GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL of ELECTRICAL and COMPUTER ENGINEERING

ECE 2026 Spring 2015
Lab #12: Cochlear Implant Simulation : An Introduction

Date: 13–16 Apr. 2015

You should read the Pre-Lab section of the lab and do all the exercises in the Pre-Lab section before your assigned lab time.

Verification: The Warm-up section of each lab must be completed **during your assigned Lab time** and the steps marked *Instructor Verification* must also be signed off **during the lab time**. When you have completed a step that requires verification, simply raise your hand and demonstrate the step to the TA or instructor. After completing the warm-up section, turn in the verification sheet to your TA *before leaving the lab.*

Forgeries and plagiarism are a violation of the honor code and will be referred to the Dean of Students for disciplinary action. You are allowed to discuss lab exercises with other students, but you cannot give or receive any written material or electronic files. In addition, you are not allowed to use or copy material from old lab reports from previous semesters. Your submitted work must be your own original work.

1 Introduction

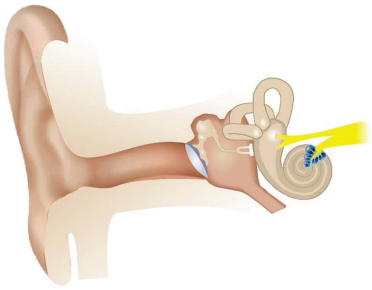
A Cochlear Implant (CI) serves to overcome deafness by directly stimulating auditory nerve fibers with electrical current thereby conveying auditory cues. The goal of this lab is to mimic the speech processing function of a cochlear implant. To better understand how a cochlear implant functions, it is useful to review the process of hearing. The ear is divided into three parts: outer, middle and inner as shown in Fig. 1a. During the process of normal hearing, the outer ear captures sound, which the middle ear converts into mechanical vibrations. These vibrations travel into the cochlea, a coiled fluid-filled tube, located in the inner ear. Dividing the fluid-filled tube is a membrane, the basilar membrane that exhibits a variable mechanical stiffness. As a result, it is more sensitive to high frequencies at the base, and more sensitive to low frequencies near the apex (see Fig. 1b). The fluid displacement reveals the frequency information of the acoustic signal. Attached to the membrane are small hair cells lined with stereocilia that bend when the membrane is displaced. When bent, these hair cells activate neurons that fire signals to the brain relaying acoustic information. Deafness is caused by the disruption of this hearing process. In the majority of the cases, it is the hair cells that are damaged, not the auditory nerves. A cochlear implant functions by directly stimulating the auditory nerves with patterns of electrical current determined by speech processing.[1]

Figure 2a depicts the four external parts of a cochlear implant. Understanding the signal processing inside the speech processor will be the focus of this lab. Figure 3 shows a block diagram of the major signal processing blocks.

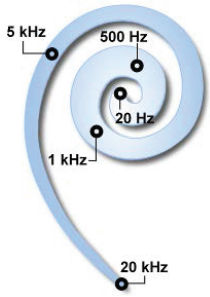
2 Pre-Lab

In this lab, the student will be able to hear what a patient with a Cochlear Implant (CI) hears. A sound file will undergo speech processing similar to that of a cochlear implant. The following sections describe the major signal processing blocks needed to decompose a speech (or audio) signal into frequency components that are consistent with the frequency-to-place mapping of the cochlea.

[1] P. C. Loizou, "Introduction to Cochlear Implants," *IEEE Engineering in Medicine and Biology Magazine,* Vol. 18, No. 1, pp. 32–42, 1999. URL (March, 2009) is: `http://www.utdallas.edu/~loizou/cimplants/tutorial/`

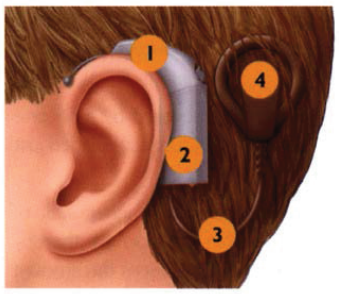


(a) Peripheral auditory system.

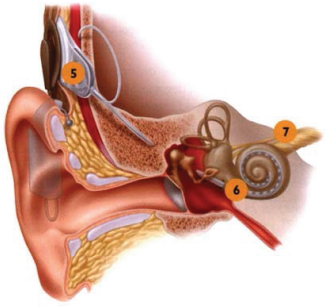


(b) Frequency-to-place mapping of the cochlea.

Figure 1: (a) The ear is divided into an outer, middle, and inner ear. The cochlea is located in the inner ear. (b) Different acoustic frequencies activate different parts of the basilar membrane, and therefore the cochlea. The acoustic frequency as a function of place is indicated in Hz.



(a) External parts of a cochlear implant.



(b) Internal parts of a cochlear implant.

