Table of Contents


```
addpath('toolboxes/malcolm_toolbox/');
addpath('resources/heli_and_boat_short');

nsegments_in = 2; % select number of segments to process for testing
testing1 = 0; %1 if testing with synthetic data, 0 if not
testing2 = 1;
```

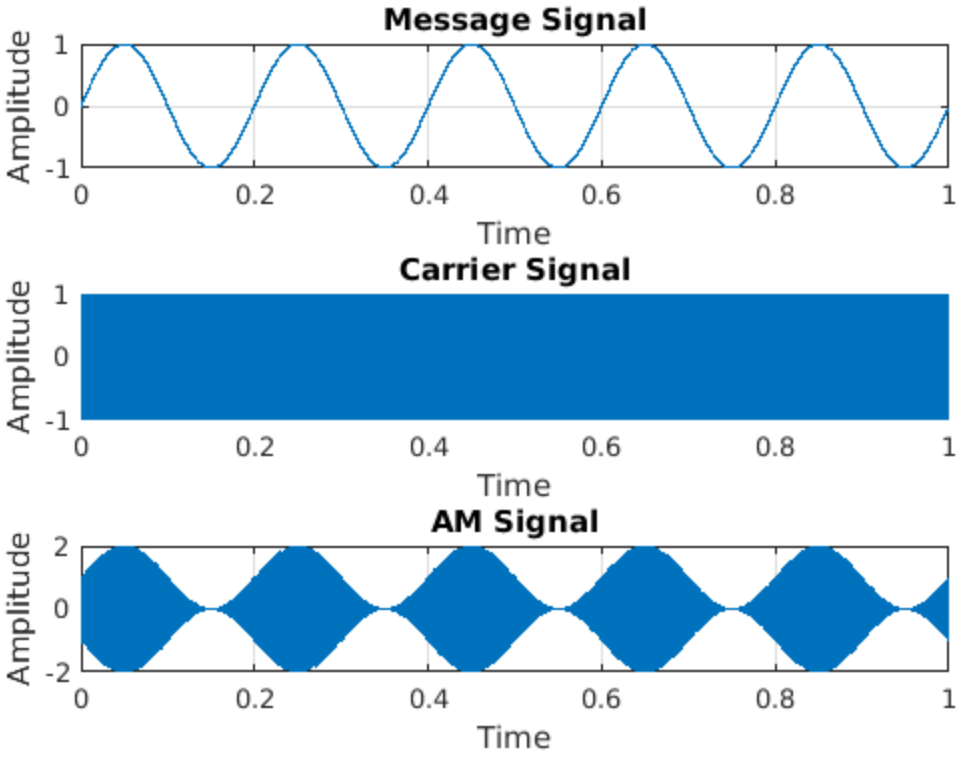
Import data

```
[x, fs] = audioread('resources/heli_and_boat_short/
heli6_short.wav'); %assume 44.1kHz
%[x, fs] = audioread('resources/Cessna.wav'); %assume 44.1kHz
x = mean(x,2);% col vector

% Resample to around 8KHz
x = resample(x,2,11);
fs = fs*2/11;
%x = resample(x,1,2);
%fs = fs/2;
xlen = length(x);
```

Test Input

```
if(testing1)
    % x: sine wave that changes its frequency linearly
    clear all;
    fs = 1000;
    t = 1:1/fs:10;
    f = 2+sin(t); %frequency oscillation between 1 and 3 Hz
    x = sin(2*pi*cumsum(f)/1000);
    figure;
    plot(t,x);
    title('test input');
    xlen = length(x);
    nsegments_in = 20;
elseif testing2
    % http://www.circuitsgallery.com/2012/07/matlab-code-for-
frequency-modulation-fm.html
    clear all;
    fm = 5; %message frequency
    fc = 800; %carrier frequency
```

```
    mi = 10; %modulation index

    fs = 10000;
    t = 0:1/fs:1;
    t = t(1:fs);
    m = sin(2*pi*fm*t);
    subplot(3,1,1);
    plot(t,m);
    xlabel('Time');
    ylabel('Amplitude');
    title('Message Signal');
    grid on;

    c=sin(2*pi*fc*t);
    subplot(3,1,2);
    plot(t,c);

    xlabel('Time');
    ylabel('Amplitude');
    title('Carrier Signal');
    grid on;

    x = (1+m).*c;%Frequency changing w.r.t Message
    subplot(3,1,3);
    plot(t,x);

    xlabel('Time');
    ylabel('Amplitude');
    title('AM Signal');
    grid on;

    xlen = length(x);
    nsegments_in = 20;

end
```



Construct final window

```
ham_t = .25; %250 ms duration window
ham_N = floor(ham_t*fs);
w = hamming(ham_N);
wshift = 4; %4hz
exp_modulator = exp(1j*2*pi*wshift/fs.*(0:ham_N-1)); %mod by 4 hz *
 2pi
exp_modulator = exp_modulator.';
w = w.*exp_modulator;
```

