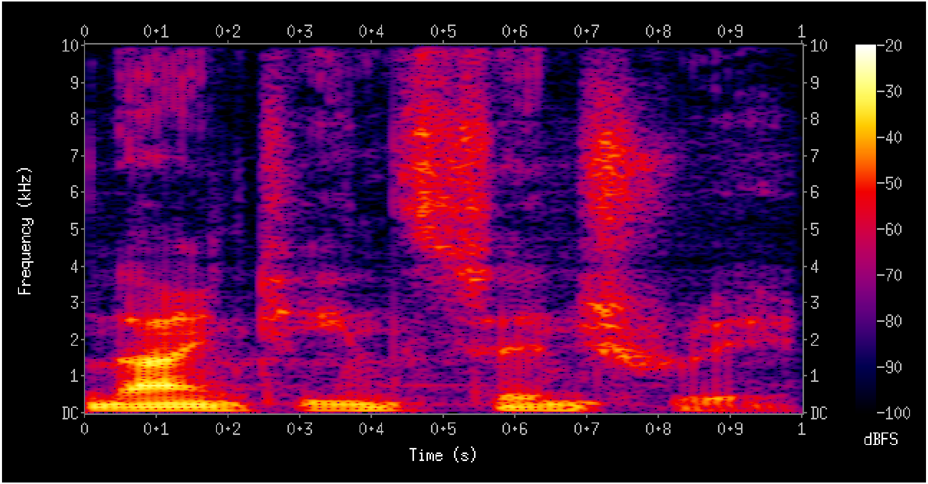
Lecture 5: Short-Time Fourier Transform and Filterbanks

Mark Hasegawa-Johnson

ECE 417: Multimedia Signal Processing, Fall 2023

Outline

Spectrogram = $20 \log_{10} |\text{Short Time Fourier Transform}|$



Short Time Fourier Transform

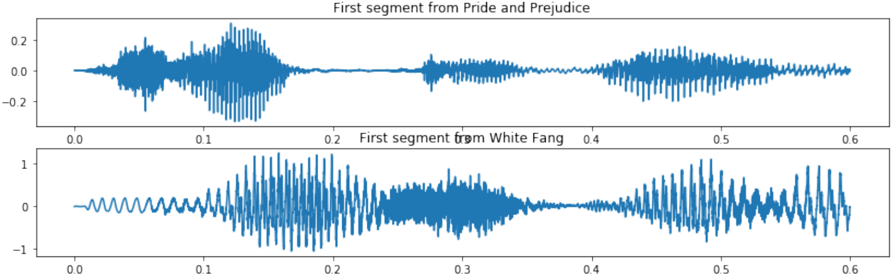
The short-time Fourier Transform (STFT) is the Fourier transform of a short part of the signal. We write either $X_m(\omega)$ of $X_m[k]$ to mean:

- The DFT of the short part of the signal that starts at sample m,
- windowed by a window of length $L \leq N$ samples,
- evaluated at frequency $\omega = \frac{2\pi k}{N}$.

The next several slides will go through this procedure in detail, then I'll summarize.

Step #1: Chop out part of the signal

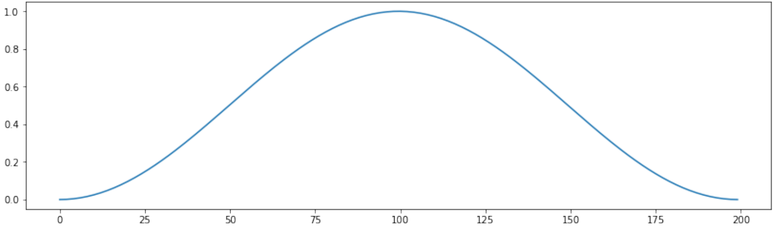
First, we just chop out the part of the signal starting at sample m. Here are examples from Librivox readings of *White Fang* and *Pride and Prejudice*:



Step #2: Window the signal

Second, we window the signal. A window with good spectral properties is the Hamming window. The length of the window might be L, which might be less than the FFT length N:

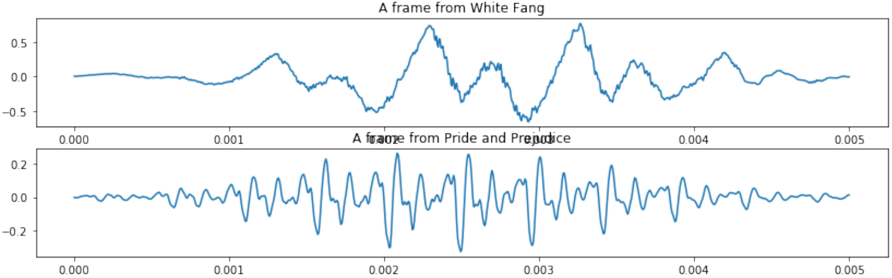
$$w[n] = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi n}{L-1}\right) & 0 \le n \le L-1 \\ 0 & \text{otherwise} \end{cases}$$



Step #2: Window the signal

Here is the windowed signals, which is nonzero for $0 \leq n - m \leq (L-1)$:

$$x[n, m] = w[n-m]x[n]$$

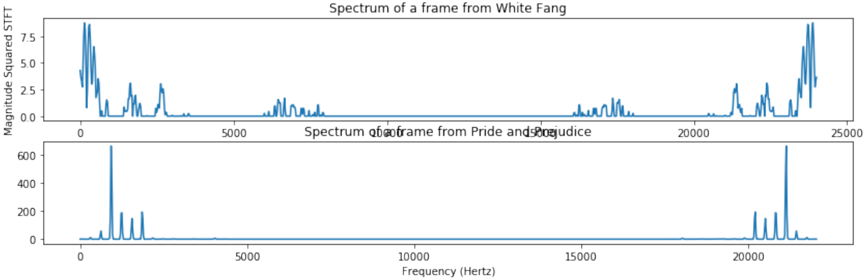


Step #3: Fourier Transform

Finally, we take the DTFT of the windowed signal. The result is the STFT, $X_m(\omega)$:

$$X_m(\omega) = \sum_{n=m}^{m+(L-1)} w[n-m]x[n]e^{-j\omega(n-m)}$$

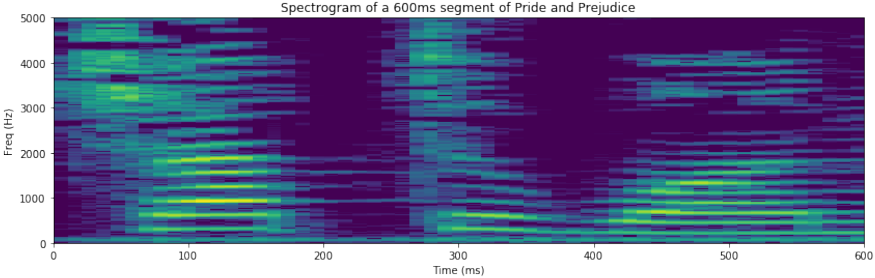
Here it is, plotted as a function of k:



Spectrogram $= 20\log_{10}|\text{Short Time Fourier Transform}|$

$$20\log_{10}|X_m(\omega)| = 20\log_{10}\left|\sum_n w[n-m]x[n]e^{-j\omega(n-m)}\right|$$

Here it is, plotted as an image, with k =row index, m =column index.



