ELECENG 3TR4 Lab 2: Amplitude Modulation

Aaron Pinto
pintoa9

Raeed Hassan
hassam41

March 2, 2021

Message Signal

The message signal is given by Equation 1. In the equation, $T_m = 0.0005$ s.

$$m(t) = -2\,\mathrm{sinc}(t/T_m) \tag{1}$$

The message signal is plotted in the time and frequency domain in Figure 1. The frequency domain plot features the magnitude spectrum of the message signal. The MATLAB code used to generate the message signal and its plots are in the lab2.m MATLAB script. A portion of the MATLAB code used to generate and plot the message signal is shown in Listing 1.

Listing 1: Generating the Message Signal and Message Signal Plots

```
%message signal
global fm; % global variable so it can be used in functions
fm = 1e3;
Tm = 0.0005;
mt = -2*sinc(tt/Tm);

%% Plotting message signal (Q1)
message_signal = figure(1);
tlayout = tiledlayout(2,1);

% time domain
nexttile;
time_dom = plot(tt, mt, 'LineWidth', 2);
tim_dom_ax = gca;
set(tim_dom_ax,'FontSize',16);
xlabel('Time (s)','FontWeight','bold','Fontsize',16);
ylabel('Message m(t) (V)','FontWeight','bold','Fontsize',16);
title('Message Signal in Time Domain');
axis([-2e-3 2e-3 min(mt) max(mt)]);

% frequency domain
Mf1 = fft(fftshift(mt));
Mf = fftshift(Mf1);
abs_Mf = abs(Mf);

nexttile;
freq_dom = plot(freq, abs_Mf, 'LineWidth', 2);
freq_dom_ax = gca;
set(freq_dom_ax,'FontSize',16);
xlabel('Frequency (Hz)','FontWeight','bold','Fontsize',16);
ylabel('|M(f)|','FontWeight','bold','Fontsize',16);
title('Magnitude Spectrum of the Message Signal');
axis ([-5e3 5e3 0 max(abs(Mf))]);
```

