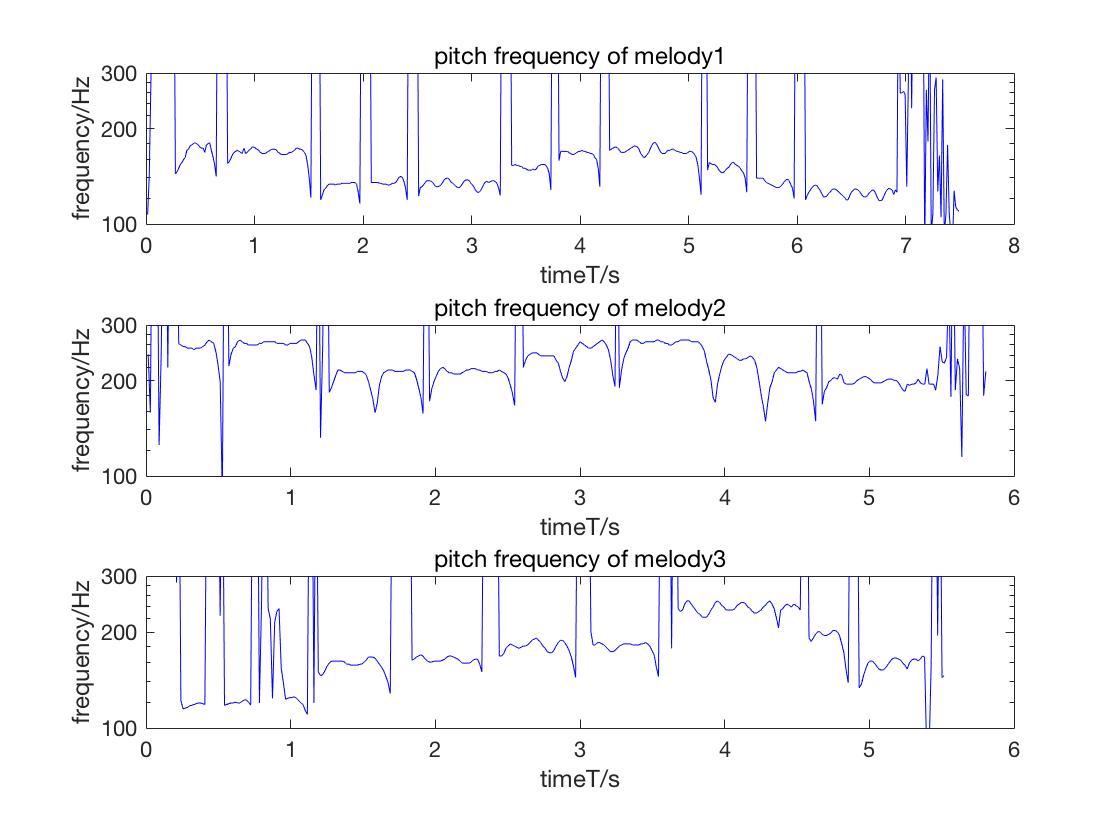
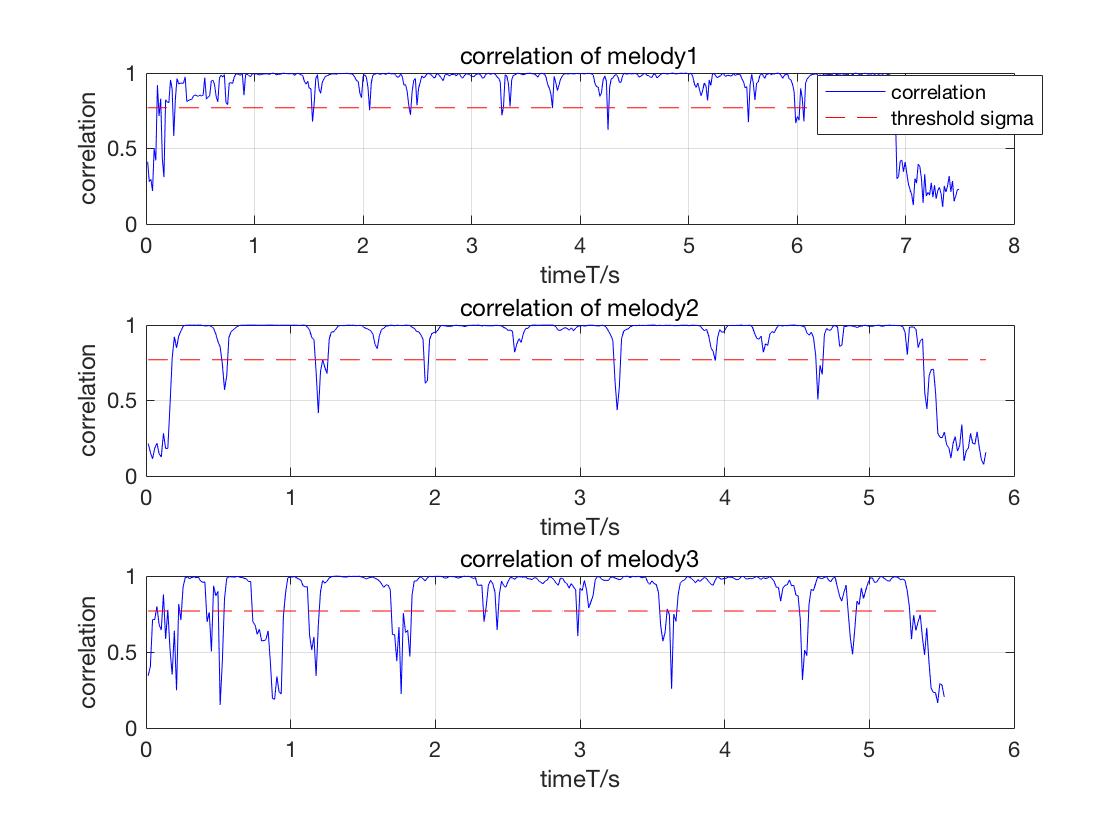
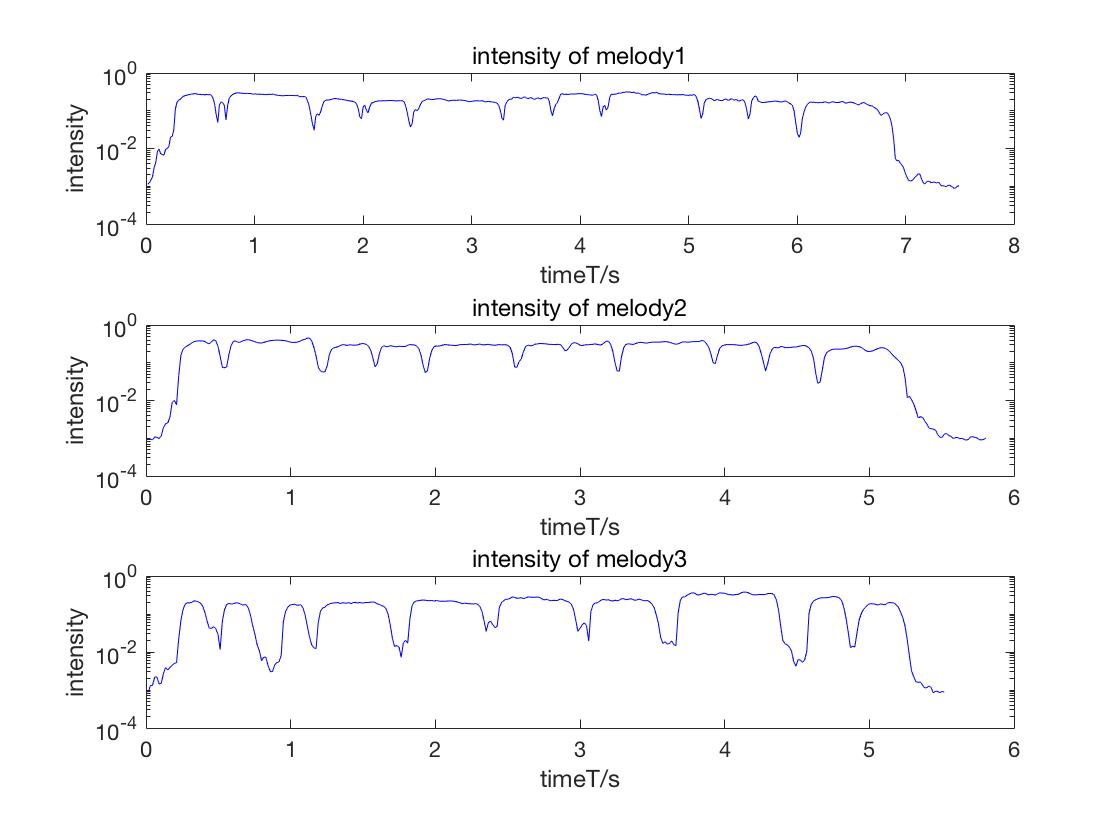
1. **Pitch, correlation and intensity for 3 melodies are listed respectively.**





