Short-Time Fourier Transform and Its Inverse

Ivan W. Selesnick

April 14, 2009

1 Introduction

The short-time Fourier transform (STFT) of a signal consists of the Fourier transform of overlapping windowed blocks of the signal. In this note, we assume the overlapping is by 50% and we derive the perfect reconstruction condition for the window function, denoted $w(n)$.

The window $w(n)$ is assumed to be supported (non-zero) for $n = 0, \ldots, N-1$. For example, Figure 1 shows a window of length $N = 10$. In this example, the window is a symmetric half-cycle sine window. The N-point half-cycle sine window is given by

$$w(n) = \sin\left(\frac{\pi}{N}(n + 0.5)\right), \qquad n = 0, \ldots, N-1. \tag{1}$$

The figure shows several shifted windows. The shifted windows are shifted by half the length of the window. (In practice, the window is much longer than 10 samples. A short window is used here for illustration.)

The m-th windowed block of the signal $x(n)$ is given by $x(n) \cdot w(n - m \cdot N/2)$. For example, Figure 2 shows several windowed blocks. The m-th windowed block is denoted as $s(m, n)$:

$$s(m, n) := x(n) \cdot w(n - m \cdot N/2) \tag{2}$$

For example, Figure 2 shows the m-th windowed block for $m = 0, \ldots, 4$.

The short-time Fourier transform is obtained by taking the DTFT of each windowed block:

$$S(m, \omega) := \mathrm{DTFT}\{x(n) \cdot w(n - m \cdot N/2)\} \tag{3}$$

The short-time Fourier transform of a discrete-time signal $x(n)$ is denoted by

$$S(m, \omega) = \mathrm{STFT}\{x(n)\}.$$

In practice, the DTFT is computed using the DFT or a zero-padded DFT.

2 Inverse STFT

The inverse STFT begins with the inverse DTFT of $S(m, \omega)$ to recover $s(m, n)$.

