Modulation Toolbox Tutorial and User Guide
Version 2.1

Pascal Clark and Les Atlas
{clarkcp, atlas}@u.washington.edu

7 September 2010

Contents

1 Introduction

1.1 What is the Modulation Toolbox?

The Modulation Toolbox is a set of MATLAB functions for analyzing and modifying modulation frequency spectra in speech, music and other natural sounds. In an acoustic signal, we assume that information-bearing components are of the form

$$x(t) = m(t) \cdot c(t) \tag{1}$$

where $x(t)$ is an observed time-domain signal, and $m(t)$ is a low-frequency modulator multiplied by a high-frequency carrier $c(t)$. The toolbox provides functions that estimate discrete-time $m[n]$ and $c[n]$ from the samples of the observed signal $x[n]$, a problem known as *demodulation*. Once estimated, $m[n]$ represents time-domain envelope fluctuations that are subject to modulation-frequency analysis via the Fourier transform and filtering theory [1][2][3].

As of April 2009, version 2.x of the Modulation Toolbox is a significant expansion on the previous release with the inclusion of new *coherent* demodulation algorithms. Unlike the conventional Hilbert envelope, coherent demodulation explicitly estimates narrowband carriers of a signal in order to compute the modulators. With the appropriate constraints, coherent carrier detection guarantees properties for effective and distortion-free filtering of modulators [1]. Version 2.1 is the latest release of the Modulation Toolbox encompassing multiple demodulation algorithms, coherent and non-coherent, for comparison in signal processing applications.

The Modulation Toolbox is the result of many contributors under the supervision of Professor Les Atlas. Versions 2.0 and 2.1 were authored between 2008 and 2010 by Pascal Clark with major contributions from Adam Greenhall, Elliot Saba, Brian King, and Xing Li. It is built upon the earlier version (v1.23) authored by Steven Schimmel, Chad Heinemann, and Jeff Thompson.

1.2 Who Should Use the Modulation Toolbox?

The Modulation Toolbox is intended for researchers and students. It is freely available for non-profit purposes from the ISDL website at

http://isdl.ee.washington.edu/projects/modulationtoolbox

To use the Modulation Toolbox, we recommend having a basic understanding of Fourier transforms, filters, and spectrograms. The Modulation Toolbox was written and tested with MATLAB version R2007b in Windows, and it requires the MATLAB Signal Processing Toolbox.

If you are using version 2.0 then there a few things to keep in mind when deciding to upgrade to version 2.1. First, version 2.1 is largely the same as 2.0 in terms of signal processing. The difference is that 2.1 includes new high-level functions with flexible user interfaces and built-in default parameter settings. These new functions represent the primary elements of decomposition (`moddecomp`), modulation spectral analysis (`modspectrum`), modulation filtering (`modfilter`), and audio synthesis (`modsynth`). However, version 2.1 is not entirely backward-compatible with 2.0. As a result of small changes throughout the toolbox codebase, version 2.1 may not be compatible with your existing scripts. Refer to Section 7 for details.

1.3 Terms of Use

1.4 Contact Information

The Modulation Toolbox is designed to further the field of modulation research through experimentation and collaboration. We welcome any questions, suggestions, or comments that you might have. Please direct your feedback to Pascal Clark [clarkcp@u.washington.edu] and Professor Les Atlas [atlas@u.washington.edu].

If you publish results obtained from the toolbox software, please use the following format for your citation:

Les Atlas, Pascal Clark, and Steven Schimmel, *Modulation Toolbox Version 2.1 for MATLAB*, September 2010, University of Washington, http://isdl.ee.washington.edu/projects/modulationtoolbox/

1.5 Overview

The contents of this user guide and tutorial are as follows. Section 2 introduces and motivates the need for coherent demodulation in modulation spectral analysis. Section 3 provides a working knowledge of demodulation, analysis, and modification as implemented by the toolbox. Then, Section 4 discusses the real-world meaning of complex-valued coherent modulators. Sections 5 and 6 give an overview of the Modulation Toolbox codebase, followed by a summary of new developments in version 2.1 in Section 7. A list of references appears at the end, prior to appendices related to coherent carrier estimation.

2 Modulation Spectral Analysis and Filtering

This section motivates and introduces the basics of modulation spectral analysis and filtering by way of a synthetic example and an actual speech example. Refer to `tutorial1_modulationFrequency.m` for the source code.

2.1 A Synthetic Example

Consider the following test signal consisting of a low-frequency modulator multiplied by a linear chirp (starting at 800 Hz and increasing at a rate of 100 Hz/sec).

$$\begin{aligned} m(t) &= \sin(2\pi 2t) + \sin(2\pi 6t) \\ c(t) &= \sin(2\pi(800t + 50t^2)) \\ x(t) &= m(t) \cdot c(t) \end{aligned} \tag{2}$$

Let the vector `x` contain samples of $x(t)$ at a rate of 16000 Hz over a period of 4 seconds. The modulation spectrum of this signal depends on how we choose to demodulate it. We can plot the coherent modulation spectrum in 500-Hz wide subbands with the command

```
modspectrum( x, 16000, ‘cog’, 500 );
```

and compare it to the non-coherent (Hilbert envelope) modulation spectrum generated by

```
modspectrum( x, 16000, ‘hilbert’, 500 );
```

We will describe the spectral center-of-gravity (COG) and the Hilbert envelope methods in more detail in Section 3. For now it suffices to understand that coherent demodulation estimates bandlimited components without the interference introduced by the conventional Hilbert envelope.

The results of the above function-calls appear in Figure 1. Note that the coherent spectrum accurately represents the peaks in modulation frequency corresponding to the 2-Hz and 6-Hz components of our defined modulator $m(t)$. Conversely, the non-coherent spectrum shows phantom cross-terms at 0, 4 and 8 Hz.

Next we will attempt to filter the modulator in order to isolate the 2-Hz component. One way to do this is with a coherent modulation filter with a 4-Hz lowpass cutoff via

```
yCOG = modfilter( x, 16000, [0 4], ‘pass’, ‘cog’, 500 );
```

and non-coherently via

```
yHilb = modfilter( x, 16000, [0 4], ‘pass’, ‘hilbert’, 500 );
```

The modulation-filter operation is defined as the time-domain multiplication of a filtered modulator with the original detected carrier signal. Thus we expect the

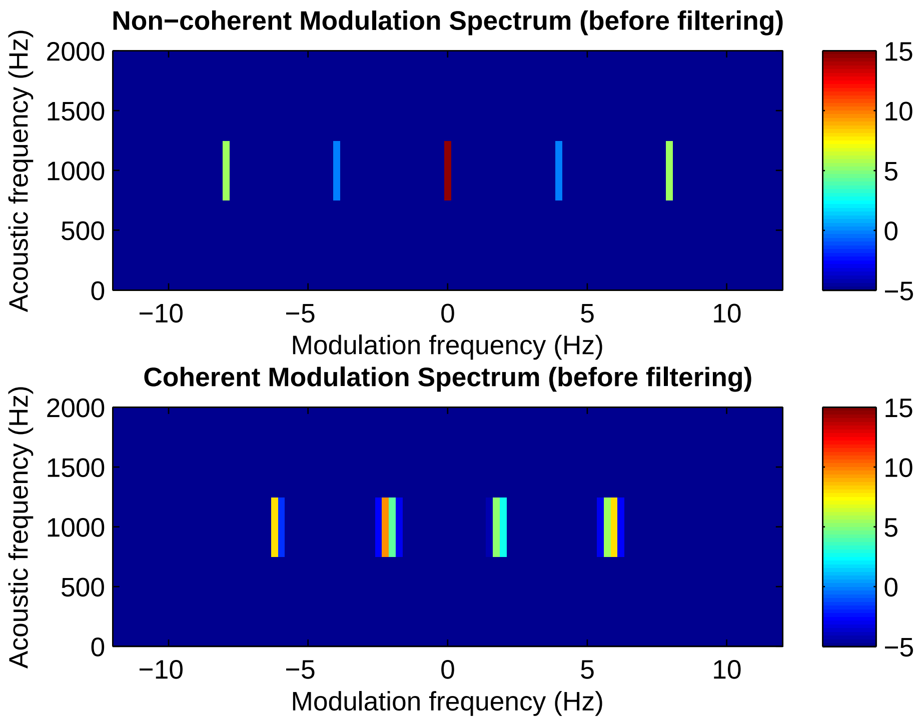


Figure 1: Modulation spectra generated by non-coherent Hilbert envelope demodulation (top) and coherent spectral COG demodulation (bottom). With respect to the modulator definition given in (3) the coherent spectrum correctly displays 2- and 6-Hz modulations. The colormap is in dB.

output of the modulation filter to be

$$\begin{aligned} \hat{m}(t) &= \sin(2\pi 2t) \\ y(t) &= \hat{m}(t) \cdot c(t) \end{aligned} \tag{3}$$

A visual comparison of the time-domain signals in Figure 2 reveals that the coherently modulation-filtered result, `yCOG`, more accurately preserves the desired 2-Hz modulation as judged by both the shape, depth, and phase of the beat pattern. Another reason to choose the coherent approach is that `yCOG` is free of bandwidth distortion, unlike `yHilb`. As seen in Figure 3, the output of the non-coherent modulation filter shows a periodic artifact at a rate of 8 Hz – well above the modulation filter cutoff frequency! The non-coherent filtering artifact is an audible distortion that the reader can verify by running the experiment in `tutorial1_modulationFrequency.m`.