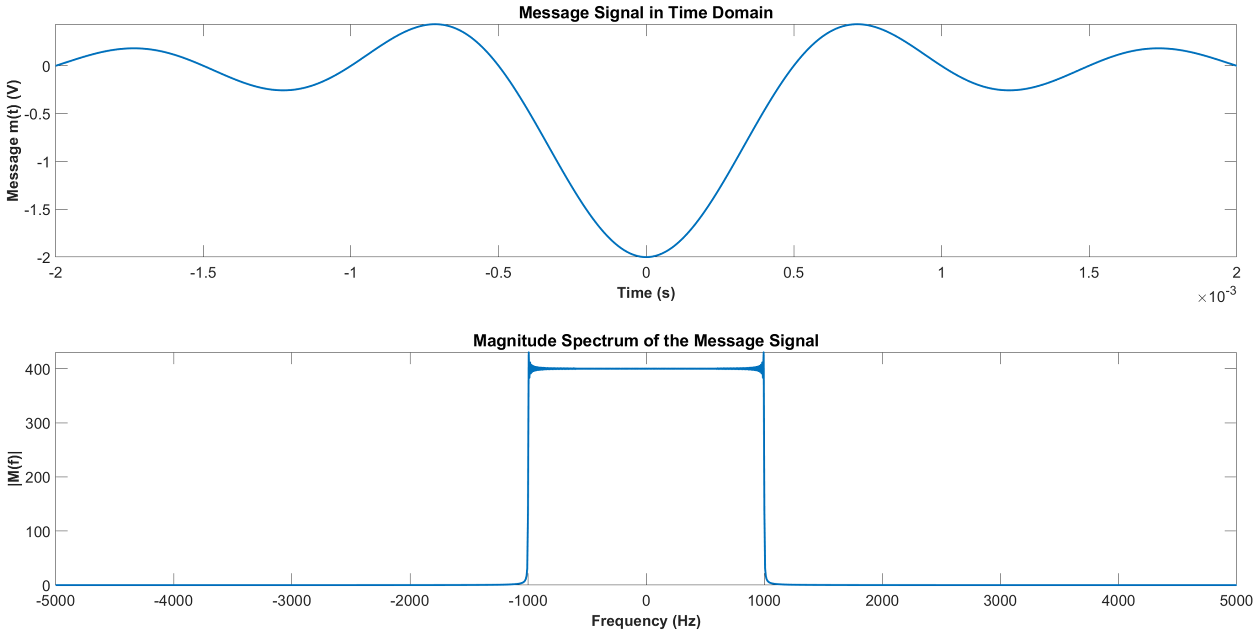

Figure 1: Message Signal Plots

The highest frequency component of the signal occurs at 995 Hz, which is the frequency bin that is closest to 1000 Hz. An analytical calculation of the magnitude spectrum of the message signal is done in Equation 2. When $T_m = 0.0005$, the rect function representing the magnitude spectrum of the message signal ranges from -1000 to 1000, which matches the magnitude spectrum we plotted in MATLAB.

$$\begin{aligned} m(t) &\rightleftharpoons M(f) \\ -2\,\mathrm{sinc}(t/T_m) &\rightleftharpoons M(f) \\ \mathrm{sinc}(t) &\rightleftharpoons \mathrm{rect}(f) \\ \mathrm{sinc}\left(\frac{1}{T_m}t\right) &\rightleftharpoons |T_m|\,\mathrm{rect}(T_m f) \\ -2\,\mathrm{sinc}(t/T_m) &\rightleftharpoons -2\,|T_m|\,\mathrm{rect}(T_m f) \\ M(f) &= -2\,|T_m|\,\mathrm{rect}(T_m f) \end{aligned} \tag{2}$$

Amplitude Modulation

The modulated signal is given by Equation 3. The carrier is given by the $c(t) = A_c \cos(2\pi f_c t)$ where $A_c = 1$ V and $f_c = 20$ kHz.

$$s(t) = A_c\,[1 + k_a m(t)] \cos(2\pi f_c t) \tag{3}$$

The MATLAB code used to generate the modulated signal is shown in Listing 2. The MATLAB function modulated_fig that is used to generate the plots for the modulated signal is shown in Listing 3.

Listing 2: Generating the Modulated Signal

```
%max of absolute of m(t)
maxmt = max(abs(mt));
%For 50% modulation
ka=0.5/maxmt;

%AM signal
st = (1+ka*mt).*ct;
```

Listing 3: Generating the Modulated Signal Plots

```
    % time domain
    nexttile;
    time_dom = plot(time_vector, signal, 'b', 'LineWidth', 2);
    % plotting envelope
    hold on
    plot(time_vector, ka*original+1,'Color','r','LineStyle', '--','LineWidth',2);
    plot(time_vector, -ka*original-1,'Color','r','LineStyle', '--','LineWidth',2);
    hold off
    legend('Modulated Signal', 'Envelope');
    tim_dom_ax = gca;
    set(tim_dom_ax,'FontSize',16);
    xlabel('Time (s)','FontWeight','bold','Fontsize',16);
    ylabel('Modulated Signal s(t) (V)','FontWeight','bold','Fontsize',16);
    title_name = "Modulated Signal in Time Domain (" + percent_modulation*100 + "% Modulation)";
    title(title_name);
    axis([-2e-3 2e-3 min(signal) max(signal)]);

    % frequency domain
    Sf1 = fft(fftshift(signal));
    Sf = fftshift(Sf1);

    nexttile;
    freq_dom = plot(freq_vector, abs(Sf), 'LineWidth', 2);
    freq_dom_ax = gca;
    set(freq_dom_ax,'FontSize',16);
    xlabel('Frequency (Hz)','FontWeight','bold','Fontsize',16);
    ylabel('|M(f)|','FontWeight','bold','Fontsize',16);
    title_name = "Magnitude Spectrum of the Modulated Signal (" + percent_modulation*100 + "% Modulation)";
    title(title_name);
```

```
    axis([-25e3 25e3 0 max(abs(Sf))]);
```

The modulated signal with 50% modulation is plotted in the time and frequency domain in Figure 2. The frequency domain plot features the magnitude spectrum of the modulated signal.