$$s(m, n) = \mathrm{DTFT}^{-1}\{S(m, \omega)\}$$

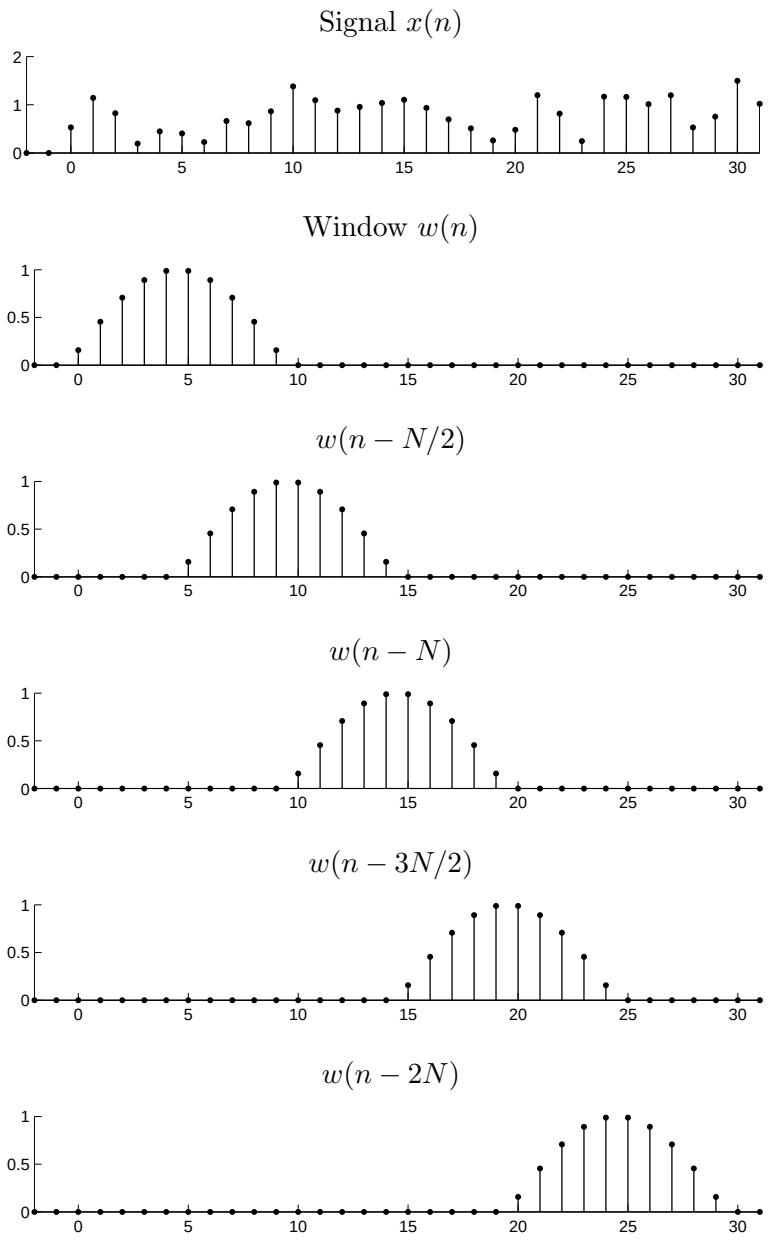


Figure 1: A window $w(n)$ of length $N = 10$ and several shifted windows $w(n - m \cdot N/2)$.

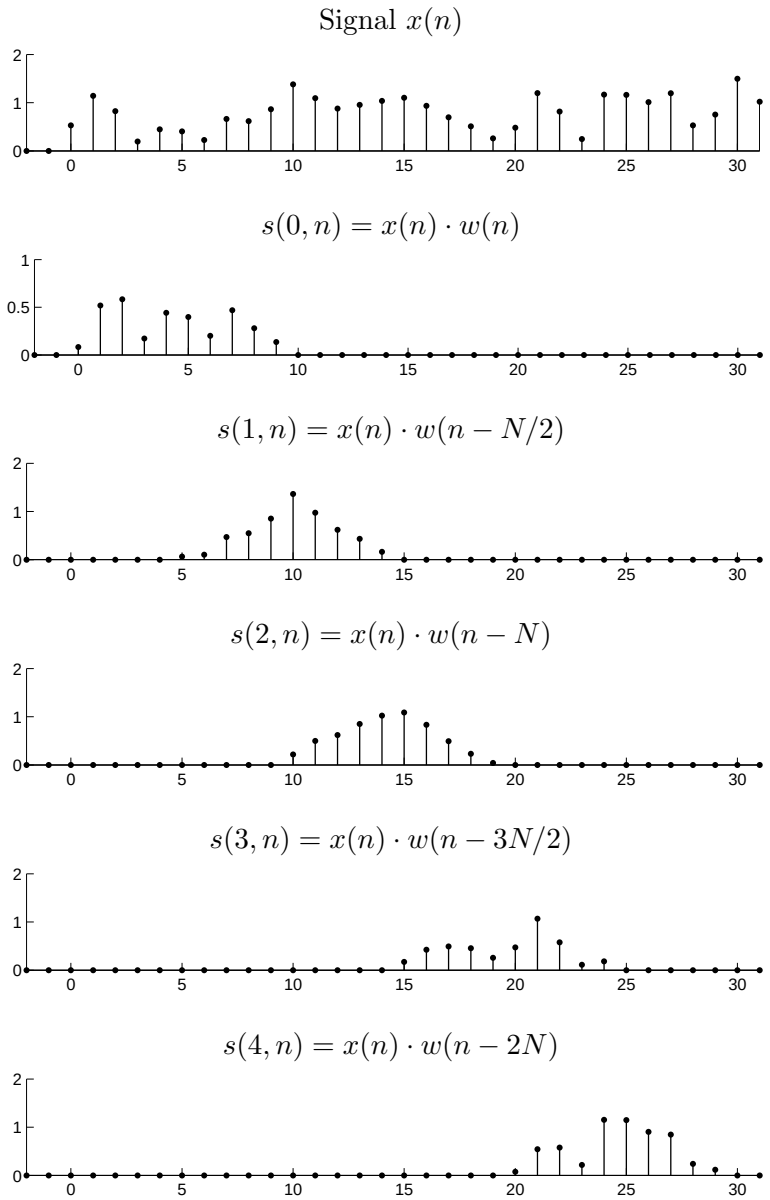


Figure 2: Windowed blocks $x(n) \cdot w(n - m \cdot N/2)$. The shifted windows are shown in Figure 1.

