.." it lists are in the expected order.

The .mseed data files may be named as desired.
SAC_PZs_Data(n) must have names of that form.

The 'CUTOFF' low-frequency plot limit must be in the same units, Hertz or Seconds, as specified in XMODE.

When PLOT_HOLCOMB is True, noiseestimate.py will not run.

The '#' sign indicates the beginning of a Python comment.

I have made a number of changes to the two original .py programs. To keep things straight, I have used '###' to comment out lines I wanted to change, and have appended '###' to any lines I added.

Note that the generator constants for Napas 1, 2 & 3, as reflected in the values used for CONSTANT are not totally correct. Correct values would have the ground noise PSD curves almost exactly overlaying each other. Given one instrument of known calibration, this is a rough but effective way for checking the calibrations of the two others.

Noiseestimate may be run by simply double-clicking it in a Windows Explorer window, so long as the .py extension has previously been associated with Python.exe.
Or, in a Command window, starting from the 'noiseestimate directory, it would be run by entering **`python.exe noiseestmate.py`** It could also be run using NE.bat, by simply entering **`NE.`**

Theplot .png image files will not be saved automatically, though they can be saved from the Photo Viewer window.