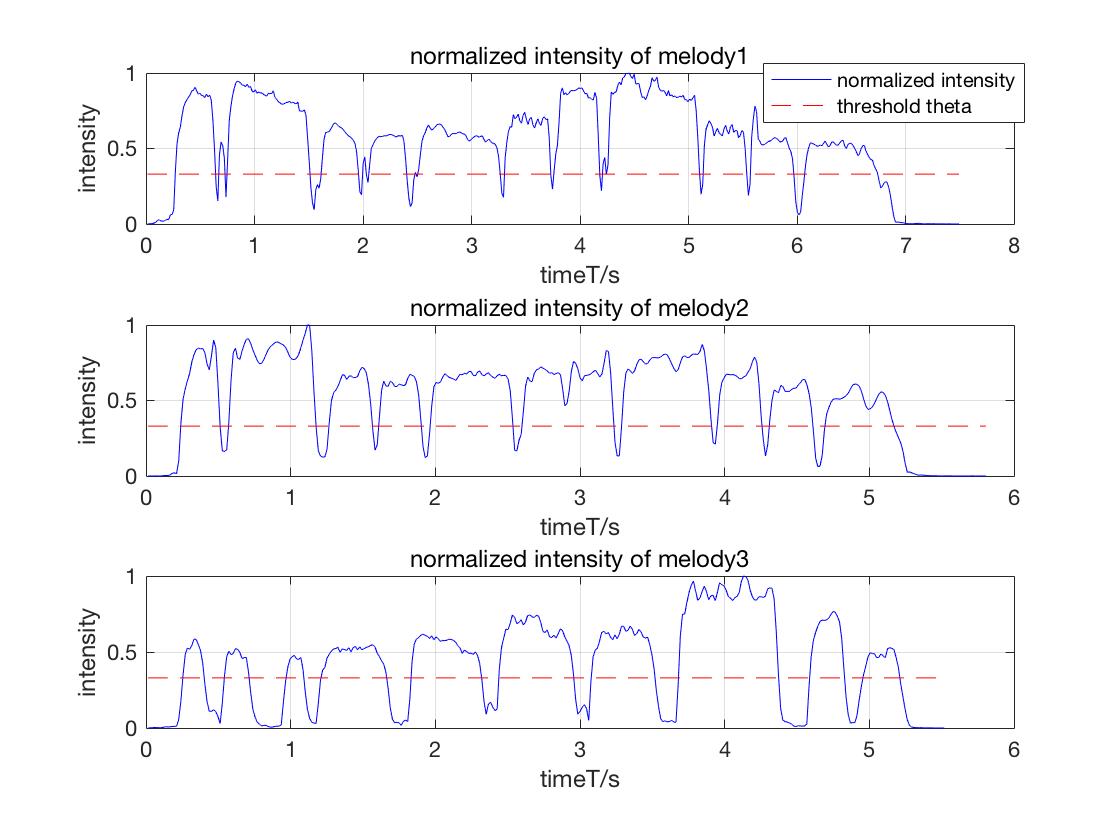


Normalize the melody intensity since we need to find a threshold for all kinds of melodies with various intensity. Then it will be more obvious to find a relatively reasonable threshold for non-harmonic region(silent/noise).

