FFT/STFT cheat sheet

COS 314: 20 April 2009

FFT: Fast Fourier transform

A method for computing the discrete Fourier transform of a signal. Its "fastness" relies on size being a power of 2. Implemented as the FFT UAna in ChucK.

STFT: Short-time Fourier transform

A method for analyzing a signal whose frequency content is changing over time. The signal is broken into small, often overlapping frames, and the FFT is computed for each frame (i.e., the frequency content is assumed not to change within a frame, but subsequent analysis frames can be compared to understand how the frequency content changes over time). Not implemented explicitly in ChucK; you implement it simply by calling an FFT upchuck repeatedly over time.

IFFT: Inverse Fast Fourier transform

Takes a spectrum buffer (a complex vector) of N bins and transforms it into N audio samples. Implemented as the IFFT UAna in ChucK.

FFT size:

The number of samples over which the FFT is computed; also the number of "bins" that comprise the analysis output. Implemented as .size() method of FFT object in ChucK.

Bin:

The content of a bin denotes the magnitude (and phase) of the frequency corresponding to the bin number. The N bins of an N-sample FFT evenly (linearly) partition the spectrum from 0Hz to the sample rate. Note that for real signals (including audio), we can discard the latter half of the bins, using only the bins from 0Hz to the Nyquist frequency.

Accessible by calling the .fvals() and .cvals() methods, for receiving float or complex arrays, respectively.

Window function:

Before computing the FFT, the signal is multiplied by a window function. The simplest window is a rectangular window, which multiplies everything inside the frame by 1 and everything outside the frame by 0. However, in practice, we choose a smoother window function that is 1 in the middle of the window and tapers to 0 or near-0 at the edges. The choice of window depends on the application.

Window type and size are properties of the FFT object in ChucK, set using, e.g.,

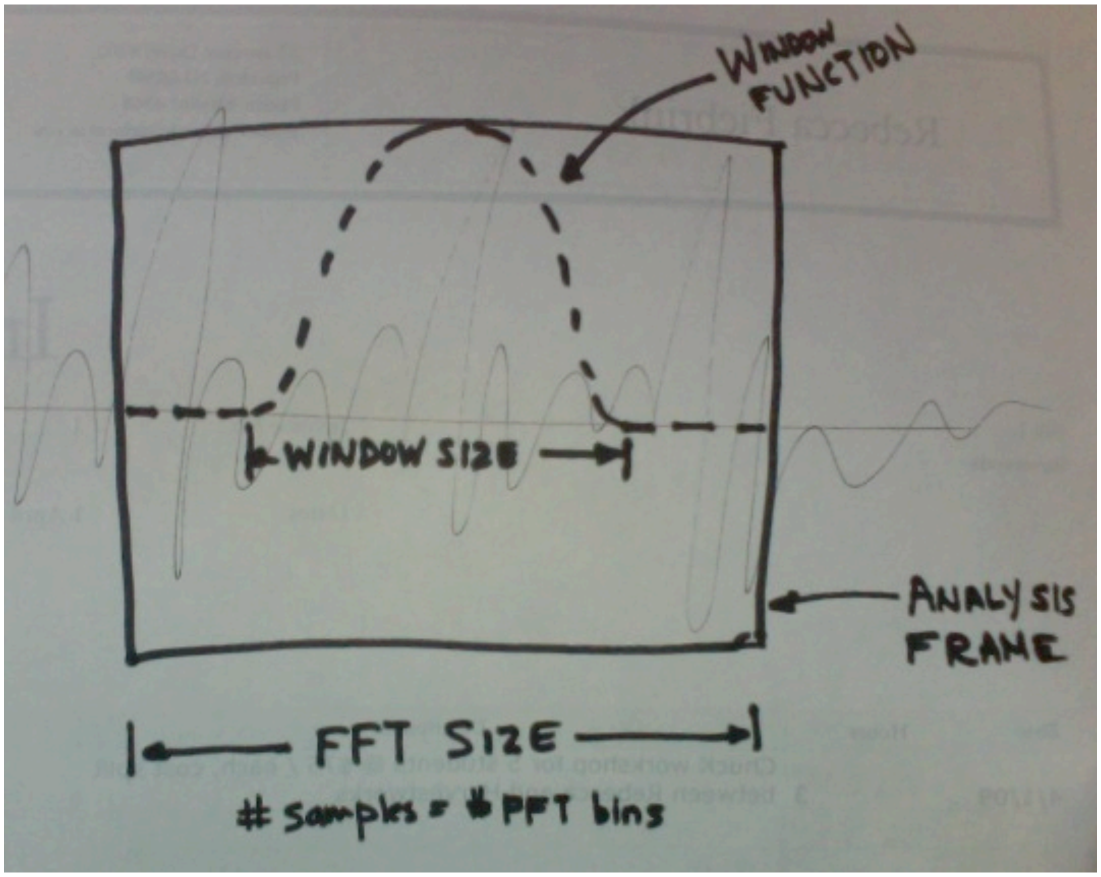
```
Windowing.hann(512) => fft.window;
```

