Engin. 100: Music Signal Processing Project #3: Specifications and Hints

- Specifications for Music Synthesizer
- Specifications for Music Transcriber
- Suggested Approaches for Transcriber

Project #3 Deliverables: Synthesizer

- Music synthesizer: Can synthesize one octave.
- Instruments: Electric guitar, trumpet, clarinet, Design your own using *additive synthesis.*
- Pull-down menu to select the instrument first.
- Durations: whole, half, quarter (length) notes.
- Can mix instruments together (play DJ here): Lay down tracks *separately* using different instruments, then add them together in Matlab.

Music Synthesizer: Generation

- Download *proj3.wav* from web site. Contents:
- Snippets: Length=32768, sample=44100 Hz.
- Electric guitar, clarinet, trumpet, tone; 12 notes.
- **NOTE**: I generated notes using Circle of Fifths and multirate filtering, so frequencies slightly off.
- Additive Synthesis: Create your own instrument, label it with your team name. Be creative here!
- Marching band: Reverb (add copies) of trumpet.

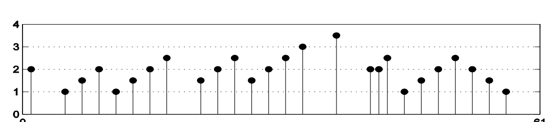
Music Synthesizer: Specifications

- Pull-down menu to select instrument. Use:
- H=uicontrol('Style','Popup','Position',[500 250 100 50], 'String','guitar|clarinet|trumpet|tone');pause;I=get(H,'Value');
- I=1→guitar;I=2→clarinet;I=3→trumpet, etc.
- Final 100 samples of each note should be 0, to assist transcriber in detecting changes of notes.
- To reverb or echo a length=N sampled signal X: Y=X(1:N-2*D)+X(1+D:N-D)+X(1+2*D:N); for a delay D≈1000; use many more than 3 echoes.
- Table on next slide; figure on slide after next.

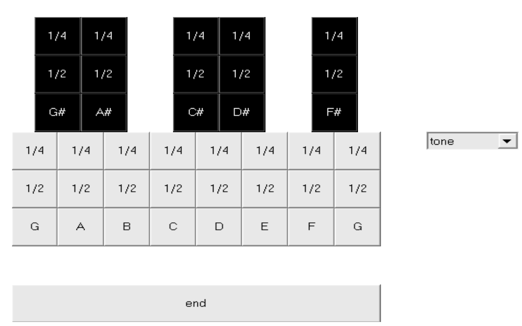
Music Synthesizer: Specifications

NOTE	1 sec.	Whole Note	Half Note	Quarter
LENGTH (end in 0s)	44100	32668+100 final zeros	16284+100 final zeros	8092+100 final zeros

Example of music **transcriber** output. Note interval information.



Music Synthesizer: Specifications



Engin. 100: Music Signal Processing Project #3: Specifications and Hints

- Specifications for Music Synthesizer
- Specifications for Music Transcriber
- Suggested Approaches for Transcriber

Project #3 Deliverables: Transcriber

- Accepts .mat or .wav file from your synthesizer.
- Generates musical staff notation as in Project #1.

- BUT: Also depicts note duration by separation.
- BUT: Must work on *music*, not just *tones*!

- Otherwise, same as Project #1 transcriber specs.
- Does not have to include a bass scale for guitar.
- ALSO: Error rate vs. SNR plot, as in Project #2.

Music Transcriber: Specifications

- Output: Musical scale & notes using stem (Project #1)
- Duration: Shown in output by separation between notes:

Note Type	Whole	Half Note	Quarter
Separation	3 spaces	1 space	0 space

- Use reshape, columns ending in 0s give note lengths. T=indices of those columns; stem(T,12log$_2$(F/440)) where F=vector of estimated note frequencies which are then mapped to musical staff notation, as in Project #1.
- Don't need to be able to handle the (bass) guitar tones.
- Do need an error rate vs. SNR plot, as in Project #2.

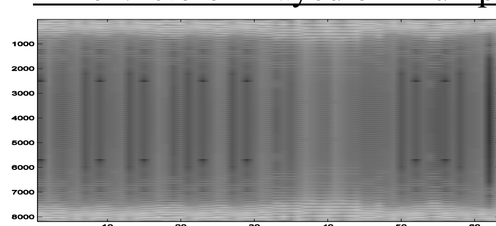
Engin. 100: Music Signal Processing Project #3: Specifications and Hints

- Specifications for Music Synthesizer
- Specifications for Music Transcriber
- Suggested Approaches for Transcriber