Now, from $s(m,n)$ we wish to recover $x(n)$ by multiplying each $s(m,n)$ by the shifted window $w(n - m \cdot N/2)$ and adding the results. We will use the same window used in the forward STFT. Multiplying the m-th windowed block by the shifted window gives:

$$s(m,n) \cdot w(n - m \cdot N/2)$$

which are illustrated in Figure 3. Figure 3 shows the windowed $s(m,n)$ for $m = 0, \ldots, 4$.

The next step of the inverse STFT adds these overlapping blocks to obtain the final signal $y(n)$:

$$y(n) = \sum_m s(m,n) \cdot w(n - m \cdot N/2) \tag{4}$$

We have called this the 'inverse' STFT, however, it is only an inverse if $y(n) = x(n)$, which in turn depends on the window $w(n)$. If the window is not chosen correctly, then the reconstructed signal $y(n)$ will not be equal to the original signal $x(n)$.

3 Perfect Reconstruction Condition

How should the window $w(n)$ be chosen so as to ensure that the 'inverse' STFT really is an inverse? Combining (2) and (4) gives

$$y(n) = \sum_m x(n) \cdot w^2(n - m \cdot N/2). \tag{5}$$

If we define the squared window function

$$p(n) := w^2(n)$$

then (5) can be written as

$$y(n) = x(n) \sum_m p(n - m \cdot N/2).$$

For perfect reconstruction ($y(n) = x(n)$) it is necessary that

$$\sum_m p(n - m\, N/2) = 1. \tag{6}$$

The half-cycle sine window satisfies this condition as illustrated in Figure 4. Note that the first $N/2$ samples and the last $N/2$ samples are exceptions. The beginning and end of the signal can be inverted using a different procedure or, if the signal is long then these relatively few points at the beginning and end may not matter.

Note that the left-hand-side of (6) is periodic with period $N/2$. Therefore it is necessary to check the condition only for $n = 0, \ldots, N/2 - 1$ (or for any other $N/2$ range of n). Moreover, over this interval, only two terms contribute; so the perfect reconstruction simplifies to:

