Carnegie Mellon DSP Electrical & Computer ENGINEERING

Digital Signal Processing (18-491/18-691)
Fall Semester, 2025

The 18-691 Final Project: Spectral Subtraction

Issued: 11/25/25

Due: 12/5/25 at 0100 via Gradescope, although 2-day late submissions will be accepted without penalty

Background: At the beginning of the semester we noted that students enrolled in 18-691 will be required to complete one additional assignment in order to receive graduate credit for the course. This is necessary in order to comply with ECE Department rules that state that courses in multiple tiers with parallel numbers must have at least somewhat different content to justify the parallel numbering.

The final assignment for the 18-691 students will be an implementation of the original spectral subtraction algorithm of Boll (1979),[1] which is available online on the course website. The project is intended to be easy and fun.

The final project will be given a grade equal to one homework assignment. Nevertheless, the grade for the final project will not be dropped from the final grade calculation, no matter how low it is. Each 18-691 student will work on his or her own on this project. The deadline for the final assignment will be December 7.

As usual, please submit your written responses and published code (as a single PDF) to the `Problem 11` assignment on Gradescope, and all code and de-noised audio files to the `Problem 11 - Matlab` assignment, also on Gradescope.

The Boll spectral subtraction algorithm

The Boll spectral subtraction algorithm is the foundational algorithm for a huge quantity of work in speech enhancement. It is extremely simple, but it has been improved on by dozens if not hundreds of subsequent research papers. Some of the latter work is reviewed superficially in a subsection of the book *Spoken Language Processing* by Hung, Acero, and Hon[2] (all of whom are former CMU students or faculty); this subsection on spectral subtraction is also provided online.

[1] Boll, S., "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **27**(2) (1979).

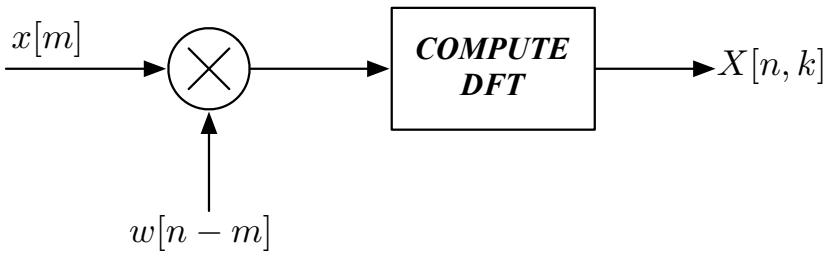
[2] Huang, X. D., Acero, A., and Hon, H.-W., *Spoken Language Processing*, Prentice-Hall PRT, 2001.

In essence, the goal of spectral subtraction is to remove (or at least attenuate) background noise in speech. This is accomplished by first estimating the power spectrum of the background noise at a time when speech is believed not to be present. The speech plus noise signal is processed using short-time Fourier analysis in the fashion described in Lecture 25, and discussed in the reference chapter by Nawab and Quatieri in the Lim and Oppenheim book, as well as the chapter in the ADSP class notes, both available on the Web. Briefly, the input signal is multiplied by a sequence of Hamming windows that are overlapped by 50 percent, computing the DFT of the finite-duration product to obtain the corresponding Fourier coefficients. The magnitude of the noise estimate is subtracted from the magnitude of the DFT coefficients in each frame. The difference of the magnitudes is combined with the original phase in each frame to produce a new set of DFT coefficients. The output time function is reconstructed by computing the inverse DFT of each frame, placing the resulting time function in the same location along the time axis as the original segment, overlapping the output functions by 50 percent as the input was overlapped.

Short-time Fourier analysis and the spectrogram

Introduction: Short-time Fourier analysis (STFA) was discussed in class in Lecture 25, and the STFA techniques used for this assignment were introduced in that lecture. The two recommended references (in addition to the class notes and the Powerpoint presentation from Lecture 25) are the notes on short-time Fourier transforms (STFTs) from 18-792 (ADSP) and the chapter on STFTs by Nawab and Quatieri in the edited text by Lim and Oppenheim that is used in ADSP. Both of these references are available online on the Lectures page of the 18-491/691 course website.

As described in the ADSP class notes, STFA is used to enable us to analyze the frequency components of a signal as it evolves over time. In Secs. 3.2 and 3.3 of the ADSP class notes we describe three ways of implementing the STFT: the Fourier transform implementation, the lowpass filter implementation, and the bandpass filter implementation. The STFT $X[n,k]$ of a time function $x[n]$ can be thought of as a matrix of complex coefficients, indicating the instantaneous power of the signal in discrete time frames (indexed by the horizontal axis of the matrix) and frequency bins (indexed by the vertical axis of the matrix). In this assignment, we will implement only the Fourier transform implementation.