Putting it all together: STFT

The STFT, then, is defined as

$$X_m(\omega) = \sum_n w[n-m]x[n]e^{-j\omega(n-m)}, \quad \omega = \frac{2\pi k}{N}$$

which we can also write as

$$X_m[k] = \text{DFT}\left\{w[n]x[n+m]\right\}$$

Outline

STFT as a bank of analysis filters

The STFT is defined as:

$$X_m[k] = \sum_{n=m}^{m+(L-1)} w[n-m]x[n]e^{-j\omega_k(n-m)}$$

which we can also write as

$$X_m[k] = x[m] * h_k[m]$$

where

$$h_k[m] = w[-m]e^{j\omega_k m}$$

The frequency response of this filter is just the DTFT of $w[-m]$, which is $W(-\omega)$, shifted up to ω_k:

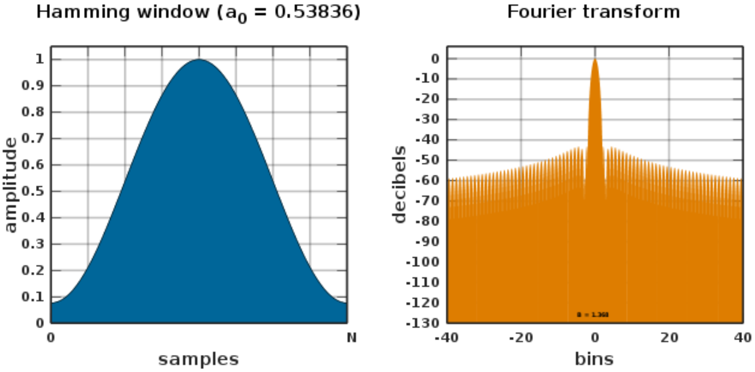
$$H_k(\omega) = W\left(\omega_k - \omega\right)$$

Hamming window spectrum

The frequency response of this filter is just the DTFT of $w[-m]$, which is $W(-\omega)$, shifted up to ω_k:

$$H_k(\omega) = W\left(\omega_k - \omega\right)$$

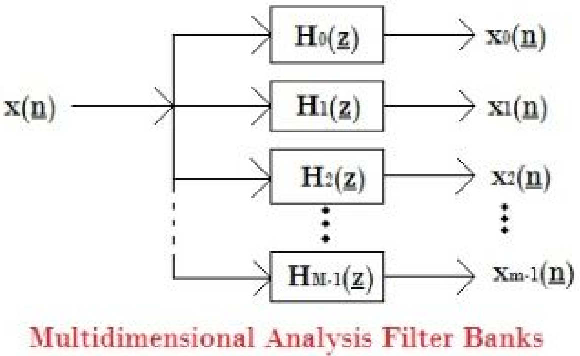
For a Hamming window, $w[n]$ is on the left, $W(\omega)$ is on the right:



By Olli Niemitalo, public domain image, https://en.wikipedia.org/wiki/Window_function

STFT as a bank of analysis filters

So the STFT is just like filtering $x[n]$ through a bank of analysis filters, in which the k^{th} filter is a bandpass filter centered at ω_k:



By Ventetpluie, GFDL,

https://en.wikipedia.org/wiki/File:Multidimensional_Analysis_Filter_Banks.jpg

Short-Time Fourier Transform

- **STFT as a Transform:**

$$X_m[k] = \text{DFT}\left\{ w[n]x[n+m] \right\}$$

- **STFT as a Filterbank:**

$$X_m[k] = x[m] * h_k[m], \qquad h_k[m] = w[-m]e^{j\omega_k m}$$

Outline

1. Short-Time Fourier Transform
2. STFT as a Linear-Frequency Filterbank
3. Inverse STFT
4. Implementing Nonlinear-Frequency Filterbanks Using the STFT
5. Summary