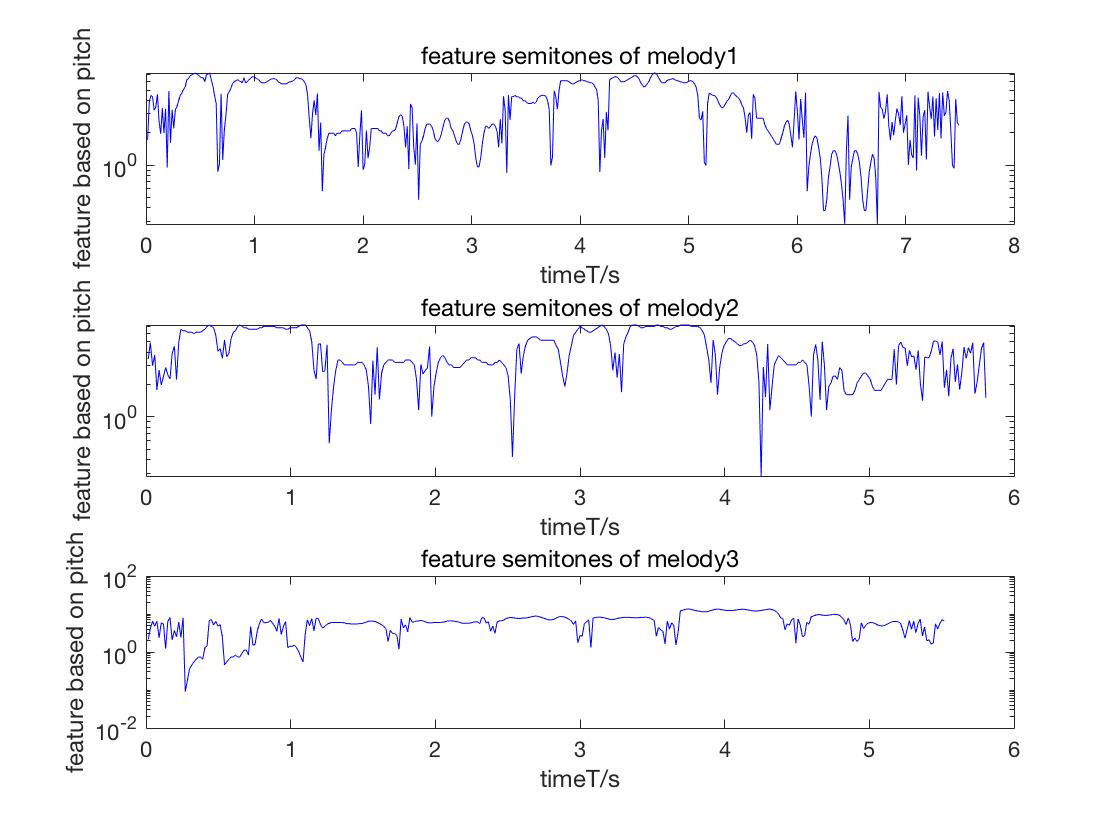
How to get the threshold for correlation and intensity:

Take the correlation for example, here we set a bound for correlation = , any correlation is beyond this bound, we believe it is melody instead of noises.

Then we calculate the mean of the left correlation of snippets, and set this mean as threshold for correlation.