For a more thorough discussion on coherent modulation filtering, refer to [4], [1] and [5].

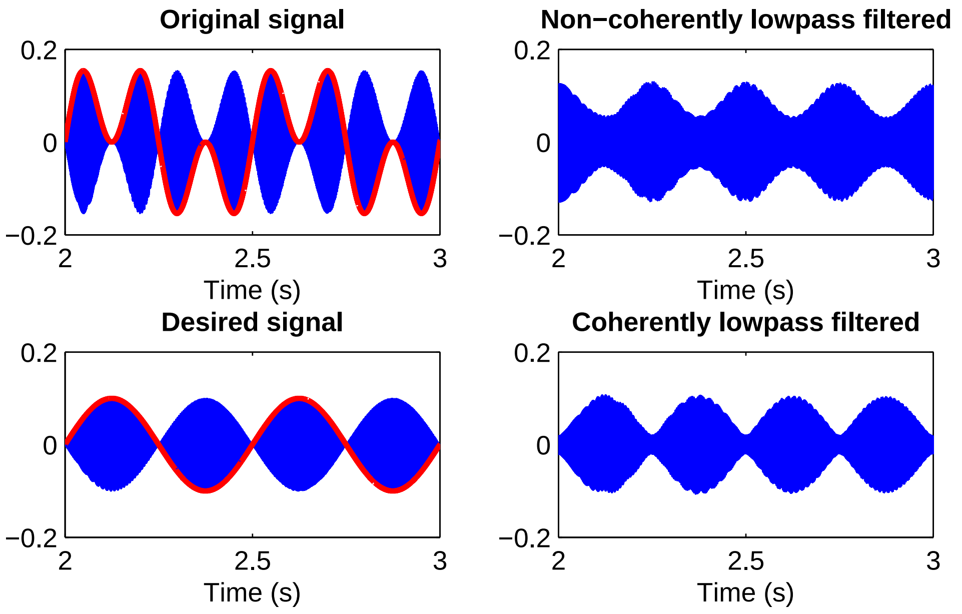


Figure 2: Time-domain plots before and after modulation filtering. Top left: the original synthetic signal with modulator overlaid in red. Bottom left: the desired signal with the 6-Hz modulation removed. Top right: non-coherently filtered signal which shows an incorrect envelope. Bottom right: coherently filtered signal with an envelope qualitatively closer to that of the ideal desired signal.

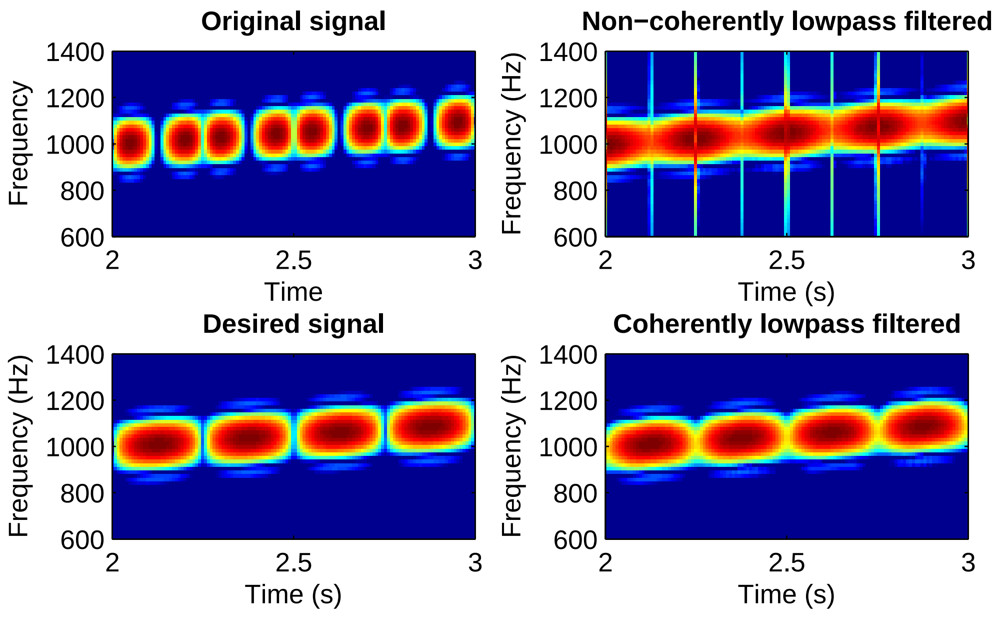


Figure 3: Spectrogram plots before and after modulation filtering, in the same arrangement as Figure 2. Note the 8-Hz transient artifact present in the non-coherently filtered signal in the top right.

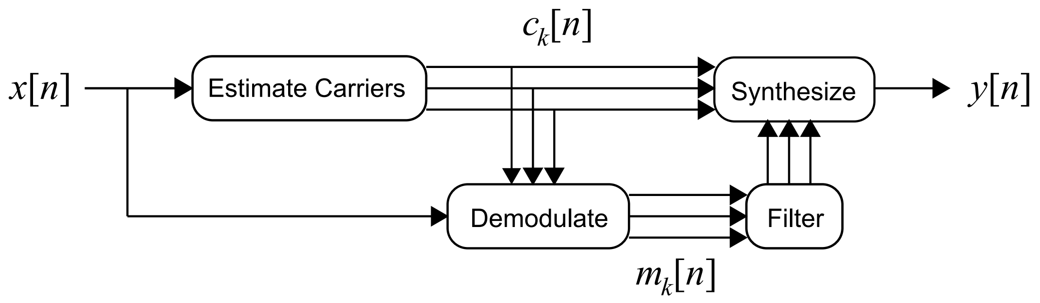


Figure 4: Modulation filtering system block diagram.

2.2 A Speech Example

The previous section demonstrated the shortcomings of the non-coherent Hilbert envelope in a synthetic modulation filtering task. An example of a naturally modulated signal is speech, in which temporal modulations are related to the articulations of the vocal tract. In this section we argue that, compared to the commonly used non-coherent Hilbert envelope, coherent methods more effectively represent speech modulations for the purpose of modulation spectral analysis and filtering.

In the following it is important to understand that modulation spectral analysis assumes the modulators can be studied and modified independent of the carriers. This concept is embodied by the modulation-filtering block diagram in Figure 4, in which filtered modulator waveforms recombine with the original carrier estimates to form a new signal $y[n]$. The synthetic example in Section 2.1 was a particular case where $x[n]$ consisted of a single modulated carrier. For a rich signal such as speech, the system in Figure 4 generalizes to a multitude of modulator-carrier pairs which can nonetheless be modified separately and recombined.

The effectiveness of modulation filtering is one way to test the separability of modulation frequencies from the carriers, whether they are coherently or non-coherently estimated. Consider the following comparison (The reader may refer to `tutorial1_modulationFrequency.m` for implementation details.) We apply a 2-Hz lowpass modulation filter on a 16 kHz speech signal $x[n]$ using coherent harmonic demodulation,

```
yHarm = modfilter( x, 16000, [0 2], ‘pass’, {‘harm’,[],0.3}, 150 );
```

coherent subband demodulation,

```
yCOG = modfilter( x, 16000, [0 2], ‘pass’, ‘cog’, 250 );
```

and non-coherent Hilbert demodulation,

```
yHilb = modfilter( x, 16000, [0 2], ‘pass’, ‘hilb’, 250 );
```

As in Section 2.1, the non-coherent output `yHilb` is noisy as a result of bandwidth distortion. Both coherently filtered signals, however, shows temporal smoothing without bandwidth expansion of the harmonics, which can be verified audibly as

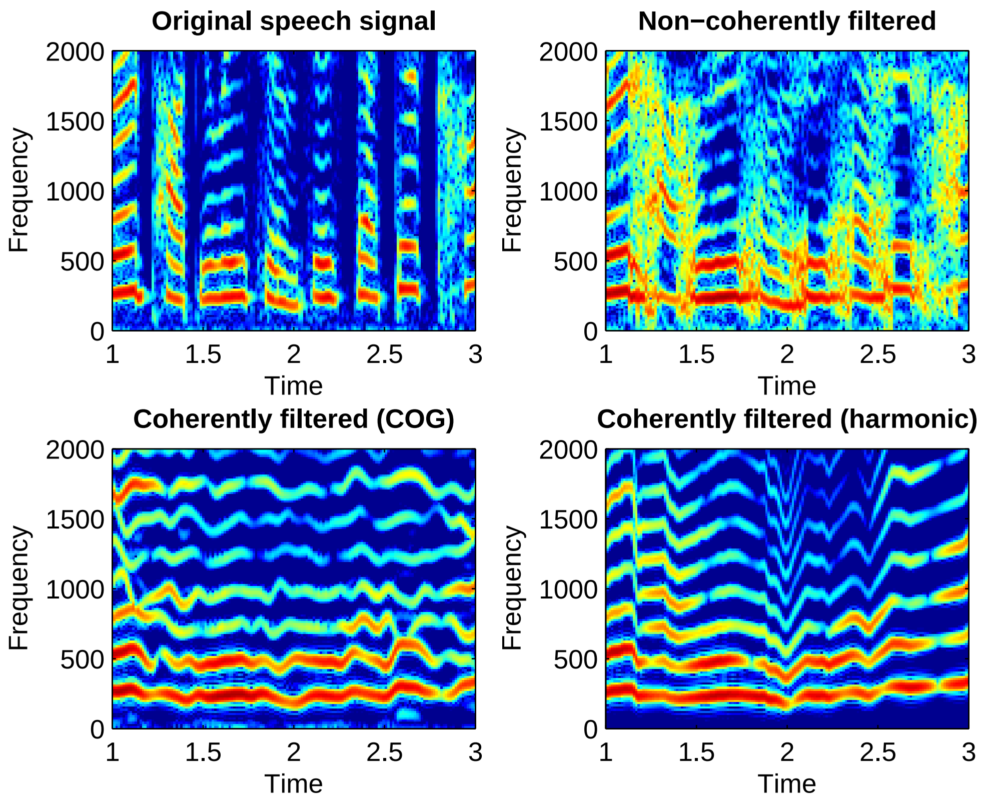


Figure 5: Spectrogram plots before and after modulation filtering. Note the spectral distortion present in the non-coherently filtered signal (top right), compared to the bandlimited coherently filtered signals (bottom left and right).

well as visually in the spectrograms in Figure 5.