$$p(n) + p(n + N/2) = 1, \qquad n = 0, \ldots, \frac{N}{2} - 1.$$

Therefore, the perfect reconstruction condition is

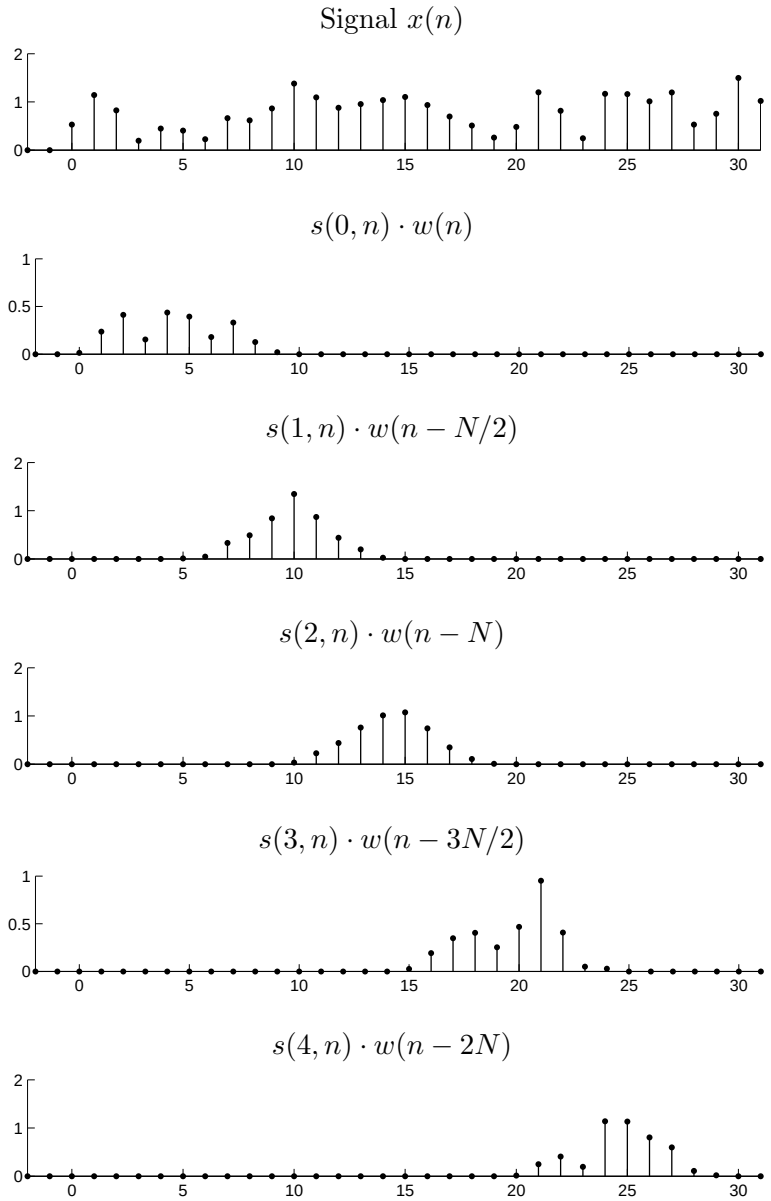


Figure 3: For the inverse STFT the overlapping signals $s(m,n) \cdot w(n - m \cdot N/2)$ are added.

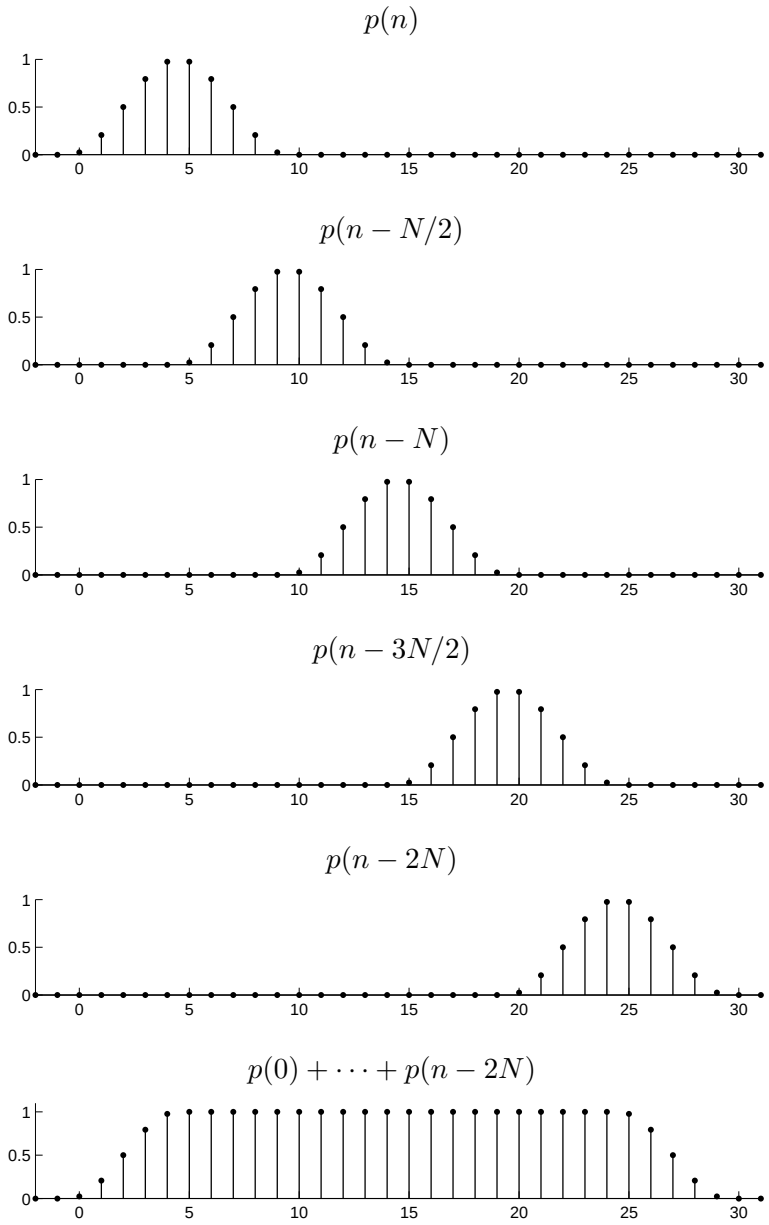


Figure 4: Illustration of the perfect reconstruction condition (6) for the 10-point half-cycle sine window.