Figure 2: Cochlear Prosthesis Components. (a) External components are (1) microphone, (2) external speech processor, (3) cable to transmitter, (4) transmitter coil. (b) Internal components are (5) receiver/stimulator, (6) electrode array, and (7) vestibulocochlear nerve.

2.1 Cochlear Implant Signal Processing

The external parts of a cochlear implant consist of a microphone, a speech processor, a transmitter/receiver, and an electrode array as shown in Fig. 2a. The speech processor is the function that can be emulated in MATLAB. There are four steps in cochlear speech processing: Pre-emphasis, multiple band pass filtering, envelope detection, and bipolar pulse generation. Figure 3 shows the signal flow through these blocks. The last step, bipolar pulse generation, is needed to produce the appropriate signal to send to the current stimulator, and then onto the electrode array, causing nerve stimulation. This step (of current stimulation) will not be duplicated in the MATLAB code. Instead, the fourth step will be envelope modulation and summation of the channels, which will allow the processed sound to be heard.

2.1.1 Pre-emphasis Filter

A pre-emphasis filter is often used to amplify high frequencies and attenuate low frequencies prior to the rest of the processing. This filtering enhances the low-energy, high-frequency consonants with respect to the high-energy, low-frequency vowels in a speech signal. A simple FIR filter such as a *first-difference* filter can provide this capability. Recall that a first-difference filter would completely remove DC from the input speech waveform. The low frequency cutoff for human hearing is generally taken as 20 Hz and most microphones and speakers used on PCs are unable to record or play frequencies in these low ranges anyway.

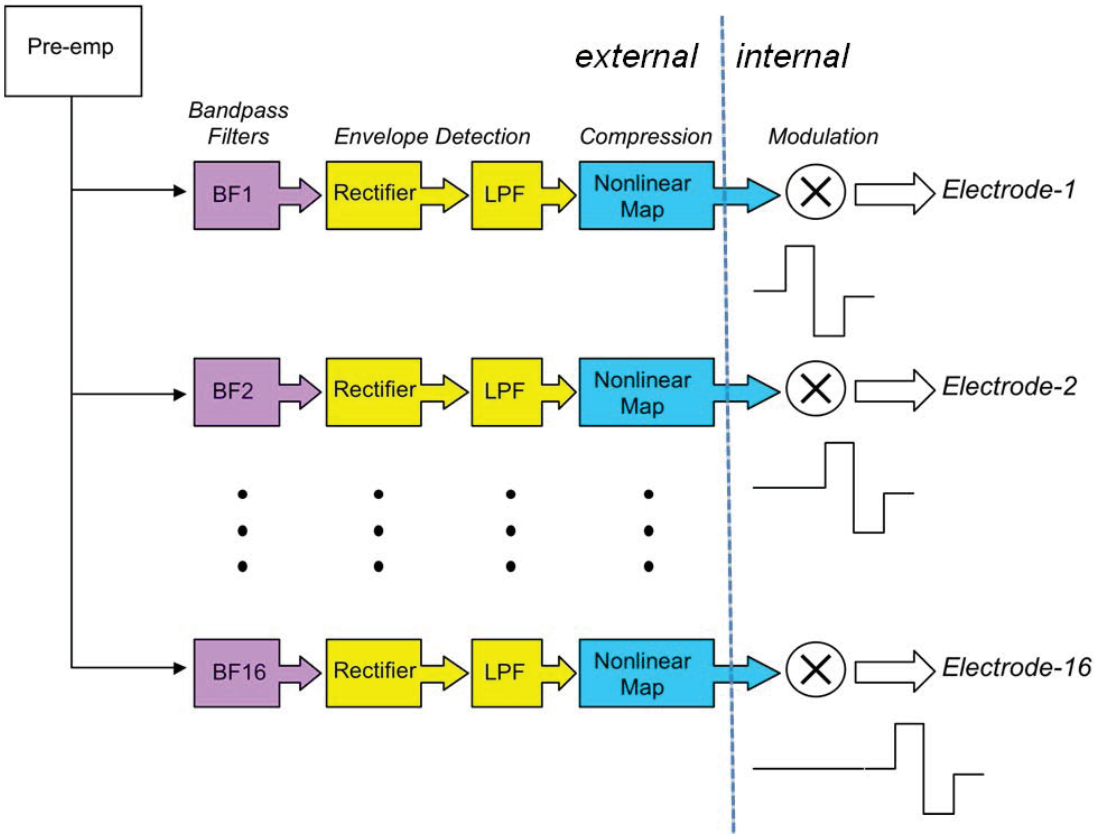


Figure 3: Sixteen channel filter bank.