Problem C11.1: (Spectrographic display)

(You will work this problem by completing the main file `main_11_1.m` that we provide, or for Python users, the function `main_11_1.py` in the template that is provided.)

Copyright 2025, Richard M. Stern and the DSP staff

The **spectrogram** is the major tool used to visualize signals as a simultaneous function of time and frequency. In essence it is a display of the log magnitude of the STFT of a signal, plotted as a function of time and frequency. In this problem you will complete the MATLAB function we provide called `spectrogram_ham_691_dft.m` that implements a spectrographic representation of a speech waveform using short-time Fourier analysis, displaying them using MATLAB. (You also complete your work in Python if you wish.) You can compare your result to that produced by the MATLAB routines `specgram` or `spectrogram`. Initially you will realize the STFT using the Fourier transform implementation as depicted above.

The speech wave that you will use as input is the utterance "Welcome to DSP-I" that we have used in class from time to time, but downsampled to a sampling rate of 8 kHz. Obtain the waveform `welcome8k.wav` in the `h691_extras` folder that os on the Homework page on the Web.

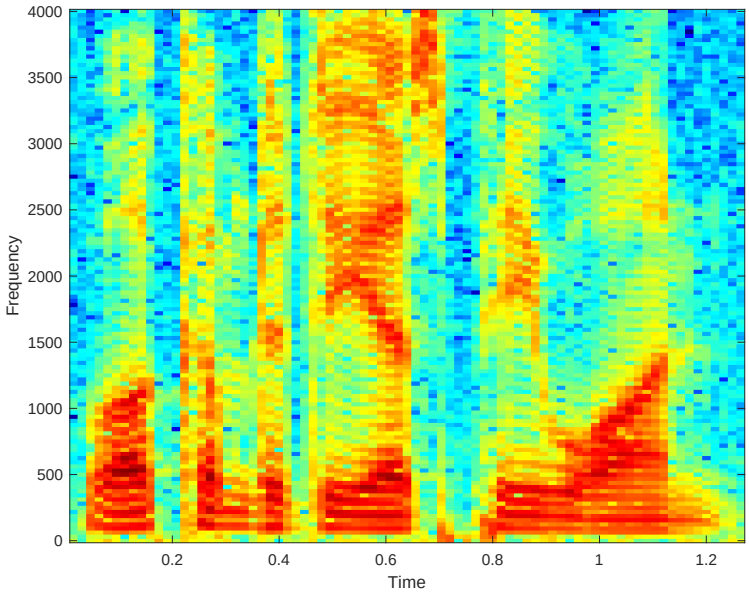
Note: Turn in the MATLAB scripts that you used to obtain your answers in all of the following problems.

(a) Given that you are using Hamming windows and the Fourier transform implementation of the STFT, what is the most efficient choice of the time between frames given the window length that would enable you to completely reconstruct the original time function using the overlap-add (OLA) method)? Explain your reasoning.

(b) What is the minimum DFT size that you would need to use for a window of length N?

(c) What is the window length N that you would need to obtain an effective bandwidth of the analysis window of W Hz? (Recall that the analysis window $w[n]$ can also be considered as a filter, as in the lowpass and bandpass implementations of the STFT.) Express your results in terms of the sampling frequency F_s, and the desired filter bandwidth W.

(d) Using your results from parts (a) through (c), calculate the complex STFT coefficients $X[n, k]$ for the "Welcome to DSP-I" utterance separately using effective filter bandwidths (from the windows) of 50 and 350 Hz. State how you achieve these bandwidths using Hamming windows.



Copyright 2025, Richard M. Stern and the DSP staff

(e) Using Hamming windows for your analysis function, create two vectors, `t` and `f` representing the nominal frame times and center analysis frequencies, respectively. Display your output using the MATLAB commands

```
IMAGESC(t,f,10*log10(abs(X)));
axis xy;
COLORMAP(JET);
```

Your spectrograms should look something like the figure above, although this example was created using a different window length.

What you will turn in: Submit electronically on Gradescope:

- Your written responses to parts (a) through (c), and published code (containing spectrograms) as a part of your report.
- Your code files used to compute and display the spectrograms.

Implementing spectral subtraction

Problem C11.2: In this problem we will develop a basic bare-bones implementation of the Boll spectral subtraction algorithm by completing the empty portions of the shell script `SpecSub691.m`.[3]

What is provided: You will find on the course website the code shell `SpecSub691.m` along with nine audio input files. We are providing an input speech signal with a familiar voice from the CMU Arctic database for text-to-speech synthesis[4] that is pre-combined with three types of noise samples: artificially generated white noise, live recording of speech babble, and a sample from the classic rock catalog ("Live With Me" by The Rolling Stones). Three signal-to-noise ratios are employed: 5, 0, and −5 dB. The input file names are obvious, and are of the form `speech_noise5.wav`, `speech_babble0.wav`, and `speech_rock-5.wav`, etc. You will need to place the nine `*.wav files` in the same directory as the calling program. Note that these files are all sampled at 16 kHz.

