**DSP package tutorial**

**Purpose:**

To provide guidance in the use, care and feeding of the Python DSP package written by Marie Roch and maintained by Marie Roch, Kaitlin Palmer and Scott Lindeneau.

**Overview:**

This tutorial provides some guidelines on the functions available in the DSP classes and some basic applications. Assumes the required pacakges, as listed on on the BitBucket website for the package have been installed and tested. These tutorials includes helper functions that both MR and KJP have written to help process the sound files.

In the first tutorial we will implement a random sample method that outputs 2000 sound sample from a list of files. The input will be a list of the 4 file days randomly selected from the DCLDE 2013 project. The project consisted of fully hand-browsing four days of continuous recordings for the presence of right whale upcalls. Data were collected from the Massachusetts coast and consist of a single channel. The files we will be loading include 96 continuous 15 minute sample files per day for a total of 384 files in total- more than could be reasonably loaded in memory.

All code is listed in the ‘Tutorial.py’ file in the head of the DSP package. Descriptions of the code can be found both in this document as well as the code itself.

**Helper Functions:**

At the beginning of code are two helper functions that I have written and excessively commented for clarity. The first function ‘get\_start\_timestamp’ is for extracting the start time of the start file assuming that it is imbedded in the filename. The start time of the sample file is one of the 4 inputs into the streamer. Presently the function is only capable of handling two file formats (scripps and Cornell) but could easily be updated to a more flexible approach.

The second function at the start of the code, ‘MakeSoundStream’ combines files from multiple folders into a single sound stream. This is rather the crux of the code for the sample files and care should be taken in in understanding what is going into the code.

**Tutorial 1- SampleStreams:**

This tutorial uses the Streams and SampleStream class. I have found them to be the most useful classes presently implemented in the DSP package because they can use a variety of different methods to return raw samples from multiple files. As per the code comments, Streams is an object containing the meta data about the soundfile(ish). We need to combine multiple streams to produce a SoundStream.

In this section the soundstream is loaded in two ways. First, the files from a single folder are loaded by first declaring a Stream object, then using Stream.add\_file() to add multiple files and finally SampleStream() to create an object capable of returning sample files from anywhere in the stream without the need to load full sound files.

The second uses the helper function makesoundstream() to load all files from the ‘DaySoundFolder’ structure into the stream. More generally, this function allows a soundstream to be made from a list of folders containing sound files.

From here the following methods are described:

Moving the pointer using set\_sample where sample is the Nth data sample in the sample stream

Moving the pointer using set\_time where time is a datestimestamp of a sample in the stream

Loading N samples into memory

Loading N samples into memory but using advance (e.g. hopsize). M samples previous and O samples next. Where M + O =N

**Beware MultiChannels!**

These objects are capable of processing multichannels and the above methods will return data from all channels in an M x N array where N is the number of samples set by the user and M is the number of channels in the datafile. There is currently a ‘set\_channel’ method in the class but it has not been implemented yet. So be careful!