Table 1: Center frequencies and bandwidths for a 16-channel filter bank (frequencies in Hz)

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
f_c	216	343	486	647	828	1031	1260	1518	1808	2134	2501	2914	3378	3901	4489	5150
B	120	135	151	170	192	216	242	273	307	345	389	437	492	553	622	700

2.1.2 Bandpass Filter Bank

At this point in the speech processing, the pre-emphasized sound waves are run through a filter bank. Table 1 gives the center frequency and bandwidth values for a 16-channel filter bank. Each channel in the filter bank is a bandpass filter (BPF) that separates out a small range of frequency components of sound signal, i.e., the frequencies within its pass band. The pass band of each BPF is from $f_c - \frac{1}{2}B$ to $f_c + \frac{1}{2}B$, so the width of the pass band is B. For example, the fifth channel given in Table 1 has a BPF that passes frequencies from 732 Hz to 924 Hz. When designing a cochlear implant, a major question would be how many frequency channels to use, and how to choose the passband width of the channels. Too few channels and the reconstructed sound will be unintelligible; however, with too many channels the computational load from the signal processing will be too great and will make it difficult the implement a system that is (relatively) inexpensive and has low power dissipation for longer battery life. The choice of center frequencies and bandwidths for the filter bank must be consistent with the frequency-to-place mapping of the cochlea. It turns out that the bandwidths (B) of the filter banks are logarithmically spaced, i.e., the difference between successive $\log(B)$ values is constant. Equivalently, the ratio between bandwidths of successive channels is constant.

2.1.3 Envelope Detection

Once the sub-band signals of the pre-emphasized speech have been determined, detection of the envelope for each sub-band is necessary. The envelope is used to modulate the current pulses that stimulate nerve fibers as addressed in Section 2.1.4. Each sub-band signal will be full-wave rectified and low-pass filtered to obtain the envelope. Full-wave rectification obtains the magnitude of the signal and low-pass filtering serves to smooth out the signal.

2.1.4 Envelope Modulation

The external signal processor for the cochlear implant produces the envelope signals, one for each channel, and then transmits these signals to the internal receiver which must use the envelopes to stimulate auditory nerve fibers at different locations along the electrode array. Amplitude-modulated biphasic signals are used to stimulate with a current output at each electrode site. *Location along the cochlea maps to frequency, so each frequency band in the filterbank is assigned to an individual electrode position.* For example, the first channel in the filter bank corresponds to the most apical (deepest) electrode site, hence the lowest frequency for stimulation. Similarly, the sixteenth channel in the filterbank corresponds to the most basal electrode site, the highest frequency.

We cannot simulate the interaction between the electrodes and the auditory nerve fibers that create the different frequency sounds we hear. Instead, we will create an acoustic simulation of a CI signal processor, meaning that our simulation will be "acoustically faithful." We use the slowly varying envelope signal from one channel to restore a signal at the center frequency of that channel, so the envelope for the n-th channel will modulate a sinusoid whose frequency is set to the center frequency of the n-th channel.[2] The sinusoids from all the channels are then summed to produce a speech signal that is like what is heard by a patient with a cochlear implant.

2.2 GUI for Filter Design

The *SP-First* GUI called `filterdesign` illustrates several filter design methods for LPF, BPF and HPF filter. The interface is shown in Fig. 4. Both FIR and IIR filters can be designed, but we will only be interested in the IIR case which would be selected from the radio button in the upper right. Once `IIR Filters` is selected, the filter specifications can be entered in the text boxes on the right. For the bandpass filter shown, the lower stopband extends from 0 to 530 Hz, and the upper stopband from 1185 to 2756.25 Hz which is half the sampling rate. The value of the magnitude (of the frequency) response in the stopbands must be less than 0.1 which is the specified stopband deviation, δ_s. The passband of this filter goes from 697 to 941 Hz and the magnitude response must be between 1 and $1 - \delta_p$ where δ_p is the allowable deviation in the passband. Lastly, it is necessary to set the order of the IIR filter which is the number of poles. The order can be set manually, but it is also possible to obtain the order from a formula that is built in to the GUI.

2.2.1 Design Some IIR Filters

For practice, design some bandpass IIR filters where the 3-dB passband goes from 0.3π to 0.35π in the $\hat{\omega}$ domain. In the first case set the order to 2, i.e., two poles. Set the stopbands to be $[0, 0.2\pi]$, and $[0.45\pi, \pi]$. Use the Show Radian Frequency under the Edit menu to get the $\hat{\omega}$-axis. Notice that this filter will not satisfy the specifications, so determine the actual stopband edges.

Next, set the GUI to Auto Order to see that a filter will be obtained to meet the specs by increasing the order. Determine where the poles and zero of this filter lie, and verify that the number of poles equals the filter order. A pole-zero plot can be viewed by right-clicking on the frequency response plot to bring up a context menu with other plots. Finally, the filter coefficients can be exported to the workspace by using the

[2] In reality "The amplitudes of the sinusoids are computed by estimating the root-mean- square (rms) energy of the envelopes every 4 msecs." Loizou ref.