With the conclusion of these motivating examples, the next section goes into more detail about the estimation of coherent and non-coherent modulators and carriers.

3 Estimating and Modifying Modulators and Carriers

This section describes coherent and non-coherent demodulation as estimators with an underlying *signal product model* given by

$$x[n] = \sum_{k=0}^{K-1} m_k[n] \cdot c_k[n] \tag{4}$$

where $x[n]$ is an observed, discrete-time fullband signal, $m_k[n]$ and $c_k[n]$ are the respective kth modulator and carrier waveforms, the $(\cdot)$ operator denotes sample-by-sample multiplication, and K is a finite number. We refer to *demodulation* as the process of estimating $m_k[n]$ and $c_k[n]$ given some $x[n]$ for all n and k. For this task it is helpful to define

$$s_k[n] = m_k[n] \cdot c_k[n] \tag{5}$$

where $s_k[n]$ is the kth analytic bandpass subband signal[1], possibly a filterbank subband signal, comprising $x[n]$.

Defining $s_k[n]$ to be analytic means that it is complex-valued. We also define carriers to be complex, as in

$$c_k[n] = \exp(j\phi_k[n]) \tag{6}$$

where $j^2 = -1$. Although complex time-domain signals are difficult to visualize, there are a number of advantages for using complex expansion. We cover these in Section 4. For now, the reader can take comfort in realizing that the complex factorizations in the Modulation Toolbox eventually project back onto the real numbers, since each analytic subband $s_k[n]$ corresponds uniquely to a real-valued bandpass signal.

Whether a demodulation operation is *coherent* or *non-coherent* depends on how $m_k[n]$ and $c_k[n]$ are estimated from a single subband signal $s_k[n]$. We refer to this as "subband demodulation." Alternatively, $c_k[n]$ can be estimated directly from $x[n]$ without an intermediate $s_k[n]$, which is the basis for coherent harmonic demodulation. In the following we give a user's overview of these methods within the Modulation Toolbox, discussing first coherent and non-coherent subband demodulation, followed by coherent harmonic demodulation, and then concluding with modification and audio synthesis.

3.1 Coherent and Non-Coherent Subband Demodulation

Subband demodulation begins by filtering $x[n]$ into bandpass analytic signals $s_k[n]$ defined by a filterbank. Finding the modulator and carrier for each subband signal is an under-determined problem, however, because there are infinitely many factorizations that produce a valid modulator and carrier such that $s_k[n] = m_k[n] \cdot c_k[n]$. Basic axiomatic assumptions can disambiguate the solution [6] [7], and fall into one of two categories: coherent and non-coherent.

[1]The use of analytic signals assumes that $x[n]$ is real-valued. However, the Modulation Toolbox can also decompose complex-valued $x[n]$ by complementing each analytic subband signal with an anti-analytic subband signal.

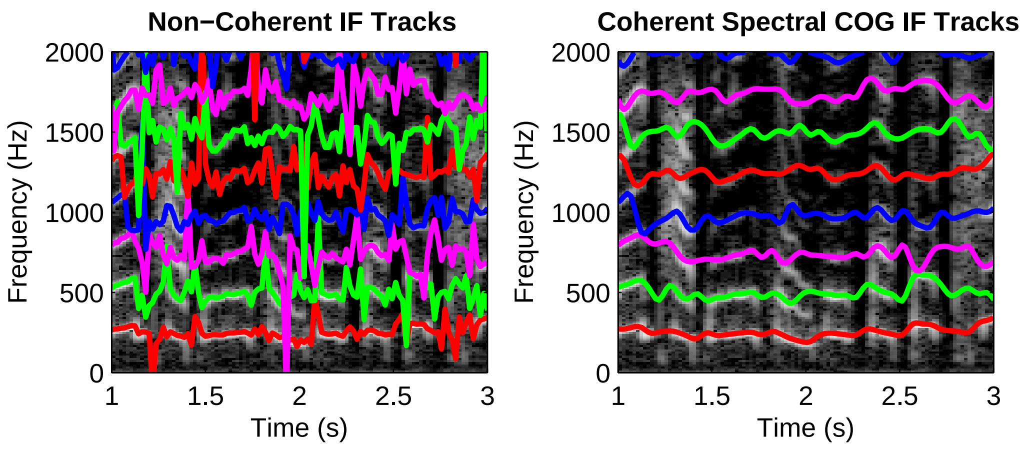


Figure 6: Comparison of subband instantaneous frequency estimates (colored lines) overlaid on top of a speech signal spectrogram (grayscale), for non-coherent (left) and coherent (right) methods.

Perhaps the easiest method to understand is non-coherent subband demodulation, which defines

$$\begin{aligned} m_k[n] &= |s_k[n]| \\ c_k[n] &= \exp(j\angle s_k[n]) \end{aligned} \tag{7}$$

The magnitude of the analytic signal is also known as the Hilbert envelope of $s_k[n]$. The modulator is thus real-valued and non-negative, and the carrier is phase-only and unit-magnitude. Although intuitive and simply implemented, the modulator and carrier are not bandlimited and not individually modifiable (see the examples in Section 2, and more detail in [1]).

Coherent subband demodulation, however, enforces bandwidth constraints on the modulator and carrier in order to allow distortion-free modification of each. The Modulation Toolbox uses the spectral center-of-gravity (COG) method [6] to smoothly track spectral concentration within the subband over time as an estimate of the instantaneous frequency $f_k[n]$. This defines the carrier phase via

$$\phi_k[n] = \sum_{p=0}^{n} f_k[n] \tag{8}$$

Coherent demodulation then generalizes (7) by defining the modulator with respect to the estimated carrier:

$$\begin{aligned} c_k[n] &= \exp(j\phi_k[n]) \\ m_k[n] &= s_k[n] \cdot c_k^*[n] \end{aligned} \tag{9}$$

For more details, refer to Appendix A.

With the appropriate settings, coherent demodulation has the immediate benefit of yielding bandlimited carriers and modulators. Using the default carrier estimation settings, Figure 6 compares subband instantaneous frequency trajectories between

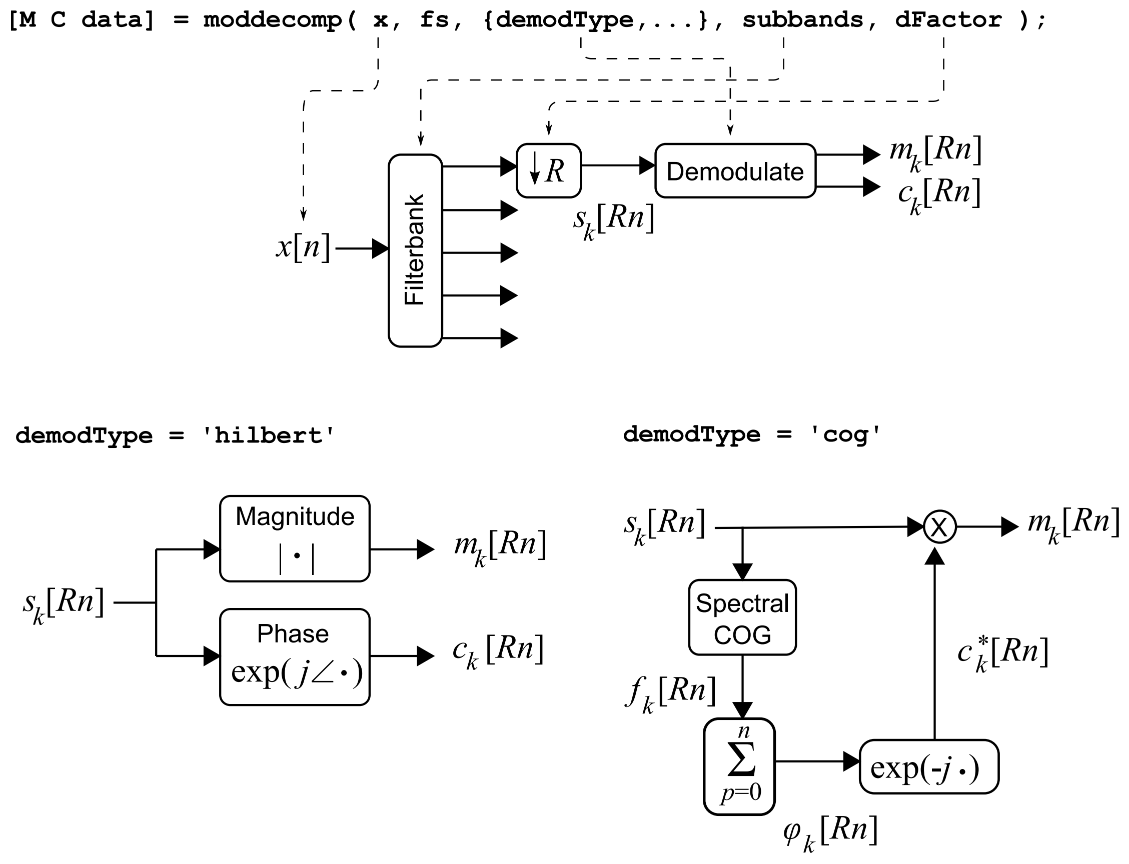


Figure 7: MATLAB function call and system block diagram for multirate subband demodulation.

non-coherent Hilbert and coherent spectral COG. Each colored line corresponds to $f_k[n]$ for a single k over time. The coherent trajectories are noticeably smoother and adhere more closely to the narrowband spectral components in the speech signal.