In working this problem, by all means re-use the code you developed for Problem C11.1 to the extent possible, in either its original form or with some modifications.

Here we summarize the major steps in performing this analysis:

- While normally the task of identifying whether speech is present or absent can be quite difficult and is a research discipline of its own (voice activity detection or VAD, also called more recently speech activity detection or SAD), we are making it easy for this assignment by guaranteeing that the first two seconds of the input utterances contain noise only. Hence you should just use the first two seconds of each excerpt to estimate the magnitude of the noise.

[3] If you are doing your work in Python, please use the identically-named Python files.

[4] `http://www.festvox.org/cmu_arctic/`

Copyright 2025, Richard M. Stern and the DSP staff

- In reality, estimating the magnitude of the noise (technically referred to as power spectral density [PSD] function estimation) had also been an active research problem for multiple decades. You should estimate the noise by breaking up the first 2 seconds of the input file into segments of length 512 samples, applying a 512-point Hamming window to each segment, computing the 512-point DFT for each windowed frame, and finally averaging the magnitude of the DFT over all frames. You should end up with a single vector of length N=512, with real values that are greater than or equal to zero. For these calculations you can either abut the successive segments of length 512 or over lap them by 50 percent.

- Now create frames of windowed input audio, extracted using Hamming windows of length 512, overlapping by 50 percent of the window length (to enable distortionless reconstruction using the overlap-add algorithm (OLA) for the STFTs). For each frame, calculate and store the 512-point DFT of the windowed time function. You will use many of the same methods that you used in obtaining your solution to Problem 11.1, although some of the parameter values are different.

- Subtract the (fixed) magnitude of the noise estimate obtained above from the magnitude of the DFT coefficients representing each of the input signal frames. Half-wave rectify the difference in magnitude to ensure that the result is greater than or equal to zero. For each frame, combine the subtracted magnitude of the frame with the phase of the original input for each DFT coefficient.

- Now reconstruct the output signal using the overlap-add (OLA) method as described in our discussion of short time Fourier analysis (STFA), Lecture 25. In other words, compute the IDFT of each frame and add the resulting real sequence at the same time indices that had been used to extract the frame initially.

- Complete your work by completing the shell function that is provided, `SpecSub691.m`. This MATLAB file includes code for reading and writing the files with appropriate filenames, as well as (optionally) for playing the audio files before and after your processing.

What to turn on in writing:

Please comment on the following:

- Comment on some of the critical design choices you made and how/why you made them.

- To what extent is your implementation is effective in attenuating the three types of noise. For which noise was the compensation most effective, and for which noise type was it least effective? Why do you think that this was the case?

- To what extent is the speech portion of the output clean or distorted? How much does this change with SNR? How much does this change with noise type?

- Comment on how you think that the algorithm could be improved.

What you will submit::

- Your response to the questions above.

Copyright 2025, Richard M. Stern and the DSP staff

- Your completed code files, with published code being attached to the report submission.
- Audio files in `.zip` from your output at 0-dB SNR. Specifically, we will listen to and evaluate the files `speech_{noise,babble,rock}0_out.wav`.
- Spectrograms of the pre- and post- denoising audio for each of these three files (included in your report submission)

Some additional coments: As has been noted, a great deal of work has been expended on improving the basic spectral subtraction algorithm. Typically these efforts address one of the following issues:

- With the actual noise (and signal) changing over time, it is easily possible for the subtraction to result in a negative magnitude which is impossible in theory and problematical in practice. In our basic implementation we adopt the simple fix of setting the floor of the subtraction to be zero. Many more sophisticated schemes have been proposed that typically involve under-subtracting the estimated noise magnitude from the magnitude of the spectral of the incoming signal.
- In reality the masking noise is not stationary. Many algorithms have been proposed to update the estimate of the power spectrum of the background noise dynamically, in an adaptive fashion. These approaches typically depend on being able to detect the absence of the target speech, though, which is a difficult problem.
- In recent years especially, greater attention has been focused on the consequences of the mismatch between the magnitude of the frequency response in the individual frames and the corresponding phases. (This happens, of course, because we are combining the phase of the original speech with the magnitude of the signal after subtracting the magnitude of the noise estimate.) Some approaches to this have been successful, but they all involve iterative nonlinear estimation techniques.

Copyright 2025, Richard M. Stern and the DSP staff