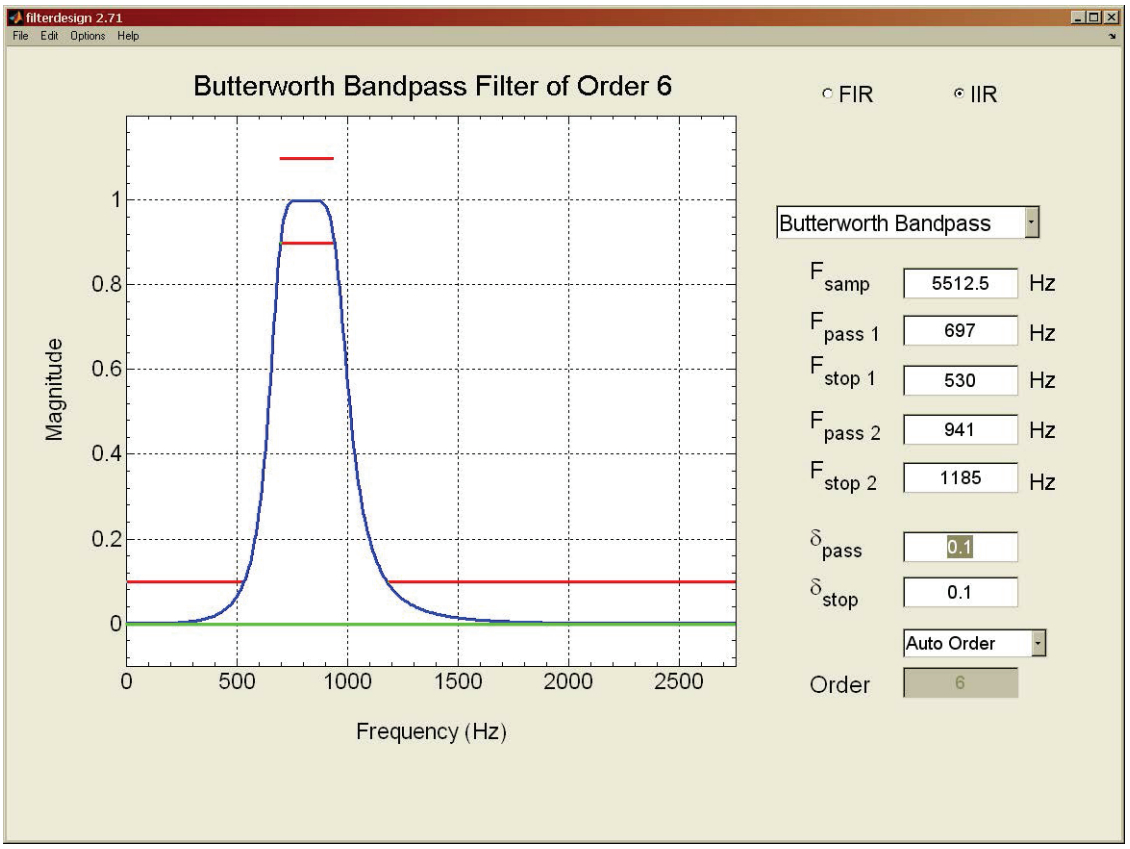


Figure 4: Interface for the `filterdesign` GUI. The default setting for the frequency axis is analog frequency f in Hz, but can be changed to $\hat{\omega}$ via the `Edit->Show Radian Frequency` menu slection. When the Filter Choice is set to `IIR Filters`, the design method gives a Butterworth filter to meet the specifications which are shown as red lines in the frequency response plot.

Export Coeffs under the File menu. With the filter coefficients in the workspace, the MATLAB function `roots` can be used to find the roots of the numerator and denominator polynomials, i.e., the zeros and poles.

2.3 Bandpass Filter Design

You will need a bandpass filter design function for this lab. Second-order IIR bandpass filters can be obtained directly from the poles and zeros. In a previous lab, you experimented with second-order IIR filters whose frequency response looks like Fig. 5.

(a) Write a few lines of MATLAB code to make the plot in Fig. 5(a), where the frequency axis is $\hat{\omega}$.

(b) Modify the code in part (a) to change the frequency axis to get the plot in Fig. 5(b). Recall that the relationship between analog frequency and digital frequency is $\hat{\omega} = 2\pi(f/f_s)$.