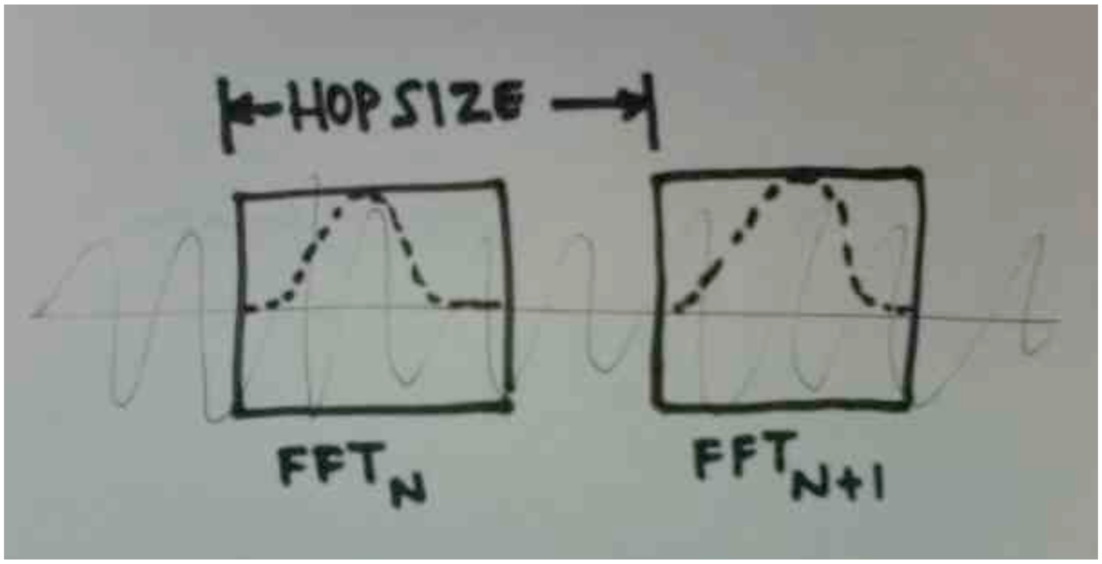
Zero-padding:

It is common practice to use a smaller window size than FFT size, then "zero-pad" all the samples that lie in between the edges of the window and the edges of the FFT frame. ChucK will automatically zero-pad if your window size is smaller than your FFT size.

Hop size:

In STFT, you must decide how frequently to perform FFT computations on the signal. If your FFT size is 512 samples, and you have a hop size of 512 samples, you are sliding the analysis frame along the signal with no overlap, nor any space between analyses. If your hop size is 256 samples, you are using 50% overlap. The hop size can be small (high overlap) if you want to very faithfully recreate the sound using an IFFT, or very large if you're only concerned about the spectrum's or spectral features' values every now and then. There is no hop size parameter in ChucK; it is implicitly defined by how much time is advanced between upchucking your FFT.


WINDOW FUNCTION
WINDOW SIZE
ANALYSIS FRAME
FFT SIZE
# samples = # FFT bins


HOP SIZE
FFT N
FFT N+1