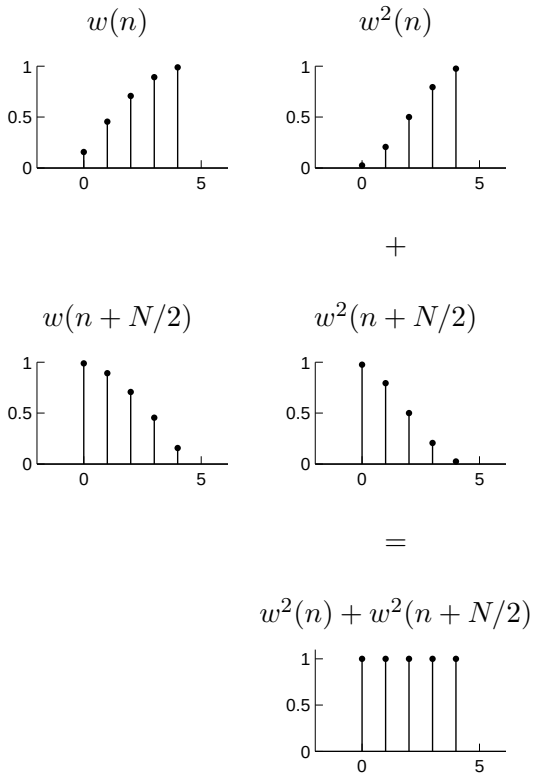


Figure 5: Illustration of the perfect reconstruction condition (7) for the 10-point half-cycle sine window.

$$w^2(n) + w^2(n+N/2) = 1, \qquad n = 0, \ldots, \frac{N}{2} - 1. \tag{7}$$

The half-cycle sine window (1) satisfies (7) as illustrated in Figure 5. Basically it satisfies (7) because of the trigonometric identity $\cos^2(\theta) + \sin^2(\theta) = 1$. Many other windows can also be designed that satisfy (7). Perhaps the simplest N-point window satisfying (7) is the rectangular window,

$$w(n) = \frac{1}{\sqrt{2}}, \qquad n = 0, \ldots, N-1,$$

however, this window is not a good window because it is not tapered (smooth) at its ends. It can therefore cause discontinuities at block boundaries when the STFT is used for noise reduction, signal enhancement, or other applications. This discontinuity is sometimes audible as a low noise.

4 Speech Noise Reduction

The STFT can be used to reduce noise in a speech signal (or other highly oscillatory signal). A simple method consists of three steps:

1. Compute the STFT of the noisy signal.

   $$S(m,\omega) = \text{STFT}\{x(n)\}$$

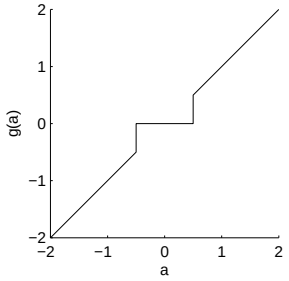
2. Threshold the STFT.

   $$S_2(m,\omega) = g(S(m,\omega))$$

   where $g(a)$ is the thresholding function:

   $$g(a) = \begin{cases} 0, & |a| \le T \\ a, & |a| > T \end{cases} \tag{8}$$

   All values less than the threshold T in absolute value get set to zero.



3. Compute the inverse STFT.

   $$y(n) = \text{STFT}^{-1}\{S_2(m,\omega)\}.$$

This is illustrated by the block diagram:

Example: Speech denoising by STFT-thresholding is illustrated in Figures 6 and 7. Figure 6 shows a noisy speech signal and its spectrogram computed using 50% overlapping. The noise is visible in both the signal and in the spectrogram. Setting the small spectrogram values to zero (thresholding) results in the spectrogram shown in Figure 7. Clearly, much of the noise in the spectrogram is eliminated. After this thresholding, the inverse STFT is computed to obtain the denoised signal also shown in Figure 7. Figure 8 shows a closer view of the noisy and denoised speech signal for closer comparison.