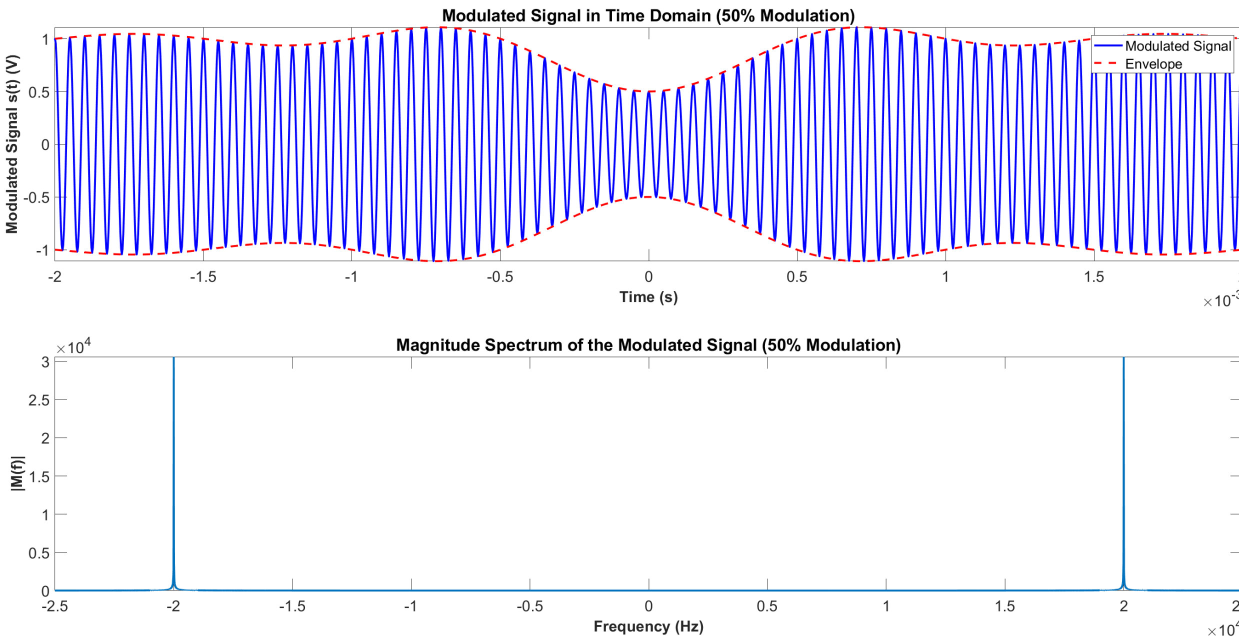

Figure 2: Modulated Signal with 50% Modulation

Amplitude Demodulation

To demodulate the signal, we pass the modulated signal through an envelope detector and then remove the DC component of the signal. The MATLAB function envelope_detector to pass the signal through an envelope detector is shown in Listing 4. The MATLAB function output_fig is to plot the demodulated plots, and a snippet of it is shown in Listing 5.

Listing 4: Envelope Detector

```
function yt = envelope_detector(signal, time_const, time_vector
   , N)
    tole = 0.1;
    yt = zeros(1,N);
    yt(1) = signal(1);
    n=1;
    for t=time_vector
        if(n > 1)
         if(signal(n) > yt(n-1))
            yt(n) = signal(n);
```

```
         else
             if((yt(n-1)-signal(n)) < tole)
               yt0 = yt(n-1);
               yt(n) = yt0;
               %time when C starts discharging
               tc = t;
             else
               yt(n) = yt0*exp(-(t-tc)/time_const);
             end
         end
        end
        n=n+1;
    end
    yt(1)=yt(2);
end
```

Listing 5: Demodulating the Signal

```
    % output of envelope detector
    nexttile;
    envelope_det = plot(time_vector, signal,'LineWidth',2);
    envelope_det_ax = gca;
    set(envelope_det_ax,'FontSize',16);
    xlabel('Time (s)','FontWeight','bold','Fontsize',16);
    ylabel('y(t) (V)','FontWeight','bold','Fontsize',16);
    title('After the envelope detector');
    axis([-2e-3 2e-3 0 max(signal)]);

    % dc removal and division by ka
    yt1 = (signal - 1) / ka;

    nexttile;
    output_signal = plot(time_vector,yt1,'r',time_vector,
       original,'k','LineWidth',2);
    legend('after DC removal','message signal');
    output_signal_ax = gca;
    set(output_signal_ax,'FontSize',16);
    xlabel('Time (s)','FontWeight','bold','Fontsize',16);
    ylabel('y1(t) (V)','FontWeight','bold','Fontsize',16);
    title('After the DC removal');
    axis([-2e-3 2e-3 min(original) max(original)]);
```

Output Signals for Time Constant $R_L C = 1/f_c$

The output signals after demodulation for when the time constant of the envelope detector is set to $R_L C = 1/f_c$ are shown in Figure 3. We can observe that the upper envelope of the output signal does follow the original message signal closely, but there is significant rippling in the output signal. The rippling in the signal would introduce a significant offset if the signal is passed through a low-pass filter to reduce the rippling, as we can see that the center of the rippled message is significantly below the upper envelope of the output signal. The significant rippling suggests that the time constant used in this scenario is too low.

