Lab 10: Spectral Factorizations

Part 1. Learning Spectral Components

In this part we will design a simple component analyzer. Download the soundfile in:

http://courses.engr.illinois.edu/cs498ps3/sounds/80s-hi.wav

This is a drum loop with four distinct sounds (bass drum, snare drum, cymbal and synthetic bell sound). We will use a spectral factorization that will allow us to extract them all. Obtain the STFT of this signal and use a DFT size of 4096, a hop size of 256 and a Hann window. This will be stored in a matrix $\mathbf{F}$ whose size will be M by N.

You now need to implement a factorization technique. This is defined as:

$$|\mathbf{F}| \approx \mathbf{W} \cdot \mathbf{H}$$
$$\mathbf{F} \in \mathbb{R}_{+}^{M \times N}, \mathbf{W} \in \mathbb{R}_{+}^{M \times K}, \mathbf{H} \in \mathbb{R}_{+}^{K \times N}$$

Where $\mathbb{R}_{+}^{A \times B}$ is the set of real-valued matrices of size A times B and $|\mathbf{F}|$ takes the absolute values of the elements of $\mathbf{F}$. In this case we will use $K = 4$. To estimate the proper values for $\mathbf{W}$ and $\mathbf{H}$ assign to them a uniformly random real value between 10 and 11 and iterate over the following equations:

$$\mathbf{V} = \frac{|\mathbf{F}|}{\mathbf{W} \cdot \mathbf{H} + \epsilon}$$
$$\mathbf{H} = \mathbf{H} \odot \left[\mathbf{W}^{\top} \cdot \mathbf{V} \right]$$
$$\mathbf{W} = \mathbf{W} \odot \left[\mathbf{V} \cdot \mathbf{H}^{\top} \right]$$

where $\odot$ denotes element-wise multiplication, the fraction denotes element-wise division. For ϵ use a small number so that you avoid a division by zero in case the denominator is zero. After each pass normalize the columns of $\mathbf{W}$ to sum to 1. Iterate over this for about 100 iterations.

Plot the columns of $\mathbf{W}$ and explain what they correspond to. Plot the rows of $\mathbf{H}$ and explain them as well. You might have to run the above procedure a couple of times since in some cases the results can come up wrong. Just to be safe, run this a dozen times and show the results that are representative of the majority of the outputs (note that each time the ordering will be different, we only care about the shapes of these quantities, not their order).

You can now try to extract each component. Take each column of $\mathbf{W}$ and compute its outer product with its corresponding row of $\mathbf{H}$. This will approximate only one component of the input spectrogram. Plot all four products and explain what they look like. Use the phase of the original input to invert these resulting spectrograms to the time domain and listen to them. What do they sound like?

Part 2. Training dictionaries for source separation

In this section we will design a system that separates speech of a known speaker from a known type of noise. Get the following two soundfiles:

http://courses.engr.illinois.edu/cs498ps3/sounds/speaker.wav

http://courses.engr.illinois.edu/cs498ps3/sounds/chimes.wav

One of them is of speech and the other one of chimes. Take the first sentence of the speech sound and a segment which is just as long from the beginning of the chime sound and add them together. This will be a mixture that we will try to separate. The rest of the data we will use for training dictionary models.

Taking the rest of the speech data run a factorization as we've done above with K = 40. Do the same with the remaining chime sound. From these you will obtain two matrices $\mathbf{W}_s$ and $\mathbf{W}_c$. These are the dictionaries of the two sounds. If you visually inspect them you will see that they look a lot like representative spectra of these two sounds.

In order to resolve the mixture we need to use these dictionaries to explain its spectrogram and then only use each dictionary's contribution to resynthesize a time signal. This essentially involves finding the $\mathbf{H}$ matrix while fixing the $\mathbf{W}$ matrix to be a concatenation of $\mathbf{W}_s$ and $\mathbf{W}_c$. You can do that using the itera`tive approach used in the previous part, but only updating $\mathbf{H}$ and not updating $\mathbf{W}$ at every iteration. If you do this on the mixture you will ultimately get a $\mathbf{H}$ that will let us know how to combine the elements of the pretrained dictionaries to approximate the input.

To extract the two sounds you need to isolate the contribution of the two dictionaries on the mixture. That will be $\mathbf{F}_s = \mathbf{W}_s \cdot \mathbf{H}_s$ and $\mathbf{F}_c = \mathbf{W}_c \cdot \mathbf{H}_c$, where $\mathbf{H}_s$ corresponds to the first 40 rows of $\mathbf{H}$ and $\mathbf{H}_c$ to its second 40 rows. $\mathbf{F}_s$ and $\mathbf{F}_c$ will correspond to the magnitude spectrograms of the two extracted sources. Just as before use the phase of the input mixture to invert these back to the time domain and listen to them. Do they sound like they are separated? Play around with the STFT parameters until you get the best sounding results.