STFT-thresholding is a non-linear noise reduction technique because the thresholding operation is non-linear. It can yield noise reduction results that are not possible with any linear time-invariant (LTI) filter.

This method is sensitive to the choice of threshold T. If the threshold T is too large, then the signal will be highly distorted. If T is too small, then the result will still be noisy.

This basic algorithm described here can be improved in several different ways. Instead of using a fixed threshold T, one can vary the threshold as a function of time and/or frequency. In addition, one can use different nonlinearities instead of the threshold function (8). Also, one can develop methods to automatically estimate an appropriate threshold value which is needed in practice.

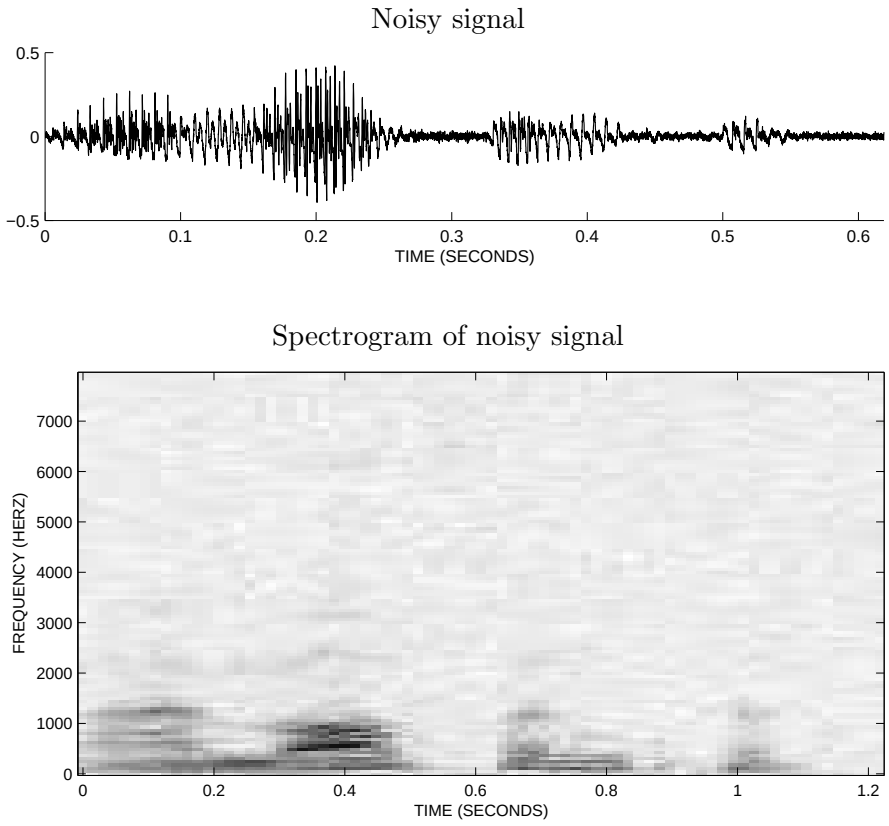


Figure 6: Noisy signal and its STFT (spectrogram). The noise visible in the spectrogram is spread out across time and frequency.