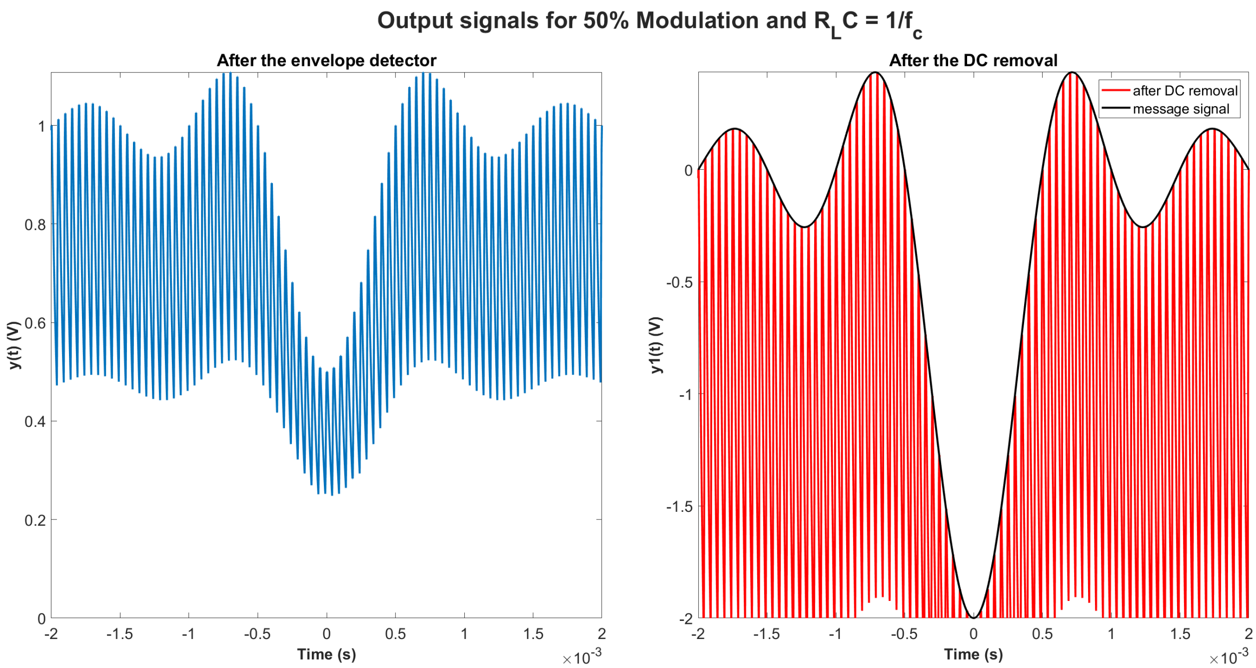

Figure 3: Demodulation Output for $R_L C = 1/f_c$

Output Signals for Time Constant $R_L C = 10T_m$

The output signals after demodulation for when the time constant of the envelope detector is set to $R_L C = 10T_m$ are shown in Figure 4. We can observe that the output signal lags behind the input signal significantly whenever there is a drop in the input signal, and the curve following the increasing portions of the input signal is very jagged. A significant portion of the input signal is lost whenever the is a drop in the input signal, and it would be impossible to accurately retrieve the input signal whenever the signal decreases too rapidly for the envelope to follow. The significant delay in following the input signal suggests that the time constant used in this scenario is too high.