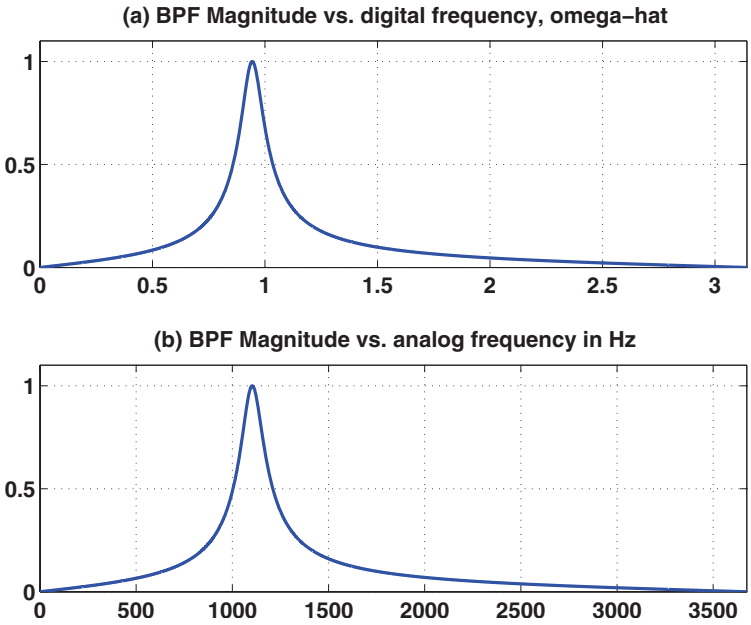


Figure 5: Frequency response of a second-order IIR bandpass filter (BPF) created in MATLAB with numerator `bb = 0.01*[1,0,-1];` and denominator `aa = poly(0.95*exp(j*0.3*pi*[1,-1]));`. (a) $\left|H(e^{j\omega})\right|$ plotted versus $\hat{\omega}$, and (b) the same frequency response versus analog frequency f (Hz), assuming that $f_s = 8000$ Hz.

3 In-Lab Exercises

3.1 Pre-emphasis Filter

Create a pre-emphasis filter to amplify high-frequencies with respect to the low frequencies.

(a) Define a pre-emphasis filter as $H(z) = 1 - z^{-1}$, which is a first-difference FIR filter. Make a plot of its frequency response (magnitude only) by using `freqz` and `plot`. Notice what type of filter this is, i.e., LPF, HPF, BPF or something else.

(b) Apply $H(z)$ to a speech signal, e.g., `catsdogs.wav`. To show the effect of the filter, make spectrograms of both the input signal and the output signal. Describe features in the spectrograms that confirm the frequency response behavior of the pre-emphasis filter.

Instructor Verification (separate page)

(c) Listen to the speech signal before and after pre-emphasis. Can you hear the difference in high-frequency spectral content? It might not be easy unless you have high-quality headphones.

3.2 Envelopes via Full-Wave Rectification and Lowpass Filtering

The output of each BPF in a filter bank is a *narrowband* signal because it only has frequency components near the center frequency of the BPF. The narrowband output signal can be modelled as the product of two components: a sinusoid at the center frequency of the channel and a slowly varying (nonnegative) envelope that changes the amplitude of the sinusoid versus time. This representation is approximate, but it models the output as an AM (amplitude-modulated) sinusoid which is useful.

The objective in *envelope extraction* is to separate out the slowly varying envelope signal. The envelope signal has its spectrum centered at zero frequency and is a signal with relatively low bandwidth. One common way to extract the envelope is to use a cascade system consisting of a full-wave rectifier (i.e., a

magnitude device) followed by a lowpass filter. These two operations for extracting the envelope signal will be treated in more detail in this part of the in-lab exercises.

To demonstrate the processing we need a test signal that is the product of two components: a slowly varying component and a much higher frequency component. The simplest example of this sort is the *AM signal* studied in Chapter 3 and given below

$$b(t) = b_1(t)b_2(t) = (\beta + \cos(2\pi f_1 t + \varphi_1))\cos(2\pi f_2 t + \varphi_2) \qquad \text{where } f_1 \ll f_2$$

where $b_2(t)$ is a high-frequency sinusoidal signal. The slowly varying signal $b_1(t)$ will control the amplitude of the sinusoid $b_2(t)$. This behavior would be obvious from a plot of $b(t)$ versus t.

Extracting the envelope signal, in this case $b_1(t)$, from $b(t)$ can be done with two simple processing steps: (1) a magnitude operator, and (2) a lowpass filter. In a hardware implementation, a full-wave *rectifier* would be used to perform the magnitude operation.

(a) Generate an amplitude-modulated sinusoid using

$$b(t) = (\beta + \cos(2\pi f_1 t))\cos(2\pi f_2 t)$$

Pick $f_1 = 81$ Hz, $f_2 = 1031$ Hz, and $\beta = 1.1$. Use a sampling rate of $f_s = 8000$ Hz to create the vector of signal values in MATLAB; call it `bb`. Create the signal vs. time for a duration of 1.3 s. In a plot of the signal, zoom in to find the feature of the plot that is the envelope of the signal.

(b) In MATLAB take the magnitude of `bb` and plot the magnitude signal versus time t. Once again zoom in to see details, and then point out the feature of the signal that is the envelope. Notice that you will also see the detailed high-frequency nature of the signal.