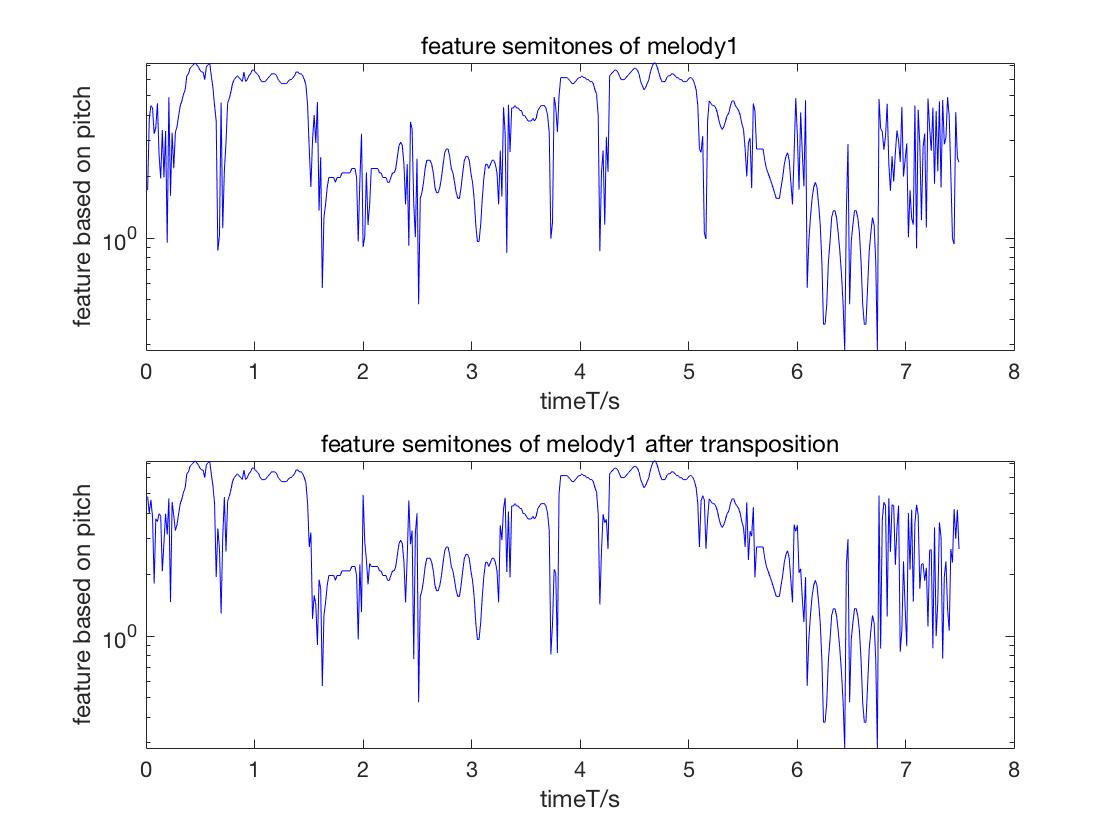


1. **feature extractor design:**
2. Based on the figure 2,4 above, use the threshold to first eliminate the non-harmonic components in the pitch, set those as low-bound. Then we get a new pitch sequence with almost harmonic regions kept.
3. Using the sequence obtained in step1, first logarithm them. Then find the minimum frequency except for low-bound, and extract it from the whole sequence which ensures that this algorithm is robust to transposition.
4. For the low-bound components, we add noises to them. Here the mean and the standard deviation are obtained from the sequence in step 2. Then we get the new sequence which takes noises into consideration as well.
5. Since the lowest octave is 20Hz-40Hz, and in logarithmic scale it has 12 equal length semitones, which indicates that the length of every semitone is equal to (log(40)-log(20))/12. Using this knowledge and the equation, we divide the sequence from step 3 by log(2)/12, the returned result is how many semitones are there between this frequency and the lowest frequency. And this is also our output feature. the following graph is the output feature from melody1,2,3 respectively, which is quite obvious that the first and second graph have a lot similarities.



1. **Transposition**

Here multiply the pitch track returned by GetMusicFeatures by 1.5, in order to test this scheme can cope with the transposition. Plot the feature from original melody and its transposed melody into the figure below, it is quite apparent that output distributions are approximately same. There is still some differences between two outputs, the reason is that noises are given randomly.



1. **possible cases this feature extraction scheme can not cope with**

from my perspective, the main problem will occur at threshold set. For example, when the original song volume is low, then intensity variation between melody and silent/noise is not apparent. So using threshold will delete non-harmonic region as well as melody snippets.