Determining the Optimal Time Constant

Based on the observations of the output signal for when the time constant is equal to $R_L C = 1/f_c$ and $R_L C = 10T_m$, we can conclude that the optimal range of time constants for the

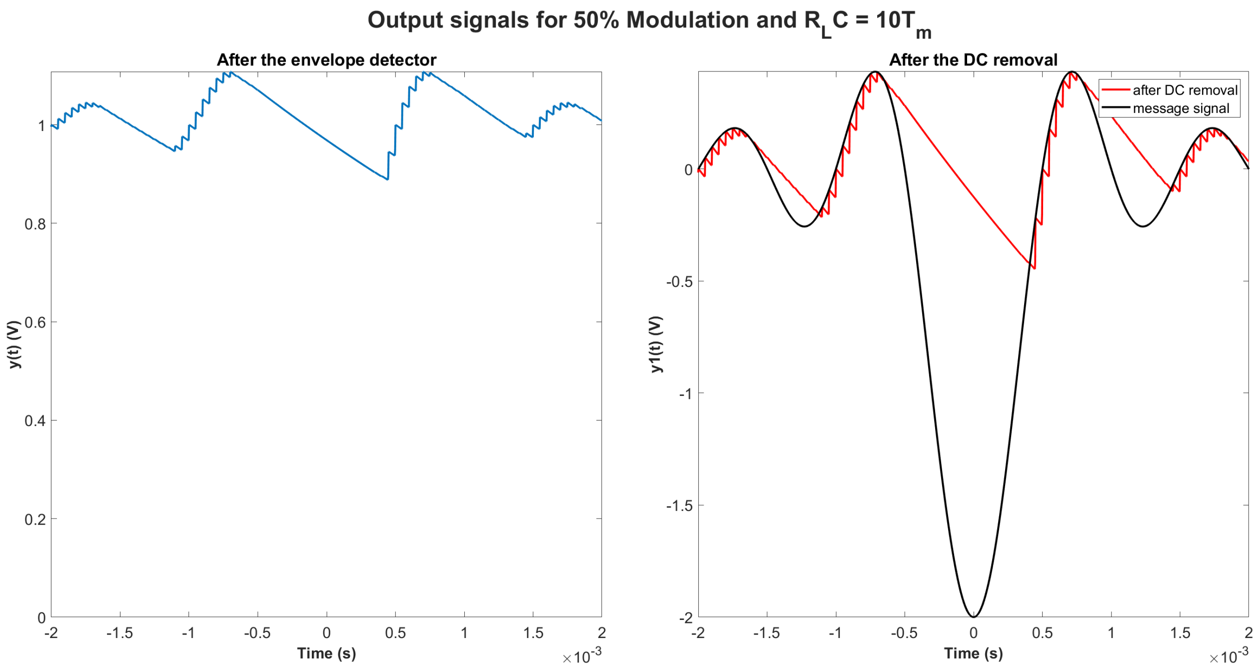

Figure 4: Demodulation Output for $R_L C = 10T_m$

envelope likely falls between the two values. We started with a time constant of T_m, which falls in the middle of the two values and found that it had followed the signal relatively well without having excessive rippling. As we explored values around T_m, we concluded that $R_L C = 1.1T_m = T_m + 1/f_c$ was the optimal value for the time constant that minimized rippling while still closely following the input signal. Time constant values from around $0.7T_m$ to $1.25T_m$ also did not differ very much from $R_L C = T_m + 1/f_c$, and could be considered to be adequate. We also passed this output signal through a low-pass filter after DC removal, using a filter generated through the built-in MATLAB function lowpass. The low-pass filter are shown in Figure 5. The output signals after demodulation for when the time constant of the envelope detector is set to $R_L C = T_m + 1/f_c$ are shown in Figure 6.

```
digitalFilter with properties:

          Coefficients: [1×57 double]

  Specifications:
     FrequencyResponse: 'lowpass'
       ImpulseResponse: 'fir'
            SampleRate: 400000
     StopbandFrequency: 3.232730000000000e+04
   StopbandAttenuation: 60
        PassbandRipple: 1.000000000000000e-01
     PassbandFrequency: 1100
          DesignMethod: 'kaiserwin'
```

