Objects Covered Cheatsheet

Pitch Detect

Object []	What it does	Example Arguments (where applicable)
[fzero~]	Estimates the fundamental frequency of an incoming monophonic signal.	
[retune~]	Alternate method for [retune~] is to use the @pitchdetection 1 and @use_16bit 1 attributes. This outputs the estimated pitch as a signal from the second outlet on the left.	[retune~ @pitchdetection 1 @use_16bit 1]
[pipo~ yin]	Requires the MUBU externals. Slices incoming audio via the YIN algorithm implementation for monophonic pitch detection. Lots of options for [attrui], but you should start with @slice.size 1710 @slice.hop 100 @slice.wind none @yin.minfreq 50 @yin.downsampling 4. Outputs a list, the first element of which is the estimated frequency.	[pipo~ slice:yin @slice.size 1710 @slice.hop 100 @slice.wind none @yin.minfreq 50 @yin.downsampling 4]
[pipo~ psy]	Requires the MUBU externals. Pitch Synchronous YIN markers used for Pitch Synchronous Overlap and Add (PSOLA) method of monophonic pitch detection. You will want to use [attrui]'s for psy.minfreq, psy.maxfreq, psy.downsampling, psy.yinthreshold, and psy.noisethresehold. Outputs a list, the first element of which is the estimated frequency.	
[zsa.freqpeak]	From the [zsa.easy_freqpeak~] abstraction. Requires the zsa.descriptors externals. Extracts a pair of Frequency/Amplitude peaks by FFT frame. Arguments are FFT size (power of 2) and overlap factor.	[zsa.easy_freqpeak~ 2048 4]
[fiddle~]	Download from https://github.com/v7b1. Miller Puckette's classic pitch detection object. Arguments are the window size (128-2048, powers of 2 only), number of pitch outlets (1-3), number of peaks to find (1-00), and the number of peaks to output. The left most output is the estimated pitch.	[fiddle~ 1024 1 20 3]
[sigmund~]	Download from https://github.com/v7b1. Another classic from Miller Puckette, [sigmund~] analyzes and reports the sinusoidal components of an incoming sound. Common attributes and arguments are @npts (number of points, powers of 2 only), @hop (hop size, powers of 2 only), pitch, notes, and env.	[sigmund~ @npts 1024 @hop 64 pitch notes env]
[bonk~]	Download from https://github.com/v7b1. Third in the Puckette trilogy of classics, [bonk~] detects attack transients in an incoming signal. Works best with percussion for obvious reasons. Lots of attributes, but the @npts and @hop are most useful and are the same as [sigmund~]. Quite useful even without any attributes.	
[patcherargs]	Generates initial arguments in a subpatch or abstraction. Set the arguments to what you want the values to be. It will output them as a list.	[patcherargs 10. 100.]

[mstoamps~]	Converts MS to samples at the current sample rate. VERY useful for changing time formats for things like [delay~] or [slide~].	
[slide~]	Filters a signal logarithmically between a change in signal values. This lets you use the output to follow the envelope of the incoming signal.	
[zl.slice]	[zl] is the list processing function in Max. There are TONS of ways to use it, but [zl.slice] is one of the most common - give it an integer as an argument, and it slices that many elements from the front of the list and outputs them on the left. Other remaining values are output to the right. All [zl] objects can be written with periods or spaces - e.g. [zl.slice] or [zl slice].	[zl.slice 4]
[multislider]	Displays data as a collection of sliders.	