To summarize, subband demodulation begins with a filterbank decomposition followed by coherent or non-coherent modulator and carrier estimation on each subband signal. The Modulation Toolbox implements this operation with the `moddecomp` command, shown conceptually in Figure 7. The parameter `demodType` is a string equal to 'cog' for coherent or 'hilbert' for non-coherent. For the purpose of computational efficiency, `moddecomp` provides a decimation option with the user-specified parameter `dFactor`. Note that the coherent option will return complex-valued modulator signals. For an interpretation of complex modulation, refer to Section 4.

3.2 Coherent Harmonic Demodulation

An alternative to subband demodulation is to coherently estimate the carriers directly from the signal itself. The general form for this approach is given by the convolved product

$$m_k[n] = \sum_{p=0}^{n} h[n-p] \cdot x[p] \cdot c_k^*[p] \tag{10}$$

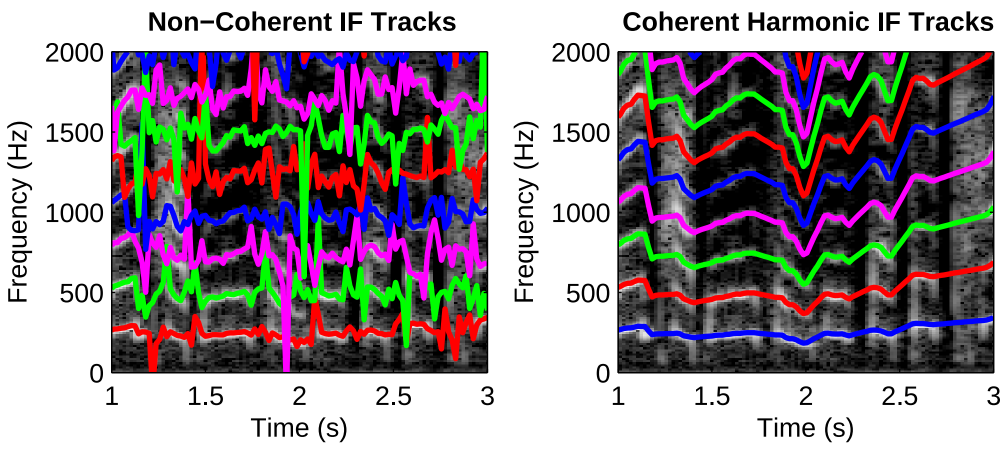


Figure 8: Comparison of subband instantaneous frequency estimates (colored lines) overlaid on top of a speech signal spectrogram (grayscale), for non-coherent (left) and coherent (right) methods.

where $c_k[n]$ is the kth complex carrier signal, $x[n]$ is real- or complex-valued, and $h[n]$ is the impulse-response of a lowpass filter. The expression in (10) is similar to (9) except the convolution follows the product instead of the other way around. Equation (10) reveals demodulation for what it really is: a frequency-shift operation that brings a spectral band in $x[n]$ to baseband where it can be extracted as a lowpass modulator signal.

In the case of harmonic demodulation, the spectral band is related to an integer-multiple of the time-varying fundamental frequency of $x[n]$. This model is appropriate for many musical signals and for voiced speech. As implemented by the Modulation Toolbox, the kth carrier is defined as

$$c_k[n] = \exp(jk\phi_0[n]) \tag{11}$$

where

$$\phi_0[n] = \sum_{p=0}^{n} F_0[n] \tag{12}$$

and $F_0[n]$ is the detected fundamental frequency or "pitch" of the signal $x[n]$. If $x[n]$ is unvoiced or weakly voiced during some time interval $n_1 < n < n_2$ then the built-in pitch detector will fill in the gap by interpolating $F_0[n]$ between the valid detections at $F_0[n_1]$ and $F_0[n_2]$. For details, refer to Appendix B.

Like coherent subband demodulation, a sufficiently smooth estimate of $F_0[n]$ ensures bandlimited modulators and carriers. Figure 8 compares non-coherent subband instantaneous frequencies to those obtained from coherent harmonic demodulation. The primary difference is that harmonic instantaneous frequencies (as a function of pitch-detection parameters) are smooth and bandlimited. Harmonic carriers are also not restricted to artificial subband boundaries, which is an advantage over both coherent and non-coherent subband demodulation strategies.

The Modulation Toolbox implements this operation with the `moddecomp` command, shown conceptually in Figure 9. For the purpose of computational efficiency,

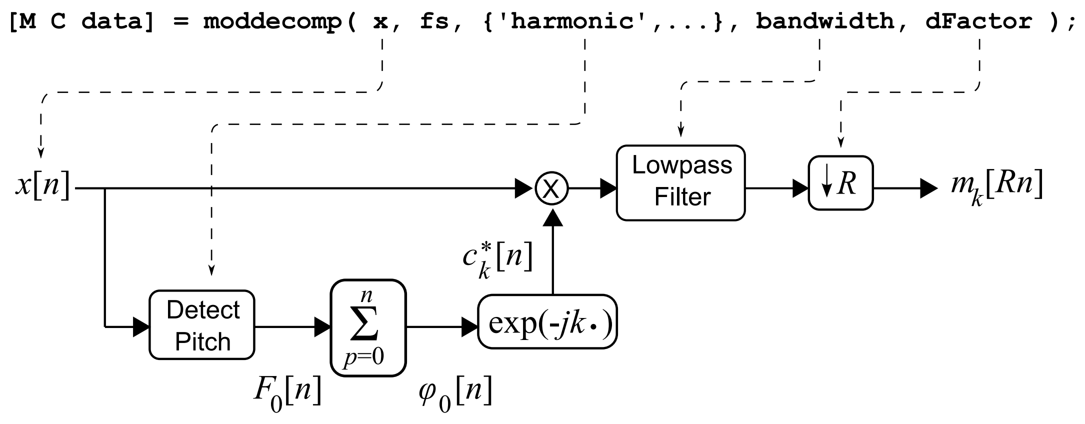


Figure 9: MATLAB function call and system block diagram for multirate harmonic demodulation.

`moddecomp` provides a decimation option with the user-specified parameter `dFactor`. Note that this option will return complex-valued modulator signals. For an interpretation of complex modulation, refer to Section 4.

3.3 Analysis, Modification and Synthesis

The previous sub-sections discussed demodulation as the first step in any modulation-based analysis or synthesis operation. After obtaining the estimated modulators, the Modulation Toolbox provides methods for analyzing the modifying $m_k[n]$ in the spectral domain. The modulation spectrum is the windowed Fourier transform across time

$$P_k[i] = \sum_{n=0}^{N-1} w[n]\, m_k[n]\, e^{-j\frac{2\pi i}{N}n} \tag{13}$$

where $w[n]$ is a data taper such as a rectangular or Hamming window, and N is greater or equal to the number of time dimensions in the array $m_k[n]$. Modulation spectral analysis is closely related to modulation filtering, defined as

$$\tilde{m}_k[n] = \sum_{q=0}^{n} g[n-q]\, m_k[q] \tag{14}$$

or, in the modulation-spectral domain

$$\tilde{m}_k[n] = \sum_{i=0}^{N-1} G[i]\, P_k[i]\, e^{j\frac{2\pi n}{N}i} \tag{15}$$

where $g[n]$ is the impulse response of a linear filter that is time-invariant with respect to $m_k[n]$ (although time-varying with respect to $x[n]$).

The Modulation Toolbox command for modulation spectral analysis is

```
P = modspectrum( x, fs, demodParams, subbands, specOptions );
```

which internally performs demodulation by calling `moddecomp`. The `demodParams` and `subbands` parameters work in the same way as for `moddecomp` and cover subband demodulation (coherent and non-coherent) as well as coherent harmonic demodulation. Calling `modspectrum` without specifying an output will automatically plot the magnitude of the modulation spectrum, as seen in Section 2. Refer to the MATLAB m-file header for usage information.

Similarly, the MATLAB command for modulation filtering is

```
y = modfilter( x, fs, filterBand, filterType, demodParams, subbands );
```

where the parameters `filterBand` and `filterType` together specify the frequency response of the modulation filter $g[n]$. Note that `modfilter` returns a vector `y`, which is the synthesized audio signal $y[n]$ after modulation filtering. This means that `modfilter` implicitly makes a call to `modsynth` in order to execute the synthesis operation

$$y[n] = \sum_{k=0}^{K-1} \tilde{m}_k[n] \cdot c_k[n]. \tag{16}$$

Modulation filtering is represented schematically in Figure 4.

