Machine Learning for Signal Processing
(ENGR E599)
Homework 3

Due date: Mar. 3, 2017, 23:59 PM

Instructions

- Submit your pdf report and a zip file of source codes to Canvas
- No handwritten solutions
  - Go to `http://sharelatex.com` ASAP and learn how to use LaTeX
- Start early if you're not familiar with the subject, programming, and LaTeX.
- Do it yourself. Discussion is fine, but code up on your own
- Late policy
  - If the sum of the late hours (throughout the semester) ¡ five days (120 hours): no penalty
  - If your total late hours is between 120:00:01 and 144:00:00 (from 5 to 6 days): you'll lose 20% of your total late homework score you earned
  - Between 144:00:01 and 168:00:00 (from 6 to 7 days): you'll lose 40% of your total late homework score you earned
  - Between 168:00:01 and 192:00:00 (from 7 to 8 days): you'll lose 60% of your total late homework score you earned
  - Between 192:00:01 and 216:00:00 (from 8 to 9 days): you'll lose 80% of your total late homework score you earned
  - If you're late by more than 9 days for any of the assignment or for some of the assignment in total, you get no points for the late submissions
- It's okay to use whatever programming language you prefer, but avoid using toolboxes.

Problem 1: Instantaneous Source Separation [2 points]

1. From `x_ica_1.wav` to `x_ica_4.wav` are four recordings we observed at an audio scene. In this audio scene, there are three speakers saying something at the same time plus a motorcycle passing by. You may want to listen to those recordings to check out who says what, but I made it very careful so that you guys cannot understand what they are saying. In other words, I

multiplied a 4×4 mixing matrix $\boldsymbol{A}$ to the four sources to create the four channel mixture:

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} s_1(t) \\ s_4(t) \\ s_4(t) \\ s_4(t) \end{bmatrix} \tag{1}$$

2. But, as you've learned how to do source separation using ICA, you should be able to separate them out into four sources: three clean speech signals and the motorcycle noise. Listen to your separated sources and transcribe what they are saying. Submit your separated .wav files along with your transcription.

3. At every iteration of the ICA algorithm, use these as your update rules:

$$\Delta \boldsymbol{W} \leftarrow \left(N\boldsymbol{I} - g(\boldsymbol{Y})f(\boldsymbol{Y})'\right)\boldsymbol{W} \tag{2}$$
$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \rho \Delta \boldsymbol{W} \tag{3}$$
$$\boldsymbol{Y} \leftarrow \boldsymbol{W}\boldsymbol{Z} \tag{4}$$

where

$$\boldsymbol{W} : \text{The ICA unmixing matrix you're estimating} \tag{5}$$
$$\boldsymbol{Y} : \text{The } 4 \times N \text{ source matrix you're estimating} \tag{6}$$
$$\boldsymbol{Z} : \text{Whitened version of your input (using PCA)} \tag{7}$$
$$g(x) : \tanh(x) \tag{8}$$
$$f(x) : x^3 \tag{9}$$
$$\rho : \text{learning rate} \tag{10}$$
$$N : \text{number of samples} \tag{11}$$

4. Don't forget to whiten your data before applying ICA!

Problem 2: Ideal Masks [2 points]

1. `piano.wav` and `ocean.wav` are two sources you're interested in. Load them separately and apply STFT with 1024 point frames and 50% overlap. Use Hann windows. Let's call these two spectrograms $\boldsymbol{S}$ and $\boldsymbol{N}$, respectively. You may want to discard the complex conjugate part, so eventually they will be an 513×158 matrix. Later on in this problem when you recover the time domain signal out of this, you can easily recover the missing part from the existing half so that you can do inverse-DFT on the column vector of full 1024 points.

2. Now you build a mixture spectrogram by simply adding the two source spectrograms: $\boldsymbol{X} = \boldsymbol{S} + \boldsymbol{N}$.

3. Since you know the sources, the source separation job is pretty simple. One way is to calculate the ideal masks $\boldsymbol{M} = \frac{\boldsymbol{S}}{\boldsymbol{S}+\boldsymbol{N}}$. By the definition of the mixture spectrogram, $\boldsymbol{S} \approx \boldsymbol{M} \odot \boldsymbol{X}$.

4. Sometimes we can only estimate a nonnegative real-valued masking matrix $\bar{\boldsymbol{M}}$ especially if we don't have an access to the phase of the sources. For example, $\bar{\boldsymbol{M}} = \frac{|\boldsymbol{S}|^2}{|\boldsymbol{S}|^2+|\boldsymbol{N}|^2}$. Go ahead and calculate $\bar{\boldsymbol{M}}$ from your sources, and multiply it to your mixture spectrogram, i.e. $\boldsymbol{S} \approx \bar{\boldsymbol{M}} \odot \boldsymbol{X}$. Convert your estimated piano spectrogram back to the time domain. Submit the .wav file of your recovered piano source.