(c) Compare spectrograms of $b(t)$, $b_1(t)$ and $|b(t)|$ to see where these three signals have frequency content. Use the spectrograms to justify that a LPF applied to $|b(t)|$ will yield the envelope $b_1(t)$.

Instructor Verification (separate page)

(d) Since lowpass filters are well-known for making signals smooth, we need a LPF that will *pass* the frequency range where we expect to find the envelope signal, and *reject* the frequency range where the high frequency components lie. The spectrograms done in the previous part should aid in choosing the passband of this LPF.

Use the `filterdesign` GUI to create a LPF with its cutoff frequency at an appropriately chosen cutoff frequency. The IIR-LPF should have three poles. Since the `filterdesign` GUI can take inputs that are analog frequencies, the cutoff frequency can be chosen directly in hertz. Use the GUI's plot of the frequency response to verify the correct locations of the passband and stopband. Finally, show the pole-zero plot (in the GUI) for the system function of the lowpass filter.

(e) Export the filter coefficients from the GUI in order to use them with the `filter` function to process the magnitude signal $|b(t)|$.[3] Make a plot of the output signal from the LPF, which will be the filtered magnitude. Compare this output to the expected shape of the envelope. Point out similarities and differences; be aware that there might be a fixed scaling between the two plots. Notice the *transient* portion of the output signal at the beginning of the output signal—this is an interval where the output signal is unlike the rest of the waveform.

Instructor Verification (separate page)

[3]The GUI will export to the default names `num` and `den`; sometimes, the numerator coefficients are negated.

4 Lab Project : For Your Summer Enjoyment

First, an observation about learning in the context of lab projects. For first-year or second-year students, labs are given with lots of step-by-step instructions. However, upon graduation you will encounter projects with little or no detailed instructions. Thus, one learning experience is to attempt a project without much hand-holding. If that were the case, this write-up could actually be no more than the following three-sentences.

> *A concise description of the lab project is to implement the entire signal processing system shown in Fig. 6 for simulating the cochlear implant system. Once you have the system working, you should run tests on several types of input signals, including male/female speech and music. Finally, you should vary the number of channels in the filter bank and compare the results.*

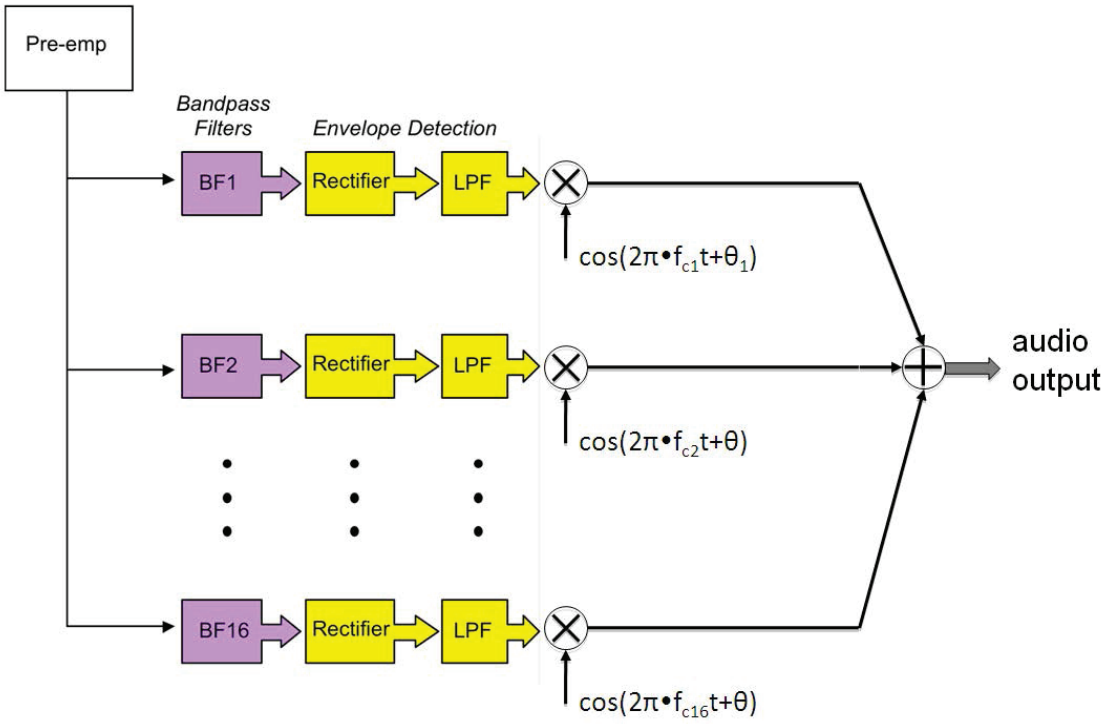


Figure 6: Acoustic simulation of the multi-channel filter bank for a cochlear prosthesis.