Should you choose to create your own modulator and carrier modifications, the `modsynth` function allows you to combine almost arbitrary `M` and `C` arrays to create an audio signal. The MATLAB function-call is

```
   y = modsynth( M, C, data );
```

where `data` is a data structure, returned by `moddecomp`, containing decomposition implementation details. For example, you may want to use different modulation filters for different subbands of a single audio signal. Mathematically this is equivalent to

$$\tilde{m}_k[n] = \sum_{i=0}^{N-1} G_k[i]\, P_k[i]\, e^{j\frac{2\pi n}{N}i}. \tag{17}$$

The generalized modulation filter in (17) can be visualized as the multiplication of two-dimensional surfaces in the joint-frequency domain [3] [2]. Instead of using `modfilter`, one way to carry this out is with the following toolbox commands:

```
   [M C data] = moddecomp( x, fs, demodParams, subbands );
   M2 = ifft( G.*fft( M, [], 2 ), [], 2 );
   y = modsynth( M2, C, data );
```

where `G` is some two-dimensional array containing the discrete samples of the desired masking function $G_k[i]$.

So far we have covered the four major operations implemented by the Modulation Toolbox: demodulation, spectral analysis, filtering and synthesis. All of these capabilities can be explored using the modulation spectrogram graphical user interface (GUI), which we introduce in the next section.

3.4 Modulation Spectrogram GUI

The *modulation spectrogram* computes the modulation spectrum within a sliding window, usually on the order of hundreds of milliseconds or seconds in duration. To open the modulation spectrogram GUI, type

```
modspecgramgui;
```

into the MATLAB command line. Go to `File -> Open` and select a .wav audio file. Using `speech_male.wav` in the `sounds` folder, for example, will produce the display in Figure 10.

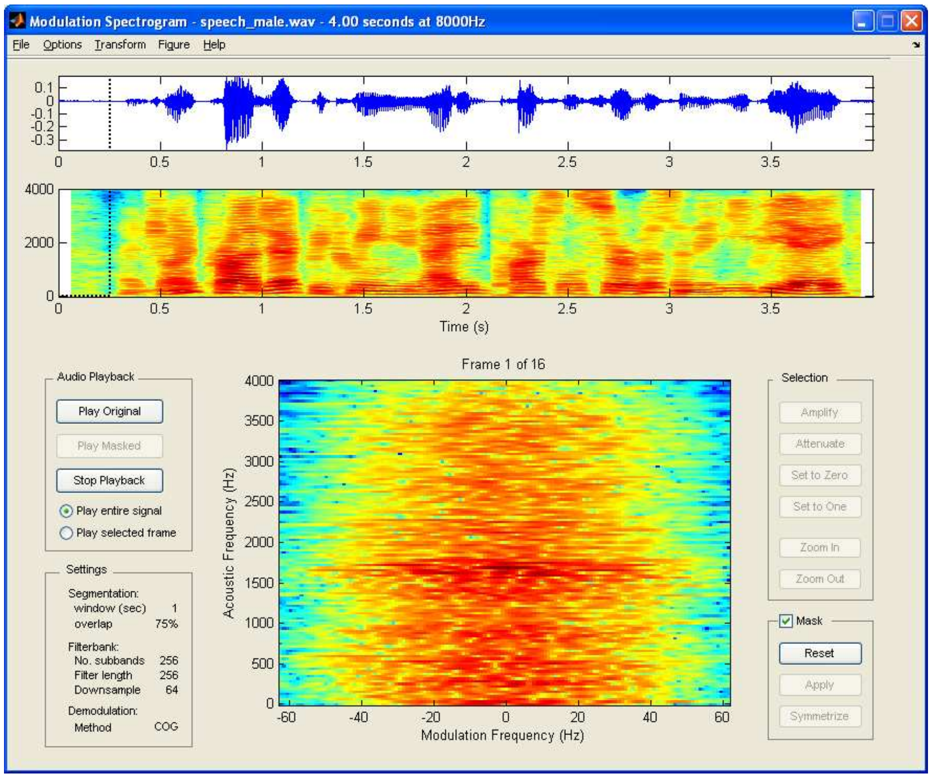
The GUI consists of three main plotting areas. In order from top to bottom they are the time-domain waveform, the time-frequency spectrogram, and finally the joint-frequency modulation spectrum. Clicking anywhere within the time-domain plot or within the spectrogram re-centers the analysis window (shown with a dotted rectangle). With each repositioning the modulation spectrum updates accordingly.

Now click `Options -> Demodulation Options`. A new dialog box should appear as seen in Figure 11. Here is where you can change the demodulation settings related to filterbank design (for coherent or non-coherent subband demodulation), pitch-detection settings (for coherent harmonic demodulation), and analysis window duration and overlap. Each of the coherent methods can also be tuned by carrier-detection parameters related to the arguments of the functions `moddecompcog.m` and `moddecompharm.m`.

Select the "Pitch-harmonic (coherent)" demodulation option and click `Ok`. You should see a new modulation spectrogram appear, as in Figure 12.

You can use the mouse cursor to make selections within the modulation spectrum as a means of applying modulation filters. As an exercise, click and drag to select an area to the right of 0 Hz in modulation frequency, and then click `Set to Zero` under the "Selection" menu. Next, click `Symmetrize` and finally click `Apply`. The result is a lowpass modulation filter operation on the signal, which should look like the example in Figure 13. Listen to the modulation-filtered audio by clicking `Play Masked`. With a sufficiently low modulation-frequency cutoff, the speech will sound slurred as high-frequency articulations are systematically removed.

Broadly speaking, modulation-frequency domain modification has potential applications in speaker separation [2] and audio compression [3], as well as general signal enhancement and separation problems. The purpose of the modulation spectrogram GUI and the associated modulation codebase is to provide tools for the continuing investigation of such fields of research.



[✓ Play entire signal] Play selected frame

[✓ Mask]

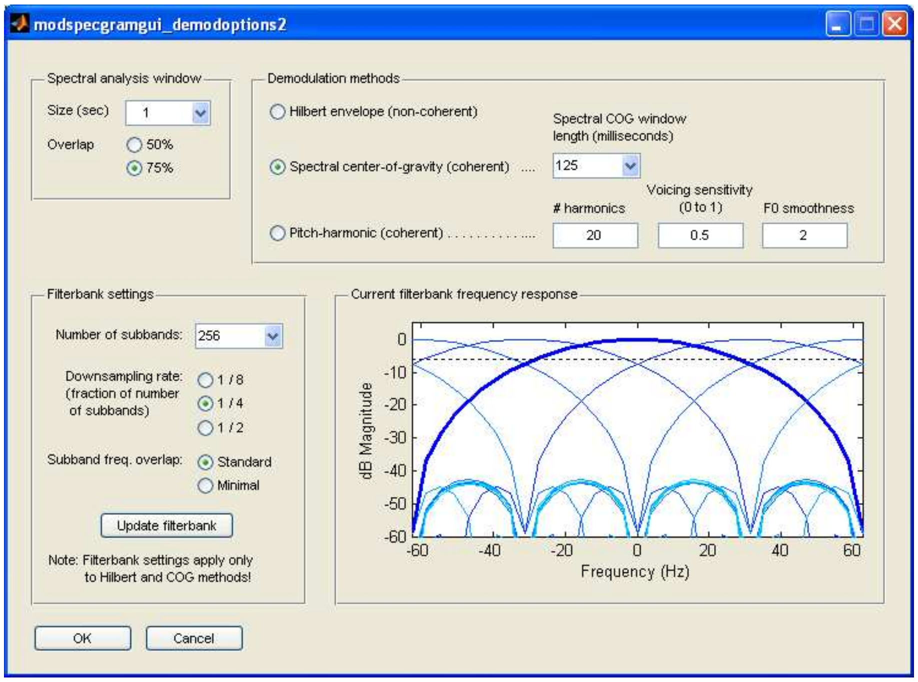
Settings

Segmentation:
window (sec) 1
overlap 75%

Filterbank:
No. subbands 256
Filter length 256
Downsample 64

Demodulation:
Method COG

Figure 10: The modulation spectrogram graphical user interface, loaded with a speech signal using default settings.


modspecgramgui_demodoptions2
Current filterbank frequency response
dB Magnitude
Frequency (Hz)
Update filterbank
OK
Cancel

Spectral analysis window

Size (sec) 1

Overlap 50% [✓ 75%]

Demodulation methods

Hilbert envelope (non-coherent)

[✓ Spectral center-of-gravity (coherent)] .... Spectral COG window length (milliseconds) 125

Pitch-harmonic (coherent) .... # harmonics 20; Voicing sensitivity (0 to 1) 0.5; F0 smoothness 2