Transcriber: Possible Approaches

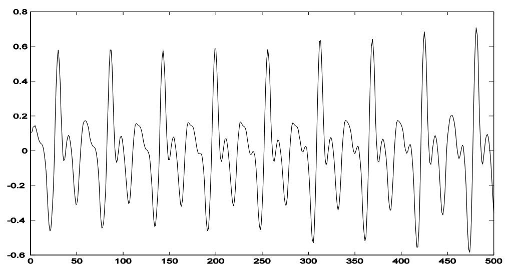
- Spectrogram: Look for peaks. Hard to program.
- Fundamental frequency identification from a limited choice of already-known frequencies.
- Autocorrelation of segment with itself: y[n]=Σx[i]x[i-n] has sharp peak at n=period.
- Harmonic Product Spectrum: Downsample and multiply spectra-this emphasizes 1st harmonic.
- All of these have been tried previously.

"The Victors" Played on Trumpet



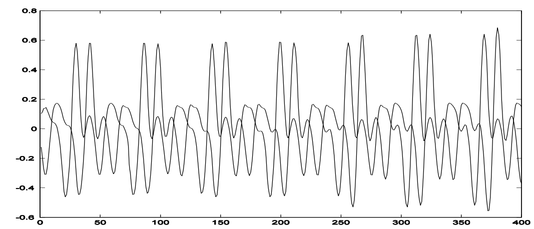
imagesc(10+log(abs(fft(reshape(Y',8192,length(Y)/8192))))), colormap(gray). Plotting 10+log(values) reduces dynamic range. Zoom in on upper (or lower) portion to see harmonics clearly.

Example: Trumpet Waveform



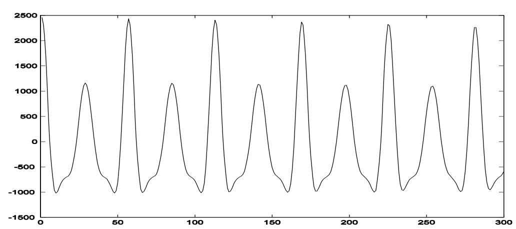
Pitch detection: Find period of periodic waveform. But how?

What is Autocorrelation of x[n]?



$y[n]=\Sigma x[i]x[i-n]=\Sigma x[i]^2$=large when n=a period.
Compute:Y=real(ifft(abs(fft(X,2*length(X)).^2));

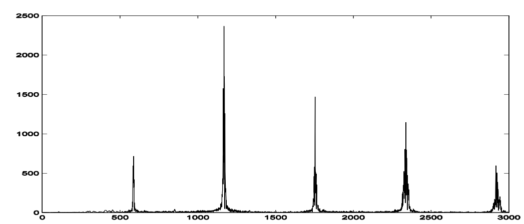
Example: Trumpet Autocorrelation



Y=real(ifft(abs(fft(X,2*length(X)).^2));plot(Y(1:300)).

Second peak at index=57→Frequency=44100/(57-1)=788 Hz

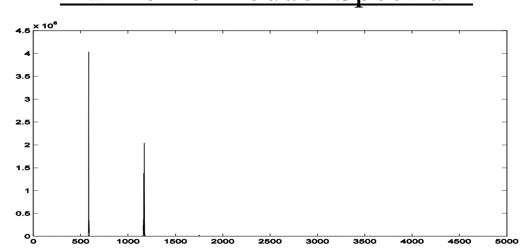
Example: Trumpet Spectrum



Fundamental at (587-1)44100/32768=788 Hz. Use Project #1.

BUT: 2^{nd} harmonic > 1^{st} harmonic! Misread as fundamental!

Harmonic Product Spectrum



F=abs(fft(X));plot(F(1:5000).*F(1:2:10000).*F(1:3:15000)) above.
Multiply spectrum by downsampled versions of itself (not signal).

Emphasizes harmonics over other stuff-easier to find peaks in noise.

Transcriber: Helpful (?) Ideas

- Noise filtering: If only interested in 1 octave, filter out all signal not in that octave (Lab #3). Helps for some approaches, but not for others.
- Frequencies I generated using Circle of Fifths don't match post-"Well Tempered Clavichord." Use pure tone selection→calibrate transcriber.
- Sub-harmonics (at fractions of fundamental)?

Issues Arising in Project #3

- The octave problem: Distinguish G (392 Hz) from G (784 Hz). Trumpet has this problem.
- Pattern recognition to identify instrument type from pattern of harmonics? Not required, but...
- Need to sell/defend your choice of method in both your team's final oral and written reports.

Conclusion

- I'm not telling you how to do this project! Not a solved problem-different approaches.
- Apply what you have learned in the course.
- Research on music synthesis/transcription.
- As always, the tech comm presentation of results is as important as results themselves. Very realistic for real-world engineering.