Figure 5: LPF Specifications

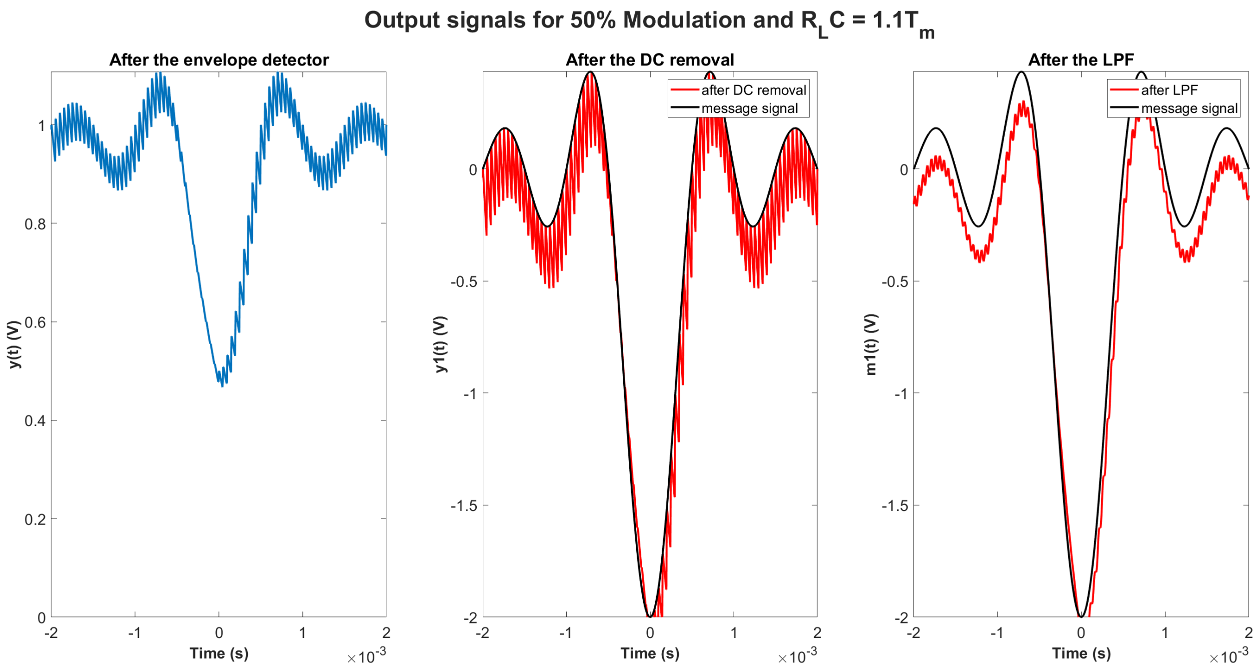

Figure 6: Demodulation Output for $R_L C = 1.1T_m$

Amplitude Modulation with over 100% Modulation

The envelope detector can only detect the positive (upper) envelope of the signal. Therefore, whenever the envelopes of the signal change signs, the envelope detector is unable to retrieve the desired input signal. This can be seen when the same message signal and carrier are modulated with 200% modulation instead of 50% modulation. The modulated signal with 200% modulation is shown in Figure 7. We can observe that the envelope of the signal (in dashed red lines) cross the y-axis multiple times, where the modulated signal in blue will also flip signs. The envelope detector will be able to realize that the sign of the envelope has changed, and will simply follow the upper envelope in those regions. Figure 8 shows the demodulated sign after envelope detection, DC removal, and low-pass filtering. We can observe that the region of the demodulated signal where the modulated signal envelope flipped signs does not match the message signal. Therefore, we can conclude that it is impossible to retrieve a signal through amplitude modulation and demodulation when there is over 100% modulation.

