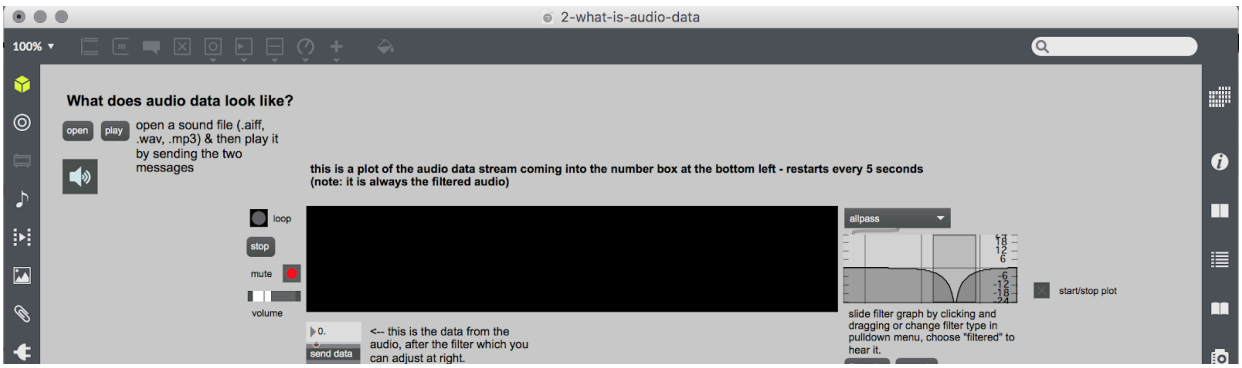
Unit 1 Lecture - Part 2

What is Audio Data?

Now that we have seen what video data looks like, it is time to look at audio data. The patch titled **2-what-is-audio-data** is a sub-patch from the Interactive Video Machine, which we will use in the next section in order to make super fun videos that are directly affected by sound. Open this patch, which is available in the Class Patch Archive.

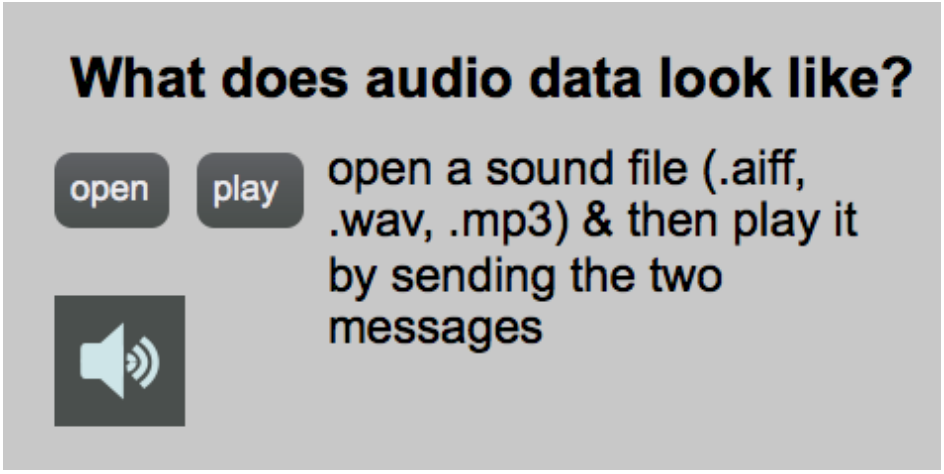
A bit of background on this patch: I created it for myself as an artistic tool….because I worked with many live electronic musicians making music on laptops, there was a natural desire to have the computers talk to each other. This was 1999 or so, and there was a lot less software out there, but regardless of that, it was not hard to just hook up the audio output of one computer to the audio input of another. I tried this many times, and I tried mapping the incoming audio to various video filters and effects - sometimes with great results, but mostly with mediocre results. Then I realized that I needed to be able to see what the audio data looked like over time, to understand how I could tell the computer to visualize it. Even though our human ears can easily distinguish the vocals from the bassline, the computer cannot. The computer “hears” the amplitude of the sound file, and for the purposes of controlling video, we just ask the computer to report the amplitude every 20ms.