Filterbank settings

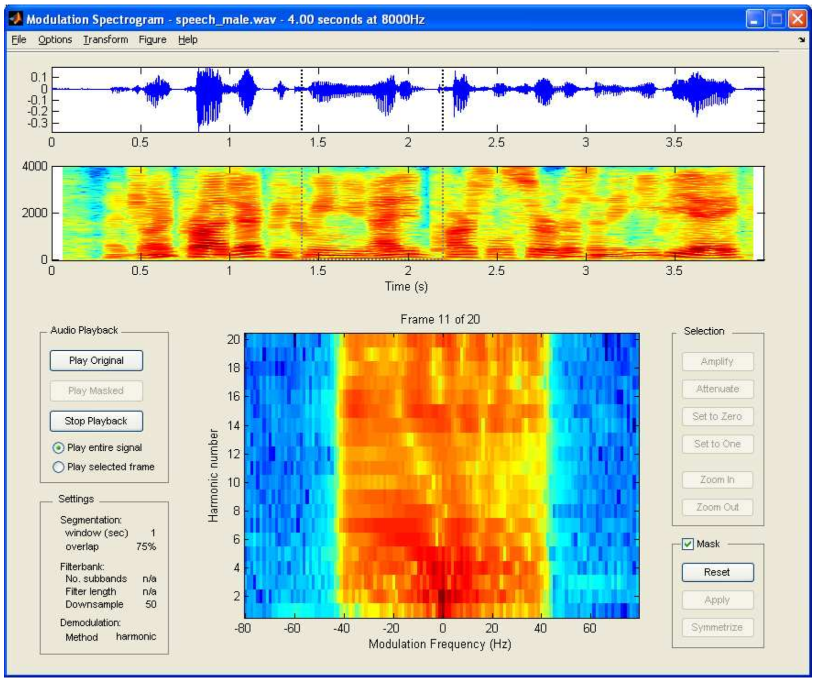
Number of subbands: 256

Downsampling rate: (fraction of number of subbands) 1 / 8 [✓ 1 / 4] 1 / 2

Subband freq. overlap: [✓ Standard] Minimal

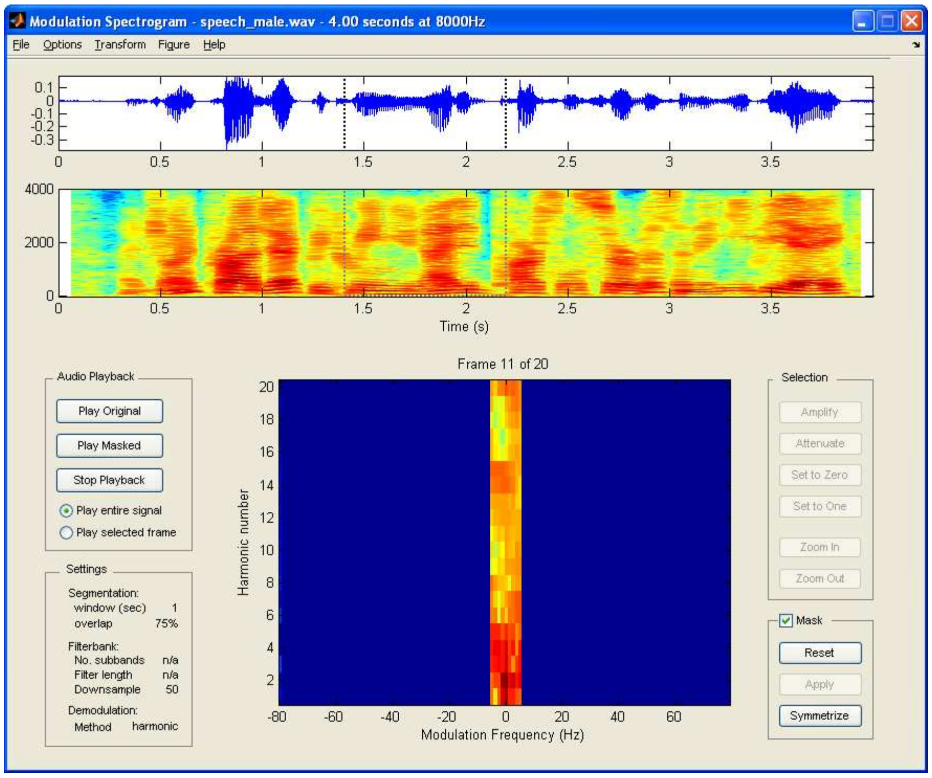
Note: Filterbank settings apply only to Hilbert and COG methods!

Figure 11: Demodulation options dialog screen.



```
Settings
Segmentation:
  window (sec)      1
  overlap         75%
Filterbank:
  No. subbands    n/a
  Filter length   n/a
  Downsample       50
Demodulation:
  Method     harmonic
```

Figure 12: Modulation spectrogram using coherent harmonic demodulation.



```
Settings
Segmentation:
  window (sec)      1
  overlap         75%
Filterbank:
  No. subbands    n/a
  Filter length   n/a
  Downsample       50
Demodulation:
  Method     harmonic
```

Figure 13: Modulation spectrogram with a lowpass "mask" or modulation filter applied uniformly to all modulators.

4 Interpretation of Complex Modulators

A consequence of bandlimited coherent demodulation is that the modulators are generally complex-valued. It is difficult at first to understand what a complex time-domain signal could mean in the "real world." However, the notion of a complex envelope is common in communications theory, and has a tangible effect on a corresponding real-valued representation. The intent of this section is to illustrate these concepts without overbearing mathematics, and to convince the reader that complex modulators are relevant to the analysis of natural signals such as speech.

The first thing to realize is that everything the Modulation Toolbox does in the complex-number domain eventually projects back onto real numbers. Each analytic subband signal $s_k[n]$ is uniquely related to a real-valued bandpass signal $x_k[n] = \mathrm{Re}\{s_k[n]\}$ by the relation

$$s_k[n] = x_k[n] + j\mathcal{H}\{x_k[n]\} \tag{18}$$

where $\mathcal{H}$ denotes the Hilbert transform. As discussed in Section 3, we assume a product model for the analytic subband

$$s_k[n] = m_k[n] \cdot c_k[n], \qquad c_k[n] = \exp(j\phi_k[n]). \tag{19}$$

For the sake of argument, let us allow $m_k[n]$ to be complex-valued and of the form $m_k[n] = i_k[n] + jq_k[n]$, where $i_k[n]$ and $q_k[n]$ are each real-valued. Euler's formula for complex exponentials leads to Rice's representation for a bandpass signal [8]:

$$x_k[n] = \mathrm{Re}\{s_k[n]\} = i_k[n] \cdot \cos(\phi_k[n]) - q_k[n] \cdot \sin(\phi_k[n]) \tag{20}$$

which is simply the superposition of two modulated carriers in quadrature. Equivalently,

$$x_k[n] = \mathrm{Re}\{s_k[n]\} = |m_k[n]| \cdot \cos(\phi_k[n] + \angle m_k[n]). \tag{21}$$

where the $\angle$ operator returns the angle of its complex argument.

When inside the cosine term in (21), the modulator phase acts as "phase-modulation" or perturbations of the fine structure of $x_k[n]$. Furthermore, the phase-modulation arises from the interaction between quadrature envelopes $i_k[n]$ and $q_k[n]$ in (20). The difference between coherent and non-coherent demodulation is in the allocation of phase between the envelope and carrier. Whereas the Hilbert envelope places all of the subband phase in the carrier, that is,

$$c_k^{Hilb}[n] \quad = \quad \exp(j\angle s_k[n]) \tag{22}$$

coherent demodulation makes the important distinction between carrier and modulator phase by defining

$$\angle s_k[n] = \phi_k[n] + \angle m_k[n] \tag{23}$$

in conjunction with (19).

The benefit of coherent demodulation is that the separation of phase can lead to bandlimited solutions for both $c_k[n]$ and $m_k[n]$, unlike the bandwidth-expanding

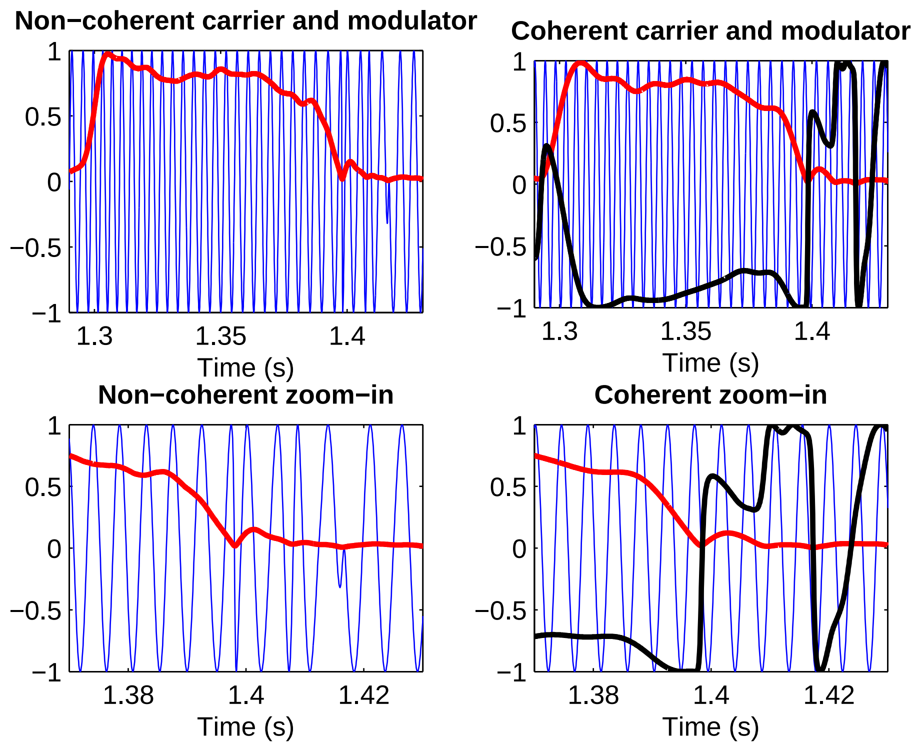


Figure 14: Non-coherent and coherent representations of amplitude modulation (red), carrier fine structure (blue) and phase modulation (black). Note how the phase inversion around 1.4 seconds is reflected smoothly in the coherent case (black curve) while manifesting as an abrupt compression in the non-coherent carrier waveform on the left.

