Project Audicon Progress Report 2

What's been done:

- Collection of a **vast** amount of uncompressed music, cataloged by genre
- I wanted this program to be able to handle sound data played at any sampling rate, so each analysis function I've created has been designed around that methodology. For instance, the `zerocrossings()` function doesn't just give total number of sign changes across the waveform, but gives it in the form of zerocrossings per second.
    - The following functions were designed to solve problems pertaining to input and preprocessing:
        - Wrapper function for `fft()` (fast Fourier transform) that will produce a normalized vector of real unweighted amplitude values, plus a vector of the cycles-per-second values of their associated component waveforms
        - [One-line] function for normalizing PCM WAV sample vectors
        - Function for splitting large block of data into smaller, uniform chunks by duration in seconds
            - Operates on two modes:
                - Mode 1 (default): gives one chunk of data beginning at `wavdata(1)`, second chunk beginning at `wavdata(2)`, etc.
                    - *my favorite method of splitting*
                    - **+** more data input for your byte
                    - **+** takes into account **every** sample with no truncation of non-conforming chunks
                    - **=** lots of redundancy (over 99.99% for CD audio)
                    - **-** memory-intensive
                - Mode 2: straightforward splits; discrete chunks
                    - **+** less redundancy
                    - **+** less memory-intensive
                    - **-** data loss can occur at end of WAV data
                    - **-** not programmed yet (will probably take up one line)
    - The following are waveform feature functions I have created. They will be used as input values in the neural network to be created.
        - Function for computing the number of *zerocrossings* (sign changes) per second for each wave chunk passed it.
        - Function for computing the *spectral flux* (sum of change of each frequency's amplitude [weighted by multiplying by frequency], from the previous chunk) of each wave chunk passed it.
        - Function for computing the *centroid* feature for each wave chunk passed it.

What still needs doing:

- Find a suitable neural network library for `Scilab` (or convert source from `Matlab`).
- Finish writing feature functions.
- Write chord detection algorithm

    - write signal vs noise detection algorithm
    - find out somehow to add detected chords to neural inputs.
- Design changeable preferences structure for versatility & for the front-end.
    - Default notes definition for note & chord extraction
    - Allowed deviation of note frequencies
    - Chord definitions (1-3-5 standard & 1-4-5 blues chords)
- Learn basic TCL in order to write a front-end
- Optimize for memory consumption
    - tested reading a 30-second 14.4 kHz WAV and it took up 21 megs!
    - Create wrapper function as a go-between for memory and disk cache

How I feel about the whole thing:

- I feel like I'm losing my mind. I feel like I'm unable to work for more than 20 minutes at a time on this project. I feel unmotivated and ... I feel like NOBODY in the world cares if I complete this, the project that has been the sole creative focus of my life for the last few months.
- I feel like I have embarassingly little to show for my efforts. I feel like every breakthrough I have took a lot longer than it should have, and is so miniscule that I may as well have been spinning my wheels.
- I feel alone. It's never bothered me before, but it's very frustrating when I ask someone about some minor aspect of the project, and end up having to explain to them WHAT digital music IS, or some such basic knowledge.