When you open the patch it will look like this:



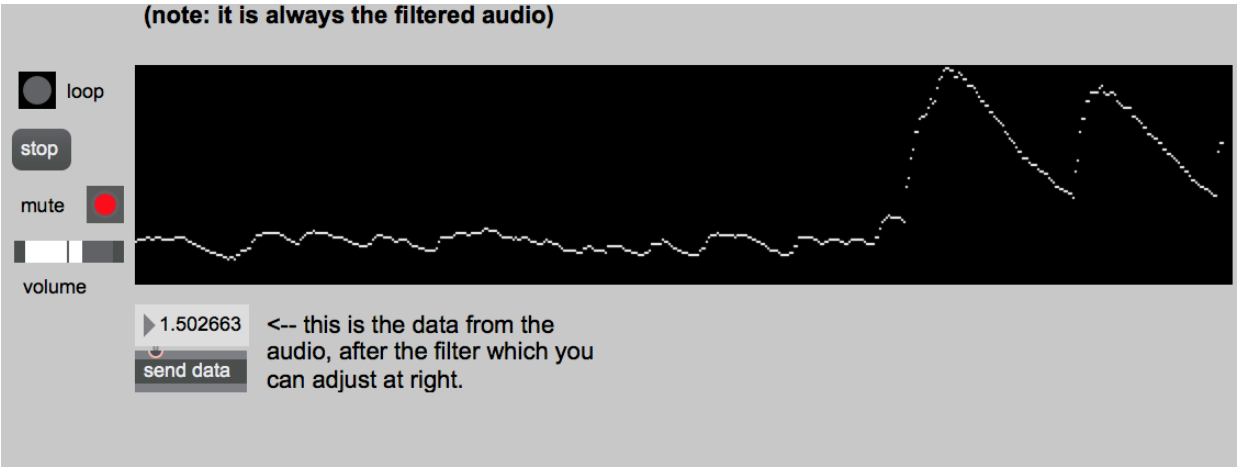
We need to send a few messages to get things going. Note that in this patch there are a lot of processes happening behind the scenes; many objects are hidden from your view. This is one of the tools for making a GUI (graphical user interface) in Max. When I give you patches that I expect you to modify on your own I do not hide things from you - but when I give you patches that are just for you to use, there will often be hidden objects. There is a different feeling to using patches with GUIs and patches without - maybe you like one style better than the other. You'll have opportunities to choose between GUI and not GUI in the future, and you will learn the basics of making your own interfaces.

Ok, on to the patch. Look at the area in the top left - and send the "open" message by clicking on it (the object you are sending this message to is hidden, but it is called sfplay~).

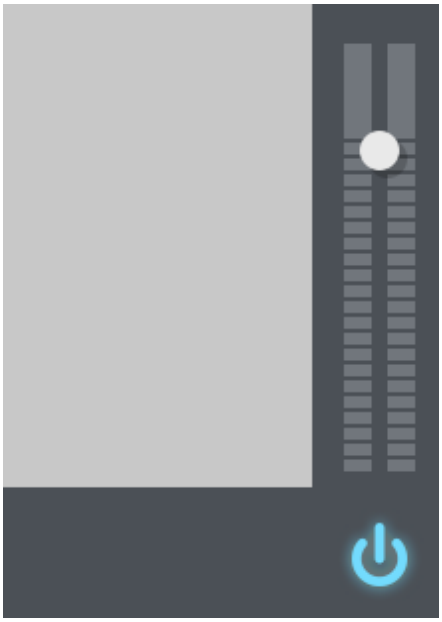


A dialog box should open asking you to choose an audio file, anything in the Class Audio files I have made available will work, or read something you already own. When that dialog box closes click on "play" - this should turn on the speaker icon, play your file and start drawing the amplitude envelope in the black area at the center of the screen.