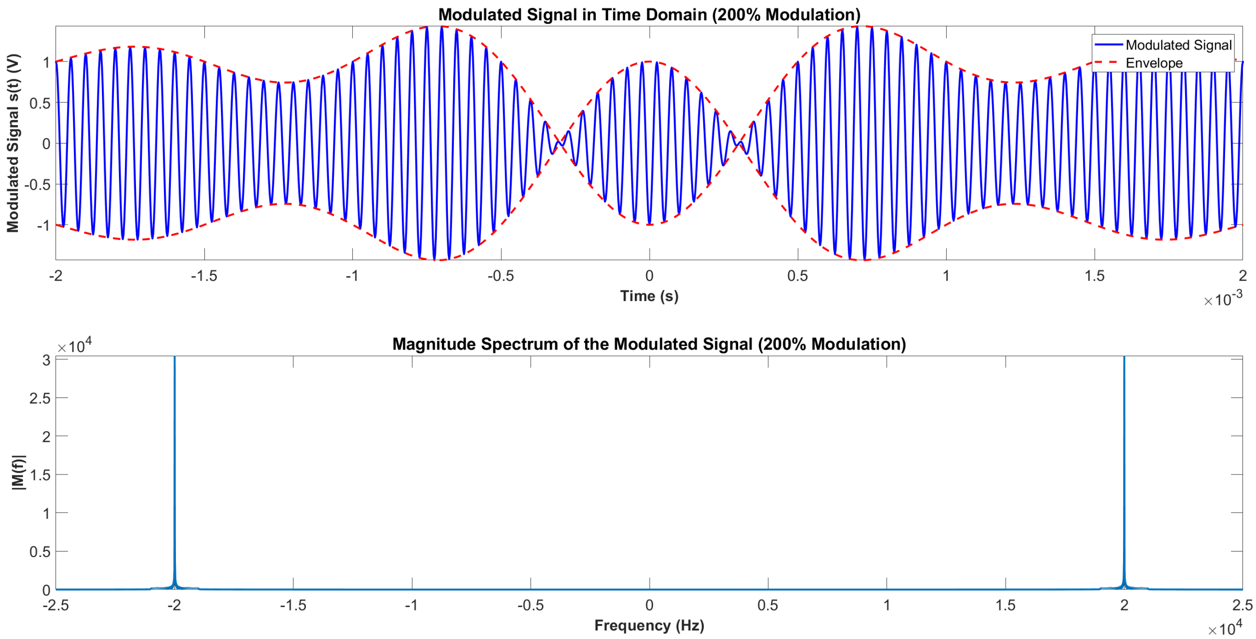

Figure 7: Modulated Signal with 200% Modulation

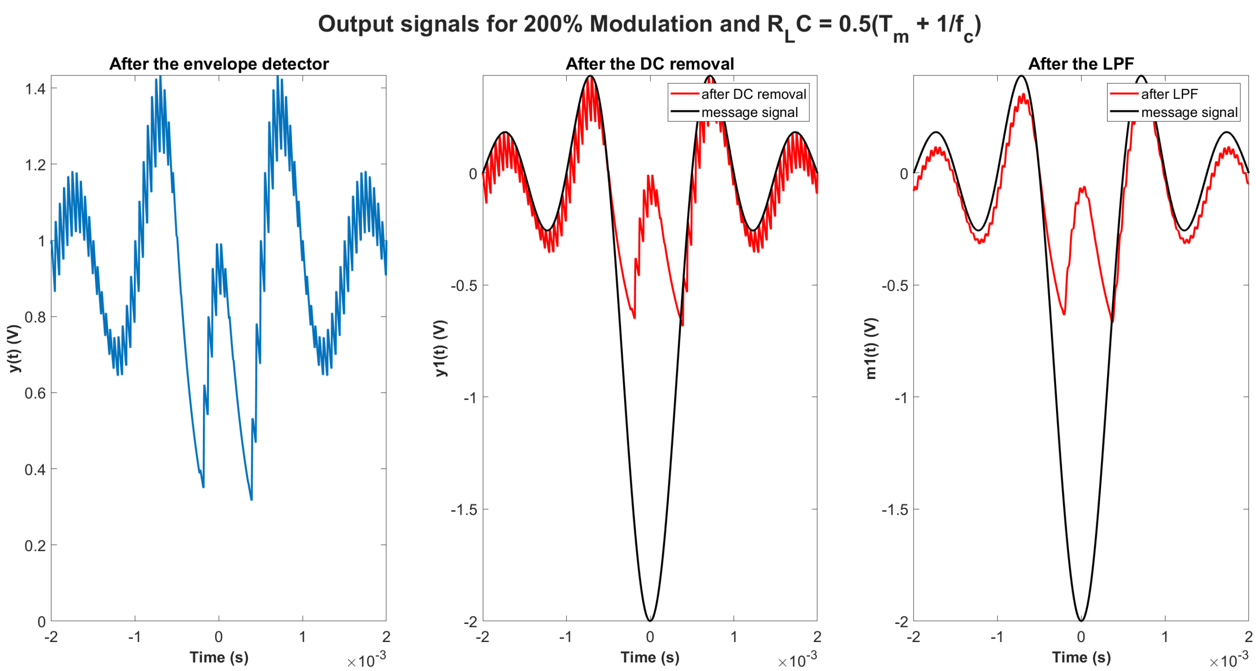

Figure 8: Demodulated Signal with 200% Modulation