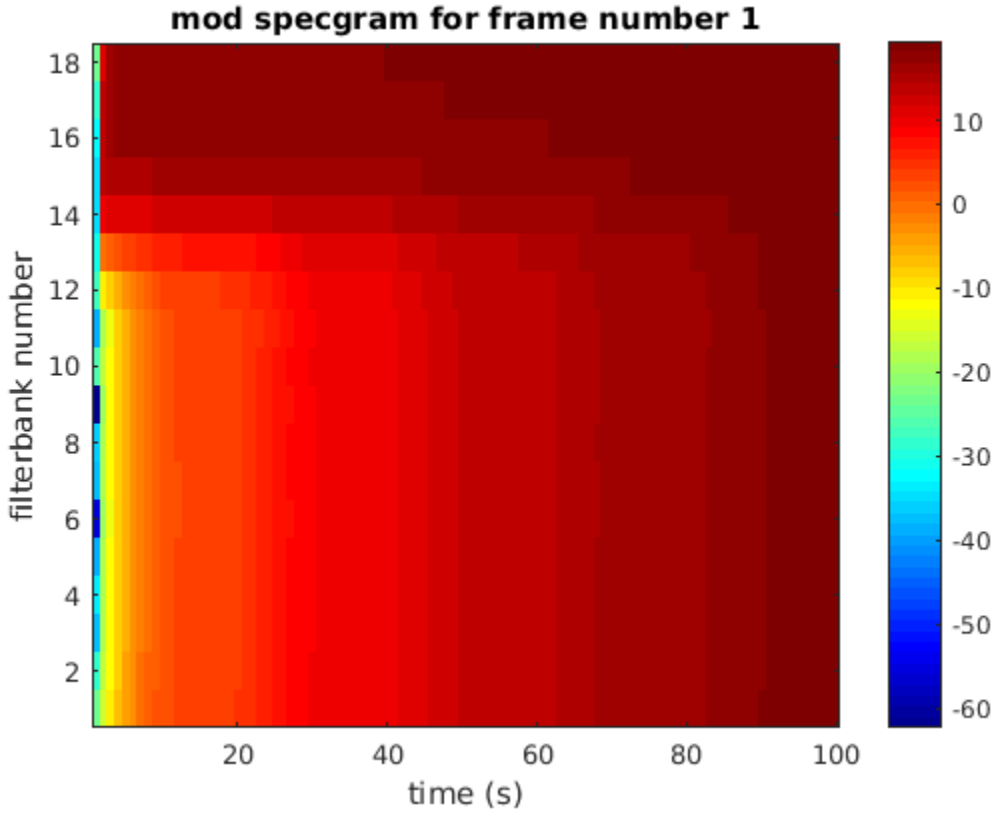
Bandpass using Gammatone Filterbank

```
% Make the center frequency vector
numChannels = 18;
lowFreq = 100; %?
fcoefs = MakeERBFilters(fs,numChannels,lowFreq);
chann_width = (fs/2-lowFreq)/numChannels; %??? not linearly spaced so
 wrong
%LOW_CF = 200;
%HIGH_CF = 4000;
%NUMCHANS = 18;
%CFS = iosr.auditory.makeErbCFs(LOW_CF,HIGH_CF,NUMCHANS);
```

Segment the data as needed (nonoverlapping)

```
segmentlen = fs;
nsegments_total = floor(xlen/segmentlen);

nsegments = min(nsegments_in,nsegments_total); % for testing

start_pos = 1;

% Operate on each time segment
for segmentind = 1:nsegments
    end_pos = start_pos + segmentlen - 1;

    x_segment = x(start_pos:end_pos);

    BM =  ERBFilterBank(x_segment, fcoefs); %operate on every col
    BM = BM.';

    for channum = 1:numChannels

        % calculate envelope and downsample
        envt = envelope(BM(:,channum)); %operate on every col
        envt = downsample(envt, 100);

        % normalize
        envt = envt./mean(envt);

        % bp filter
        bp_sig = 20*log10(abs(filter(w, 1, envt)));

        % threshold
%         bp_sig(bp_sig>0) = 0;
%         bp_sig(bp_sig<(-30)) = -30;

        out_chann(:,channum) = bp_sig;
    end

    out(:,:,segmentind) = out_chann.';

    start_pos = start_pos + segmentlen;

end


for segmentind = 1:nsegments
    figure;
    data = out(:,:,segmentind);
    imagesc(data);
    title(['mod specgram for frame number ' num2str(segmentind)]);
    ylabel('filterbank number');
    xlabel('time (s)');
    axis xy; colormap(jet);
    colorbar;
```

```
%     SPACING = (HIGH_CF-LOW_CF)/NUMCHANS;
%     yticklabels = LOW_CF:SPACING:HIGH_CF;
%     yticks = linspace(1, size(data,2), numel(yticklabels));
%     set(gca, 'YTick', yticks, 'YTickLabel', yticklabels);
end
```



Published with MATLAB® R2017a