4.1 Special Considerations

4.1.1 Sampling Rate

Although many of the previous labs have used speech signals sampled at $f_s = 8000$ Hz, for this lab a higher sampling rate must be used because the highest frequency BPF in the filterbank has a passband that extends up to 5500 Hz. Furthermore, the magnitude operator used in envelope extraction is a nonlinear system that will produce high-frequency spectral components. The best choice would be $f_s = 22,050$ Hz, so make sure that your implementation works for this sampling rate. In addition, it is relatively easy to acquire a recording at $f_s = 22,050$ Hz or $f_s = 44,100$ Hz because those are common rates related to the sampling rate used on CDs. You can increase/decrease the sampling rate by using the MATLAB M-file called `resample`, e.g., you could change from $f_s = 8000$ Hz to $f_s = 16,000$ Hz. If you write your code with a flexible sampling rate, assume that the rate will be no less than 16,000 Hz. From a previous lab you might have a recording of your own voice that you have already studied, so that signal could be used for some of the evaluations below. Other recordings are available from the resource area of `t-square`.

4.1.2 Components Available

In the in-lab exercises, the pre-emphasis filter and the envelope extraction processes were studied. Assuming that you have successful implementations of these components, only minor modifications will be needed to use them in the final system simulation.

4.2 Complete Simulation of the Cochlear Implant System

The CI system requires many filters, so filter design is the core activity in this lab.

4.2.1 Implement the Filter Bank with IIR BPFs

The filter bank consists of many IIR-BPFs running in parallel. In Table 1, we see that the passband widths of the filters are all different. In the IIR case, second-order IIR bandpass filters can be designed by selecting the pole angle from the center frequency of the bandpass, and the pole radius from the passband width. Thus it will be necessary to write a loop that takes the center frequency and bandwidth of each channel, determines the radius and angles of the two poles for each IIR filter. Plotting all the frequency responses together would illustrate how the channel BPFs cover the entire band of the input signal. The implementation of the filter requires filter coefficients when calling `filter` to process the signal The filter coefficients can be determined from the poles and zeros.

4.2.2 Envelope Extraction

The output of the BPFs are narrowband signals that can be viewed as the product of a slowly varying envelope signal with a high-frequency sinusoid at the center frequency of the BPF. The envelope can be extracted by taking the magnitude and then averaging with a smoothing LPF.

(a) A LPF is used because the frequency content of the envelope is concentrated in the low-frequency region. The LPF should be designed as a 3-pole Butterworth IIR LPF with the `filterdesign` GUI so that the passband can be controlled to be flat over a relatively wide frequency band. One question that must be addressed is whether to use a different LPF for each channel, i.e., different passband edges. The following two approaches are suggested for study: (1) use the same LPF in each channel, (2) use two or three different LPFs, one for the very narrow low-frequency BPFs, one for the wider high-frequency BPFs, and one for the mid-range BPFs.

You should address this issue and make a decision for your implementation, which should be justified in your report. Consider the two factors that constrain the cutoff frequency of the smoothing LPF, and describe how these factors vary across the set of BPFs.

(b) The magnitude operation produces a signal that is nonnegative, so it has a nonzero average value. This is DC content of the envelope which can be viewed as a constant offset. If we view the envelope as containing the interesting temporal variations in the signal, we could argue that this constant offset would convey no information. For the filter design, this suggests that we need a filter that is lowpass-like, but one that rejects DC. However, direct design of such a modified LPF could be difficult. Instead, we can cascade two filters to do the job: the Butterworth LPF from the previous part followed by a *notch filter* that removes DC.

A simple, but effective, notch filter can be produced with an IIR system function that has a zero exactly on the unit circle together with a pole close by.

$$H(z) = \tfrac{1}{2}(1+a)\frac{1-z^{-1}}{1-az^{-1}}$$

The pole location can be varied to sharpen the notch by moving the pole very close to the zero. Plotting a frequency response will demonstrate the "notch" capability of this filter; pick a value for a and justify your choice. The pole must be inside the unit circle; why?

(c) Plot the frequency response of the cascade system consisting of the Butterworth LPF from part (a) and the DC-notch filter from part (b).

(d) Compare the sound outputs and spectrograms with and without the DC-notch filter.

4.2.3 Envelope Modulation

The envelope modulation is done by multiplying the slowly varying extracted-envelope signal by a high frequency signal that represents the channel. For the acoustic simulation done in this lab, the appropriate choice would be a sinusoid at the center frequency of the channel.

Note: Excellent sound quality is *not* the primary objective when implementing the CI filter bank. Instead, the objective is to create a sound output that portrays how the user of a CI device would experience sound. The sinusoidal modulation is an easy way to combine the envelope signals, but it is sensitive to the DC offset of the envelope. Hence, the DC-notch filter might make the output sound better. An alternative modulation is to multiply the envelope by a random signal whose frequency content lies in the channel of interest.