Now the center of your patch should look like this:

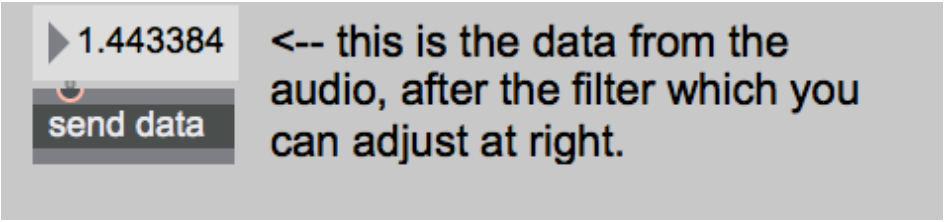


If you do not hear audio or see the audio amplitude plot then revisit the section above and try again. Make sure you are reading something from the Class Audio folder (you should have created this already and downloaded the content to it); also make sure the speaker icon is light blue, and not grey. If it is grey, click on it. One other thing to check is the area at the bottom right of the patch:



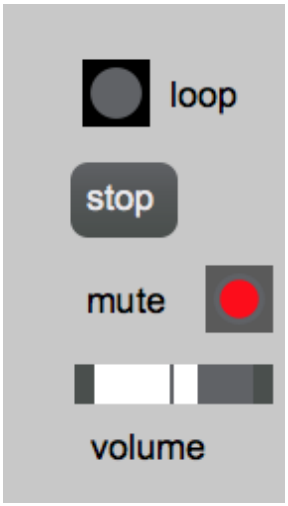
This icon is for turning the audio processor on and off for the entire program - if it is orange or grey then try clicking on it until it turns blue. If this is still not working please visit the Troubleshooting area of the content or contact me directly.

The whole reason for this patch is to turn sound into a number stream so we can use it to dynamically control video. It happens right here:



It doesn't really matter what the actual value is because we will scale it up and down as needed, what matters is that it is a representation of the sound as it changes over time. You can see how hard it is to understand the trend of a stream of numbers in a number box, but how the plot above it gives you great information on how dynamic your sound is. Try a few different files, some are more dynamic than others. I have found that many high dollar productions (Top 40 and the like) are not very dynamic - this is likely due to the desire to have all frequencies as loud as possible to get that pop music sound (any audio mastering people in class that want to share more on this?). My good friend AGF has one song in the Class Audio file and that is a fun one (she is reading old school HTML code in german, over a bare bones techno track she made).

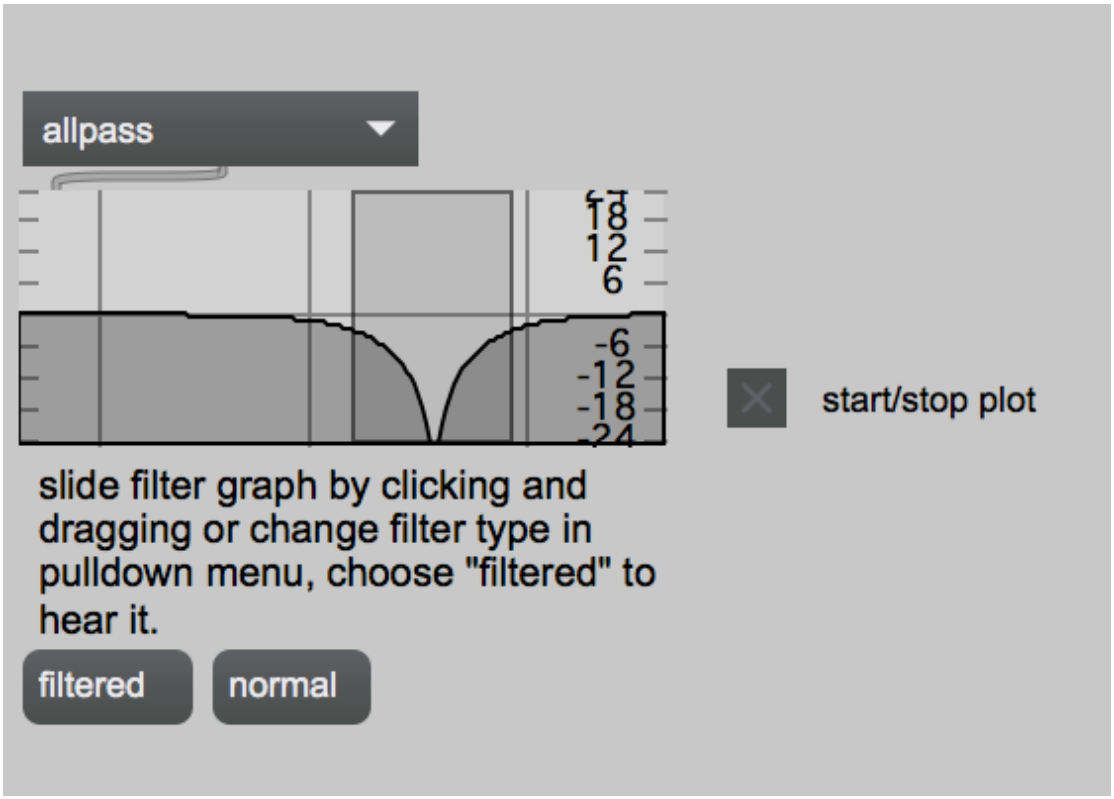
There is more that can be done here, first a few little things:



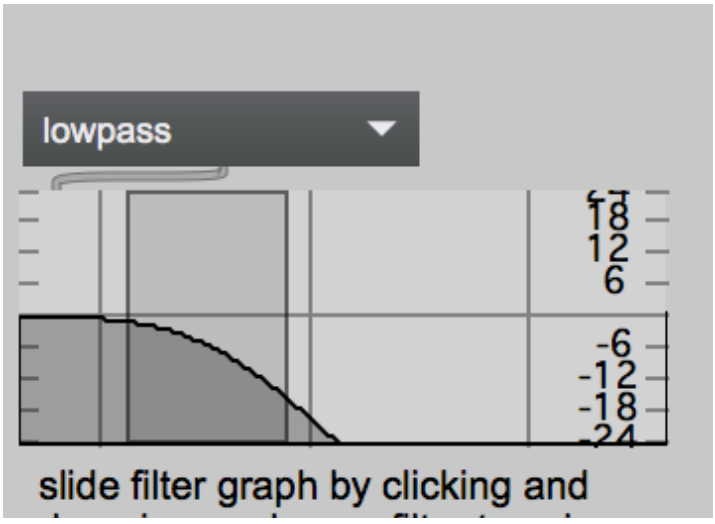
Using these objects you can loop, stop, mute and change the volume. If you do not loop, the audio file will stop after it plays through one time, and then you have to send the play message again.

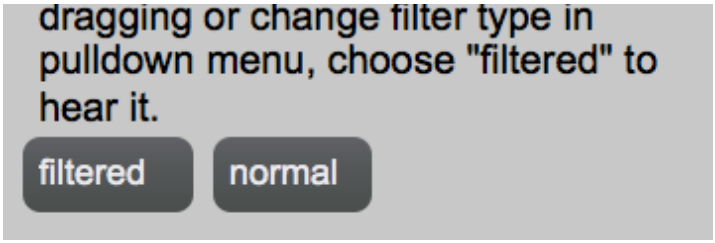
Now let's look at audio filtering and continue our discussion of how computers hear sound. When we convert the sound to a number, we are sampling the peak amplitude of all frequencies every 20 milliseconds (audio is producing data much faster than that but we do not need it for video which is normally changing frames

only 30 times per second). If we know there is interesting content within a certain frequency range, we can filter out all of the other frequencies and focus on that. Then we would be sampling just a portion of the total frequency range and we may be able to get the data from just the bassline or the vocals. This happens to the right of the audio plot:



By default when you open the patch we are using the “allpass” setting of the filter object called biquad~ (this object is hidden from view). To change the filter, choose something from the drop down menu above the small graph - I suggest “lowpass” for now. It will make your graph look like this:





The plot will immediately reflect the filtered audio, but you will not hear any difference until you click on filtered. You can play with the range of frequencies being filtered by clicking and dragging in the small graph which will move or change the shape of the filter curve. Play with this until you are convinced it is working. This is the last part of this patch.

If you are ready to combine audio data with video data then let's move on to the next patch.

Unit 1 Lecture - Part 2