Hilbert envelope. This results from the flexibility of Rice's representation in (20) through the use of a complex abstraction established in (18) and (19).

We can directly observe the effect of complex modulation in the time-domain, as seen in Figure 14. In both cases, the red curve is the subband magnitude, or $|m_k[n]|$. For the non-coherent case the blue curve is the real-valued carrier $c_k^{Hilb} = \cos(\phi_k[n] + \angle m_k[n])$, which shows high-bandwidth irregularities around 1.4 and 1.42 seconds. In the right-hand panels, however, the blue curve is the cosine carrier from Rice's representation, or $c_k[n] = \cos(\phi_k[n])$. The black curve is the corresponding phase-modulation term $\cos(\angle m_k[n])$, which switches polarity at 1.4 and 1.42 seconds. In the non-coherent case on the left, these phase inversions appear as a high-bandwidth frequency modulation. Conversely, the coherent case smoothly encodes phase inversion as rotations of $m_k[n]$ in the complex plane. This example reveals how the complex modulator $m_k[n]$ combines the red amplitude modulation and the black phase modulation into a single term, while maintaining bandwidth constraints in (19).

5 Toolbox Function Overview

The primary functions in the Modulation Toolbox – `modspecgramgui`, `moddecomp`, `modspectrum`, `modfilter`, and `modsynth` – make calls to a codebase of modular functions that you may want to use or modify for your own algorithms. Here is a listing of the Modulation Codebase, with brief descriptions of each function. More details can be found in the individual m-file headers.

- **High-Level Modification and Analysis**

`modspecgramgui`	:	Opens a graphical user interface that allows signal analysis and modification in the modulation frequency domain.
`modspectrum`	:	Plots the joint-frequency modulation spectrum of a signal.
`mospecgram`	:	Legacy version of the modulation spectrum. Although it is still supported, use of `modspectrum` instead is encouraged.
`modfilter`	:	Filters the modulators of a signal while keeping the original carriers unchanged.
`moddecomp`	:	Demodulates an audio signal, returning a collection of modulator and carrier signals.
`modsynth`	:	Recombines modulator and carrier signals to form an audio signal.
`modop_shell`	:	A template function for designing your own modulation analysis/modification/synthesis routines.

- **Demodulation**

`moddecompharm`	:	Coherently demodulates a signal based on a pitch estimate and an assumption of harmonic carriers.
`moddecompcog`	:	Coherently demodulates subband signals using carriers based on time-varying spectral center-of-gravity.
`moddecompharmcog`	:	Coherently demodulates a signal with COG-refined, quasi-harmonic carriers.
`moddecomphilb`	:	Incoherently demodulates subband signals using magnitude Hilbert envelopes.
`modrecon`	:	Reconstructs subband signals from modulator/carrier pairs.
`modreconharm`	:	Reconstructs a signal from modulators and harmonic carriers.
`detectpitch`	:	Detects the fundamental frequency of a signal, assuming a harmonic signal model.
`viewcarriers`	:	Overlays carrier frequencies with a spectrogram of the original audio signal for comparison.
`if2carrier`	:	Converts instantaneous frequency track(s) into complex-exponential carrier signal(s).
`carrier2if`	:	Extracts the instantaneous frequency track(s) from the phase of complex-exponential carrier signal(s).

- **Filtering**

`designfilter`	Designs a narrowband multirate FIR filter.
`filterfreqz`	Plots the frequency response of a multirate filter.
`narrowbandfilter`	Performs a multirate filter operation.

- **Filterbank**

`cutoffs2fbdesign`	Generates filterbank design parameters from a list of subband cutoff frequencies.
`designfilterbankgui`	Runs a graphical user interface for designing a filterbank with equispaced subbands and near-perfect synthesis.
`designfilterbank`	Designs a filterbank with arbitrary subband spacing and bandwidths.
`designfilterbankstft`	Designs a filterbank with equispaced subbands based on the short-time Fourier transform.
`filterbankfreqz`	Plots the frequency responses of the subbands in a filterbank design.
`filtersubbands`	Use a filterbank design to extract subband signals from an audio signal.
`filterbanksynth`	Recombine subband signals to form an audio signal.

6 Helpful Tips (Toolbox Conventions)

When using the Modulation Toolbox functions, it is helpful to keep in the mind the following:

- Demodulation functions return a modulator matrix `M`, carrier matrix `C` and instantaneous-frequency matrix `F` with the kth row corresponding to the kth time-domain signal component. There is always one modulator for every carrier.

- If you demodulate a real-valued signal `x`, then `real(M.*C)` will approximate `x`. Hence `real(M(k,:).*C(k,:))` is a real-valued kth subband signal with 0-dB gain. You can achieve more accurate re-synthesis using `modsynth` or `filterbanksynth`.

- Instantaneous frequency matrices `F` and fundamental frequency estimates `F0` are mapped from the range `[0, fs]` to `[0, 2]`, where `fs` is the audio sampling rate. This is the same convention used by MATLAB filter design functions in the Signal Processing Toolbox. To convert back to Hertz, use `F/2*fs`.

- Modulation Toolbox functions have some required and some optional input parameters. In the m-file headers, optional parameters are bracketed as in `<optionalParam>`. Optional parameters default to internally defined values if left undefined. Also, use the empty array `[]` to activate the default value for any optional parameter. For example:

  ```
  foo( x, param1, [], optionalParam2, [] optionalParam4 );
  ```

- The high-level functions `moddecomp`, `modspectrum`, `modfilter`, and `modsynth` have a "verbose" mode which outputs helpful diagnostics and plots. To activate, simply type the string `'verbose'` at the end of the input arguments. For example:

  ```
  moddecomp( x, fs, 'verbose' );
  moddecomp( x, fs, demodParams, 'verbose' );
  moddecomp( x, fs, demodParams, subbands, 'verbose' );
  ```

- By default, modulators are not downsampled. Hence the filterbank-design functions also default to a downsampling factor of 1. To improve run-time and reduce memory usage, you may want to use the downsampling options available in the toolbox functions. For subband demodulation methods, the carriers will be downsampled as well. For harmonic demodulation, the carriers will remain at the original audio sampling rate.

- The "harmonic reconstruction" function `modreconharm` can actually remodulate any arbitrary modulator array `M` with some carrier array `C`. Both arrays must have the same number of rows.

- Bandwidth is measured between -6 dB cutoff points for bandpass and lowpass filters. So a lowpass filter with cutoff located at B Hz will have a bandwidth of $2B$ or $4B/fs$ in normalized units.

7 Version Info

The Modulation Toolbox has gone through several revisions since its inception. The most recent release is version 2.1, following 2.0 and 1.23. Version 1.23 consisted of the modulation spectrogram GUI and the `modspecgram` function. Versions 2.x extend the functionality of 1.23 with a codebase for coherent demodulation and filtering functions.

Versions 2.1 and 2.0 are very similar algorithmically. The main difference is that 2.1 has been revised substantially with high-level demodulation functions and simplified user interfaces. We recommend using the latest version of the toolbox, but users of version 2.0 should be aware that 2.1 is not entirely backwards compatible with 2.0 and may not work with your existing scripts. The following is a list of notable (but not comprehensive) changes in 2.1 since 2.0:

- There are new high-level functions for one-line demodulation operations: `moddecomp`, `modspectrum`, `modfilter`, `modsynth`.
- The `modspecgram` function is still supported but has been replaced by `modspectrum`.
- The pitch-detection algorithm in `detectpitch` has been substantially revised for increased accuracy and parameter control. The new user interface for pitch-detection is also reflected in `moddecomp` and the modulation spectrogram GUI.
- The functions `modfilt`, `modfiltdesign`, and `modfiltfreqz` have been renamed `narrowbandfilter`, `designfilter` and `filterfreqz` since they are not necessarily related to modulation.
- The `viewcog`, `viewhilb`, and `viewpitch` functions have been replaced with the more general `viewcarriers`.
- There are new functions for conversion between instantaneous frequencies and carrier signals (`if2carrier` and `carrier2if`).
- `designfilterbank` only allows a single downsampling factor across all subbands.
- The input parameters for `moddecompharm` and `moddecompharmcog` have changed order for consistency with the rest of the toolbox (particulary with `moddecomp`).
- `moddecompharm` and `moddecompharmcog` return one carrier for each modulator.
- The carrier-detection window length is a required parameter in `moddecompcog` and `moddecompharmcog`.
- `detectpitch` returns F_0 values in normalized frequency which interface directly with `moddecompharm` and `moddecompharmcog`.
- The `truncate` option in `filterbanksynth` has been removed and replaced with a `keeptransients` option in `designfilterbank` and `designfilterbankstft`.

- The Hierarchical Lapped Transform (HLT) option has been replaced by Discrete Wavelet Transforms in the modulation spectrogram GUI.
- `filterbanksynth` discards imaginary quantization error if below a heuristic threshold.
- `filtersubbands` and `filterbanksynth` can accurately process complex input signals.

8 Further Reading

The following references provide background for modulation analysis and modification as implemented by the Modulation Toolbox.

Modulation filtering theory:

- P. Clark and L. Atlas, "A sum-of-products model for effective coherent modulation filtering," *Proc. IEEE ICASSP, Taipei*, pp.4485-4488, 2009.
- P. Clark and L. Atlas, "Time-frequency coherent modulation filtering of non-stationary signals," *IEEE Trans. Sig. Process.*, vol. 57, no. 11, pp. 4323-4332, Nov. 2009.
- S.M. Schimmel, "Theory of Modulation Frequency Analysis and Modulation Filtering, with Applications to Hearing Devices," Ph.D. dissertation, University of Washington, 2007.
- S. Schimmel and L. Atlas, "Coherent envelope detection for modulation Filtering of speech," *Proc. IEEE ICASSP*, vol. 1, pp. 221-224, March 18-23, 2005.