5. Listen to the recovered source. Is it too different from the original? One way to objectively measure the quality of the recovered signal is to compare it to the original by using a metric called Signal-to-Noise Ratio (SNR):

$$SNR = 10 \log_{10} \left(\frac{\sum_t \{s(t)\}^2}{\sum_t \{s(t) - \hat{s}(t)\}^2} \right), \tag{12}$$

   where $s(t)$ is the t-th sample of the original source and $\hat{s}(t)$ is that of the recovered one. Evaluate the SNR between `piano.wav` and your reconstruction for it.

6. Yet another masking scheme is something called Ideal Binary Masks (IBM). This time, we use a binary (0 or 1) masking matrix $\boldsymbol{B}$, which is definded by

$$B_{ft} = \begin{cases} 1 & \text{if } S_{ft} > N_{ft} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

7. Create your IBM from the sources, and apply it to your mixture spectrogram, $\boldsymbol{S} \approx \boldsymbol{B} \odot \boldsymbol{X}$. Do the inverse STFT. How does it sound? What's its SNR value?

Problem 3: Single-channel Source Separation [2 points]

1. `trs.wav` is a speech signal of a speaker. Convert this signal into a spectrogram and take its magnitudes. Let's call this magnitude spectrogram $\boldsymbol{S}$. Learn an NMF model out of this such as $\boldsymbol{S} \approx \boldsymbol{W_S H_S}$. You know, $\boldsymbol{W_S}$ is a set of basis vectors. If you discard the complex conjugates, your $\boldsymbol{S}$ is a 513×990 matrix. If you choose to learn this NMF model with 30 basis vectors, then $\boldsymbol{W_S} \in \mathbb{R}_+^{513 \times 30}$, where $\mathbb{R}_+$ is a set of nonnegative real numbers. You're going to use $\boldsymbol{W_S}$ for your separation.

2. Learn another NMF model from `trn.wav`, which is another training signal for your noise. From this get $\boldsymbol{W_N}$.

3. `x_nmf.wav` is a noisy speech signal made of the same speaker's different speech and the same type of noise you saw. By using our third NMF

model, we're going to denoise this one. Load this signal and convert it into a spectrogram $\boldsymbol{X} \in \mathbb{C}^{513\times 131}$. Let's call its magnitude spectrogram $\boldsymbol{Y} = |\boldsymbol{X}| \in \mathbb{R}_+^{513\times 131}$. Your third NMF will learn this approximation:

$$\boldsymbol{Y} \approx [\boldsymbol{W_S W_N}]\boldsymbol{H}. \tag{14}$$

What this means is that for this third NMF model, instead of learning new basis vectors, you reuse the ones you trained from the previous two models as your basis vectors for testing: $\boldsymbol{W} = [\boldsymbol{W_S W_N}]$. As you are very sure that the basis vectors for your test signal should be the same with the ones you trained from each of the sources, you initialize your $\boldsymbol{W}$ matrix with the trained ones and don't even update it during this third NMF. Instead, you learn a whole new $\boldsymbol{H} \in \mathbb{R}_+^{60\times 131}$ that tells you the activation of the basis vectors for a given time frame. Implementation is simple. Skip the update for $\boldsymbol{W}$. Update $\boldsymbol{H}$ by using $\boldsymbol{W} = [\boldsymbol{W_S W_N}]$ and $\boldsymbol{Y}$. Repeat.

4. You can think of $\boldsymbol{W_S H}_{(1:30,:)}$ as an estimation of $\boldsymbol{S}$. But, you need its corresponding phase information to convert it back to the time domain. It might not be perfect, but the phase matrix of the input mixture, $\angle \boldsymbol{X} = \frac{\boldsymbol{X}}{\boldsymbol{Y}}$, can be used to recover the complex valued spectrogram of the speech source. Then, you can get the time domain signal. Submit your signal.

5. Because $\boldsymbol{W_S H}_{(1:30,:)}$ can be seen as the speech source estimate, you can also create a magnitude masking matrix out of it:

$$\bar{\boldsymbol{M}} = \frac{\boldsymbol{W_S H}_{(1:30,:)}}{\boldsymbol{W_S H}_{(1:30,:)} + \boldsymbol{W_N H}_{(31:60,:)}} = \frac{\boldsymbol{W_S H}_{(1:30,:)}}{[\boldsymbol{W_S W_N}]\boldsymbol{H}}. \tag{15}$$

Use this masking matrix to recover your speech source. Submit the wav file. Compared to the result from 3.4, which one do you think will have a larger SNR?

Problem 4: Motor Imagery [2 points]

1. `eeg.mat` has the training and testing samples and their labels. Use them to replicate my classification experiments in Lecture 7 (not the entire lecture, but from S3 to S8 and S37). But, instead of naïve Bayes classification, do a kNN classification. Report your classification accuracies from various choices of the number of PCs and the number of neighbors (I think a table of accuracies will be great). You don't have to submit all the intermediate plots. What I need is the accuracies and your source code. Please note that there was a typo in L7 S5: to get the same data matrix your hop size must be 48 samples, not 16 samples.