Short-Time Fourier Transform

- **STFT as a Transform:**

$$X_m[k] = \text{DFT}\left\{w[n]x[n+m]\right\}$$

- **STFT as a Filterbank:**

$$X_m[k] = x[m] * h_k[m], \qquad h_k[m] = w[-m]e^{j\omega_k m}$$

The inverse STFT

STFT as a transform is defined as:

$$X_m[k] = \sum_{n=m}^{m+(N-1)} w[n-m]x[n]e^{-j2\pi k(n-m)/N}$$

Obviously, we can inverse transform as:

$$x[n] = \frac{1}{Nw[n-m]} \sum_{k=0}^{N-1} X_m[k]e^{j2\pi k(n-m)/N}$$

The inverse STFT

We get a better estimate of $x[n]$ if we average over all of the windows for which $w[n-m] \neq 0$. This is often called the overlap-add method, because we overlap the inverse-transformed windows, and add them together:

$$x[n] = \frac{\sum_m \frac{1}{N} \sum_{k=0}^{N-1} X_m[k] e^{j\omega_k (n-m)}}{\sum_m w[n-m]}$$

Often, the denominator is a constant, independent of n. That happens automatically if there has been no downsampling; it is

$$W(0) = \sum_{m=0}^{N-1} w[m]$$

STFT: Forward and Inverse

- **Short Time Fourier Transform (STFT)**:

$$X_m[k] = \sum_n w[n-m]x[n]e^{-j\omega_k(n-m)}, \quad \omega_k = \frac{2\pi k}{N}$$

- **Inverse Short Time Fourier Transform (ISTFT, OLA method)**:

$$x[n] = \frac{1}{NW(0)} \sum_m \sum_{k=0}^{N-1} X_m[k]e^{j\omega_k(n-m)}$$

ISTFT as a bank of synthesis filters

Inverse Short Time Fourier Transform (ISTFT):

$$x[n] = \frac{1}{NW(0)} \sum_m \sum_{k=0}^{N-1} X_m[k] e^{j\omega_k (n-m)}$$

The ISTFT is the sum of filters:

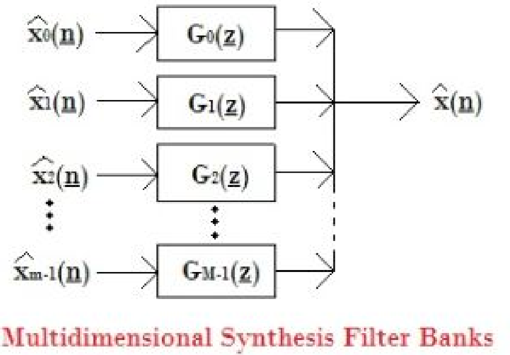
$$\begin{aligned} x[n] &= \frac{1}{W(0)} \sum_m \sum_{k=0}^{N-1} X_m[k] e^{j\omega_k (n-m)} \\ &= \sum_{k=0}^{N-1} \left(X_m[k] * g_k[m] \right) \end{aligned}$$

where

$$g_k[m] = \begin{cases} \frac{1}{W(0)} e^{j\omega_k m} & 0 \le m \le N-1 \\ 0 & \text{otherwise} \end{cases}$$

ISTFT as a bank of synthesis filters

So the ISTFT is just like filtering $X_m[k]$ through a bank of synthesis filters, in which the k^{th} filter is a bandpass filter centered at ω_k:

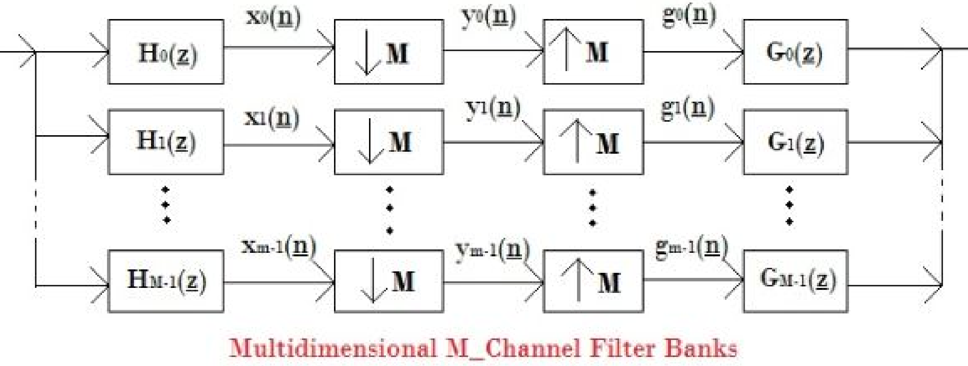


By Ventetpluie, GFDL,

https://en.wikipedia.org/wiki/File:Multidimensional_Synthesis_Filter_Banks.jpg

The whole process: STFT and ISTFT as a filterbanks

We can compute the STFT, downsample, do stuff to it, upsample, and then resynthesize the resulting waveform:



By Ventetpluie, GFDL,

https://en.wikipedia.org/wiki/File:Multidimensional_M_Channel_Filter_Banks.jpg

Outline

Short-Time Fourier Transform

- **STFT as a Transform:**

$$X_m[k] = \text{DFT}\left\{w[n]x[n+m]\right\}$$

- **STFT as a Filterbank:**

$$X_m[k] = x[m] * h_k[m], \qquad h_k[m] = w[-m]e^{j\omega_k m}$$

Relative Benefits of Transforms vs. Filters

- **STFT as a Transform:** Implement using Fast Fourier Transform.

$$X_m[k] = \text{DFT}\left\{w[n]x[n+m]\right\}$$
$$\textbf{Computational Complexity} = \mathcal{O}\left\{N\log_2(N)\right\}\ \text{per } m$$
$$\textbf{Example:} N = 1024$$
$$\textbf{Computational Complexity} = 10240 \text{ multiplies/sample}$$

- **STFT as a Filterbank:** Implement using convolution.

$$X_m[k] = x[m] * h_k[m]$$
$$\textbf{Computational Complexity} = \mathcal{O}\left\{N^2\right\}\ \text{per } m$$
$$\textbf{Example:} N = 1024$$
$$\textbf{Computational Complexity} = 1048576 \text{ multiplies/sample}$$

What about other filters?

- Obviously, FFT is much faster than the convolution approach.
- Can we use the FFT to speed up other types of filter computations, as well?
- For example, can we model the bandpass filtering operations of the human ear from the STFT?

What about other filters?

- We want to find $y[n] = f[n] * x[n]$, where $f[n]$ is a length-N impulse response.
- Complexity of the convolution in time domain is $\mathcal{O}\{N\}$ per output sample.
- We can't find $y[n]$ exactly, but we can find $\tilde{y}[n] = f[n] \circledast (w[n-m]x[n])$ from the STFT:

$$Y_m[k] = F[k]X_m[k]$$

- It makes sense to do this only if $F[k]$ has far fewer than N non-zero terms (narrowband filter).

Bandpass-Filtered Signal Power

In particular, suppose that $f[n]$ is a bandpass filter, and we'd like to know how much power gets through it.
So we'd like to know the power of the signal $\tilde{y}[n] = f[n] \circledast (w[n-m]x[n])$. We can get that as

$$\sum_{n=0}^{N-1} \tilde{y}[n]^2 = \frac{1}{N}\sum_{k=0}^{N-1} |Y_m[k]|^2$$
$$= \frac{1}{N}\sum_{k=0}^{N-1} |F[k]|^2 |X_m[k]|^2$$

Outline

Summary

- **STFT as a Transform:**

$$X_m(\omega) = \sum_n w[n-m]x[n]e^{-j\omega(n-m)}, \quad \omega_k = \frac{2\pi k}{N}$$

- **STFT as a Filterbank:**

$$X_m(\omega) = x[m] * h_k[m], \qquad h_\omega[m] = w[-m]e^{j\omega m}$$

- **Other filters using STFT:**

$$\text{DFT}\left\{f[n] \circledast (w[n-m]x[n])\right\} = H[k]X_m[k]$$

- **Bandpass-Filtered Signal Power**

$$\sum_{n=0}^{N-1} \tilde{y}[n]^2 = \frac{1}{N}\sum_{k=0}^{N-1} |F[k]|^2 |X_m[k]|^2$$