Harmonic coherent modulation filtering:

- B. King and L. Atlas, "Coherent Modulation Comb Filtering for Enhancing Speech in Wind Noise," *International Workshop on Acoustic Echo and Noise Control*, Seattle, WA, 2008.
- Q. Li and L. Atlas, "Coherent Modulation Filtering for Speech," *Proc. IEEE ICASSP, Las Vegas*, 2008.
- S. Schimmel and L. Atlas, "Target Talker Enhancement in Hearing Devices," *Proc. IEEE ICASSP, Las Vegas*, 2008.

Modulation-frequency domain analysis and modification:

- S.M. Schimmel, L.E. Atlas, and K. Nie, "Feasibility of single channel speaker separation based on modulation frequency analysis," *Proc. IEEE ICASSP*, vol. 4, pp. 605-608, April 2007.
- M.S. Vinton and L.E. Atlas, "Scalable and progressive audio codec," *Proc. IEEE ICASSP*, vol. 5, pp. 3277-3280, 2001.

References

[1] P. Clark and L. Atlas, "Time-frequency coherent modulation filtering of nonstationary signals," *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4323–4332, Nov. 2009.

[2] S.M. Schimmel, L.E. Atlas, and K. Nie, "Feasibility of single channel speaker separation based on modulation frequency analysis," *Proc. IEEE ICASSP*, vol. 4, pp. 605–608, April 2007.

[3] M.S. Vinton and L.E. Atlas, "Scalable and progressive audio codec," *Proc. IEEE ICASSP*, vol. 5, pp. 3277–3280, 2001.

[4] S. Schimmel and L. Atlas, "Coherent envelope detection for modulation filtering of speech," *Proc. IEEE ICASSP*, vol. 1, pp. 221–224, March 18-23, 2005.

[5] P. Clark and L. Atlas, "A sum-of-products model for effective coherent modulation filtering," *Proc. IEEE ICASSP, Taipei*, pp. 4485–4488, 2009.

[6] P.J. Loughlin and B. Tacer, "On the amplitude- and frequency-modulation decomposition of signals," *J. Acoust. Soc. Am.*, vol. 100, no. 3, pp. 1594–1601, 1996.

[7] L. Cohen, P. Loughlin, and D. Vakman, "On an ambiguity in the definition of the amplitude and phase of a signal," *Elsevier Signal Process.*, vol. 79, pp. 301–307, June 1999.

[8] A. Papoulis, "Random modulation: A review," *IEEE Trans. Acoust. Speech, Signal Process.*, vol. 31, no. 1, pp. 96–105, Feb 1983.

A Spectral Center-of-Gravity (COG) Demodulation

As mentioned in Section 3, we begin with an analytic subband signal $s_k[n]$. This means that $s_k[n]$ is uniquely related to a real-valued bandpass signal $x_k[n]$ through the Hilbert transform $\mathcal{H}$ via

$$s_k[n] = x_k[n] + j\mathcal{H}\{x_k[n]\} \tag{24}$$

The modulation product model assumes that $s_k[n]$ is of the form

$$s_k[n] = m_k[n] \cdot c_k[n], \qquad c_k[n] = \exp(j\phi_k[n]). \tag{25}$$

Hence, we obtain the complex modulator via the demodulation rule:

$$m_k[n[= x_k[n] \cdot c_k^*[n]. \tag{26}$$

To find the modulator, we need only define the phase of the carrier, $\phi_k[n[$. Traditionally, the so-called *instantaneous frequency* of an oscillating signal $c_k[n]$ is the derivative of the phase. In discrete-time we approximate the derivative with a first-order difference,

$$f_k[n] = \phi_k[n] - \phi_k[n-1], \qquad \phi_k[-1] = 0 \tag{27}$$

where the carrier frequency at each time $f_k[n]$ is in radians/sample. The first-order difference is a rather crude approximation to the derivative, but it has the advantage of being invertible via the cumulative sum

$$\phi_k[n] = \sum_{p=0}^{n} f_k[p]. \tag{28}$$

From these equations we have a procedure for coherently demodulating the bandpass signal $s_k[n]$:

1. Detect the subband instantaneous frequency $f_k[n]$ (more details below),
2. Integrate via (28) to obtain $\phi_k[n]$,
3. Exponentiate to obtain $c_k = \exp(j\phi_k[n])$,
4. Demodulate via (26) to obtain $m_k[n]$.

As required by Step 1 above, we still need the instantaneous frequency of the subband. The center-of-gravity approach defines $f_k[n]$ as the average frequency of the instantaneous spectrum of $s_k[n]$ at time n [6]. Conceptually we estimate the instantaneous spectrum with a short-time Fourier transform,

$$S_k(\omega, n) = \sum_p g(p) s_k(n+p)\, e^{-j\omega p} \tag{29}$$

where $g[p]$ is a short spectral-estimation window, for example a Hamming or Hann window. Then, the center-of-gravity is defined as

$$f_k[n] = \frac{\int_{-\pi}^{\pi} \omega |S_k(\omega, n)|^2 \, d\omega}{\int_{-\pi}^{\pi} |S_k(\omega, n)|^2 \, d\omega}. \tag{30}$$

Of course, we must discretize the Fourier transform in practice by using the DFT. The short-time Fourier transform for the subband is actually

$$S_k[i,n] = \sum_p g[p] s_k[n+p] e^{-j2\pi(i/L)p}, \qquad i = 0 ... L-1 \tag{31}$$

where L is the DFT size. The corresponding center-of-gravity estimate is then

$$f_k[n] = \frac{\sum_{i=0}^{L-1} r[i] \, |S_k[i,n]|^2}{\sum_{i=0}^{L-1} |S_k[i,n]|^2} \tag{32}$$

where $r[i]$ is a weighting function that acts like the ω term in (30) while accounting for the circularity of the DFT. It is defined as

$$r[i] = \begin{cases} 2\pi i/L, & 0 \le i \le L/2 \\ 2\pi i/L - 2\pi, & L/2 < i < L \end{cases} . \tag{33}$$

B F_0 Detection for Harmonic Demodulation

In this section we summarize the most important aspects of the `detectpitch` algorithm in regards to demodulation processing.

Given an audio signal $x[n], n \in [0, N-1]$, the first step of the algorithm is to block $x[n]$ into 50-millisecond frames every 25 milliseconds,

$$x_i[n] = w[n - iR] \cdot x[n] \tag{34}$$

where $w[n]$ is a Hamming window of length $2R$ and $R = f_s/40$. The algorithm then makes a binary voicing decision for each $i \in [0, N/R - 1]$, referred to here as $v[i]$. If the ith frame contains enough signal energy (relative to an automatically signal-adaptive threshold), then the frame is considered "voiced," $v[i] = 1$, and $F_0[iR] = f[i]$. Otherwise, $v[i] = 0$ and $F_0[iR] = 0$.

Next, interpolation yields $F_0[n]$ for all n. Interpolation consists of two sub-steps. First, the unvoiced portions are filled in based on straight-line interpolation between voiced segments of $F_0[iR]$. Then, a sampling-rate conversion yields the upsampled-by-R fundamental frequency estimate $F_0[n]$. Figure 15 shows $F_0[iR]$ before and after interpolation.

In each voiced frame, the estimate $f[i]$ results from 1) peak-finding in the signal autocorrelation and 2) refinement based on fitting to a least-squares harmonic model with the fundamental frequency as the free parameter. These methods are well established in the pitch-estimation literature, and the comments for `detectpitch.m` contain references for further reading.

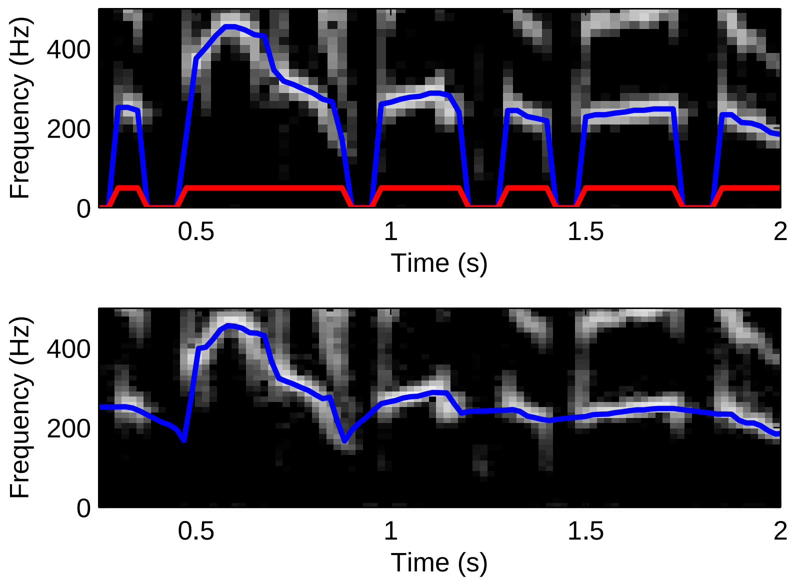


Figure 15: Top: Overlay of a signal spectrogram and the estimated fundamental frequency, where nonzero portions of the blue curve correspond to voiced regions. Bottom: The interpolated version of the fundamental frequency, with unvoiced regions filled in.