4.2.4 Different Number of Channels

You should initially write the simulation so that it works for sixteen channels. With properly designed filters a sixteen-channel system should give intelligible speech. An interesting question is determining the appropriate number of channels for the filter bank. Therefore, you should also provide a simulation of the four-channel case. Below are the lists of the center frequencies and corresponding bandwidths for the 4-channel and 16-channel cases:

```
fc4 = [460 953 1971 4078];
bw4 = [321 664 1373 2842];

fc16 = [216 343 486 647 828 1031 1260 1518 1808 2134 2501 2914 3378 3901 4489 5150];
bw16 = [120 135 151 170 192  216  242  273  307  345  389  437  492  553  622  700];
```

4.2.5 FYI: Defining the Channel Bandwidths

The information above was provided by a study examining the number of channels needed to understand speech.[4] Here is an algorithm for choosing the bandwidths in a N-channel filter bank:

1. Determine the range of frequencies to be covered by the filterbank, say $[f_a, f_b]$.

2. The ratio between bandwidths for neighboring channels is $B_n = \rho B_{n-1}$, where ρ is a constant (greater than one) that is the ratio of successive bandwidths. Decide on a value for ρ, e.g., in the 16-channel filter bank of Table 1, $\rho = 9/8 = 1.125$, and $[f_a, f_b] = [156, 5500]$. (The numbers in the table were obtained by rounding f_c and B to integer values.)
   The eight-channel case is $\rho = 5/4 = 1.25$, and $[f_a, f_b] = [262, 5500]$; and the four-channel case is $\rho = 2.07$, and $[f_a, f_b] = [300, 5500]$

3. Then we can solve for B_1 from the following:

$$f_b - f_a = \sum_{n=1}^{N} B_n = \sum_{n=1}^{N} \rho^{n-1} B_1 = B_1 \sum_{n=0}^{N-1} \rho^n = B_1 \frac{1-\rho^N}{1-\rho}$$

   Recall that the bandwidths are generated from the recursion $B_n = \rho B_{n-1}$.

4. The center frequencies are generated via:

$$f_{c_1} = f_a + \tfrac{1}{2} B_1 \qquad \text{and} \quad f_{c_n} = f_{c_{n-1}} + \tfrac{1}{2} B_{n-1} + \tfrac{1}{2} B_n$$

[4] P. C. Loizou, et. al., "On the number of channels needed to understand speech," *J. Acoustical Soc. of America,* vol. 106, pp, 2097–2103, 1999.

4.3 Testing

You should perform at least two types of tests with your working system. First, vary the input signal by using male and female speakers. Describe any differences that you hear by finding cases where the system works well and other cases where you can observe serious degradations. Show a couple of spectrograms to help explain how the system works. Spectrograms would also be useful to compare the final output to the original—a fair comparison would be to compare to the pre-emphasized original.

Second, testing with music input would also be interesting since a patient with a cochlear implant would have the device optimized for speech, but would be hearing all sorts of sounds. One issue would be the bandwidth coverage of the filter bank with respect to the expected range of frequencies in music. Spectrogram would help in comparing the output to the original input.

In your tests you should vary the number of channels in the filter bank, and listen for intelligibility. Fewer channels would make for a less-expensive system because the hardware would be a lot simpler.

4.3.1 Debugging

Here is one hint on debugging: use a single sinusoid as the input signal. When the frequency of the test sinusoid equals the center frequency of one of the channels, it seems that you should get (almost) perfect reconstruction at the output. In any event, you can analyze how the sinusoid goes through the system because there is only one spectral line to track.

4.3.2 Demonstration

When you turn in the lab report, a short demonstration of the working system will be necessary. The purpose of the demo will be to show how your system handles music, as well as speech. In addition, you can explain the design decisions that were made when creating the bandpass filters for the filter bank.

Lab #12
ECE-2026 Spring-2015
INSTRUCTOR VERIFICATION SHEET

Turn this page in to your TA before the end of your lab period.

Name: _____ Date of Lab: _____

Part 3.1 Demonstrate the pre-emphasis filter by showing spectrograms of a speech signal before and after filtering. Compare the spectrograms and point out changes after filtering.

Verified:_____ Date/Time:_____

Part 3.2(c) Show the magnitude of the AM signal vs. time. Point out the feature on the plot that would be called the envelope. Explain the frequency content of $b(t)$ and $b_1(t)$ versus that of $|b(t)|$.

Verified:_____ Date/Time:_____

Part 3.2(d) Explain how you designed the lowpass filter with the `filterdesign` GUI, and how you used the filter coefficients. Write the call to the MATLAB function `filter` below. In addition, write out the difference equation that corresponds to the LPF.

```
yy = filter(
```

$y[n] =$

Verified:_____ Date/Time:_____

Part 3.2(e) Plot the output signal from the LPF, and compare to the known envelope. Explain differences.