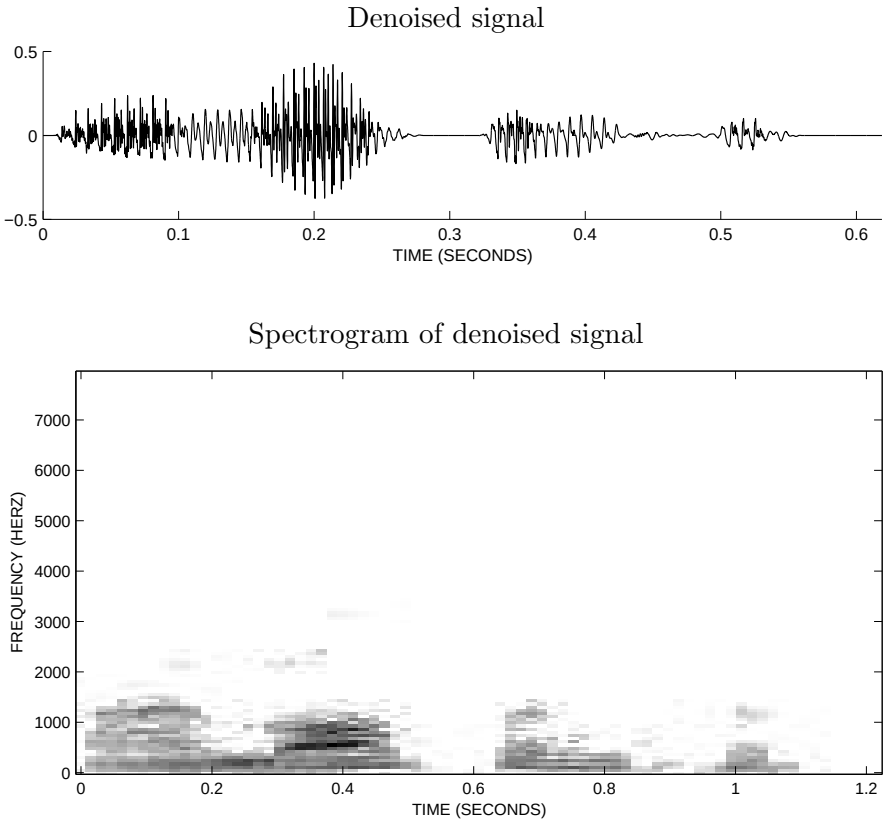


Figure 7: Denoised signal and its STFT (spectrogram). Thresholding the spectrogram eliminates most of the noise.

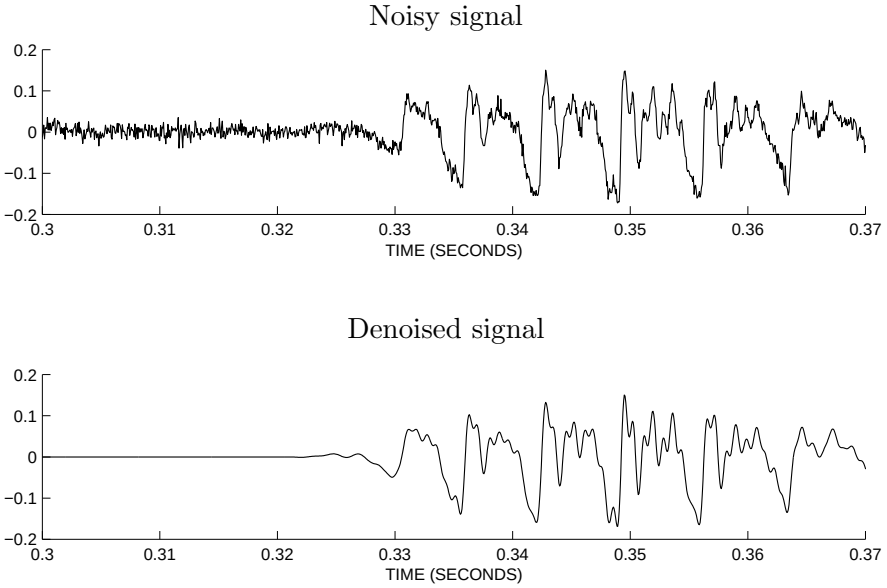


Figure 8: Noisy and denoised speech signal (magnified view).

5 Exercises

1. Write a MATLAB program to implement the STFT with 50% overlapping and a second program to implement its inverse. Verify numerically that the inverse program reconstructs a test signal.

2. Find another window satisfying the perfect reconstruction condition for 50% overlapping, use it with your STFT program, and verify that it reconstructs a test signal.

3. Find the perfect reconstruction (PR) condition for an STFT with 2/3 overlapping. Find a window that satisfies your PR condition. Write a MATLAB program to implement the STFT and its inverse with 2/3 overlapping and verify the reconstruction of a test signal.

4. Use the STFT and thresholding to reduce the noise level in a noisy speech signal. Try to use your own speech signal recorded using a computer (most audio recordings with consumer equipment contain some hiss; try to reduce the sound of the hiss).

   In case your STFT program does not work, you may use the provided function `my_stft` provided as a `.p` file. A `.p` file is a type of MATLAB file that can be executed in MATLAB but whose contents are not viewable (see the `pcode` function).