Course Name: Digital Signal Processing Design

Course Number and Section: **14:332:447:01**

Dereverberation: Removing Unwanted Echoes and Reverb from Recorded Audio

Part 6 of 6: Flanger

Submitted by: Lance Darragh

Digital Audio Effects, Flanging

Flanging is an effect that uses the same filter that we used to add a single echo into the signal, only allowing the value of the delay to vary sinusoidally with time. The result is a comb filter that sweeps up and down the frequency spectrum. The comb filter has its peaks at even multiples of f_s/D, and notches at odd multiples of $f_s/2D$. By allowing the value of D to vary sinusoidally with time in the time domain we get a sweeping back and forth along the frequency spectrum in the frequency domain.

The input output relationship is now:

(33) $y(n) = x(n) + ax(n - d(n))$

where

(34) $d(n) = (D/2)*[1 - \cos(2\pi F_d n)]$, $\quad 0 \leq d(n) \leq D$

and

and F_d is a low-frequency sinusoid, say a few Hz.

Because the value d(n) can take on non-integer values, we need to either truncate, round, or interpolate the values. Interpolation methods are more accurate, but to keep the code more straight-forward we will simply round the values here. For our example we will allow the delay d(n) to range from 0 to 3 msec with a modulating frequency of 2 Hz. The code for the example is shown in Appendix A.6, and the graphs of the input and output signals are shown below.

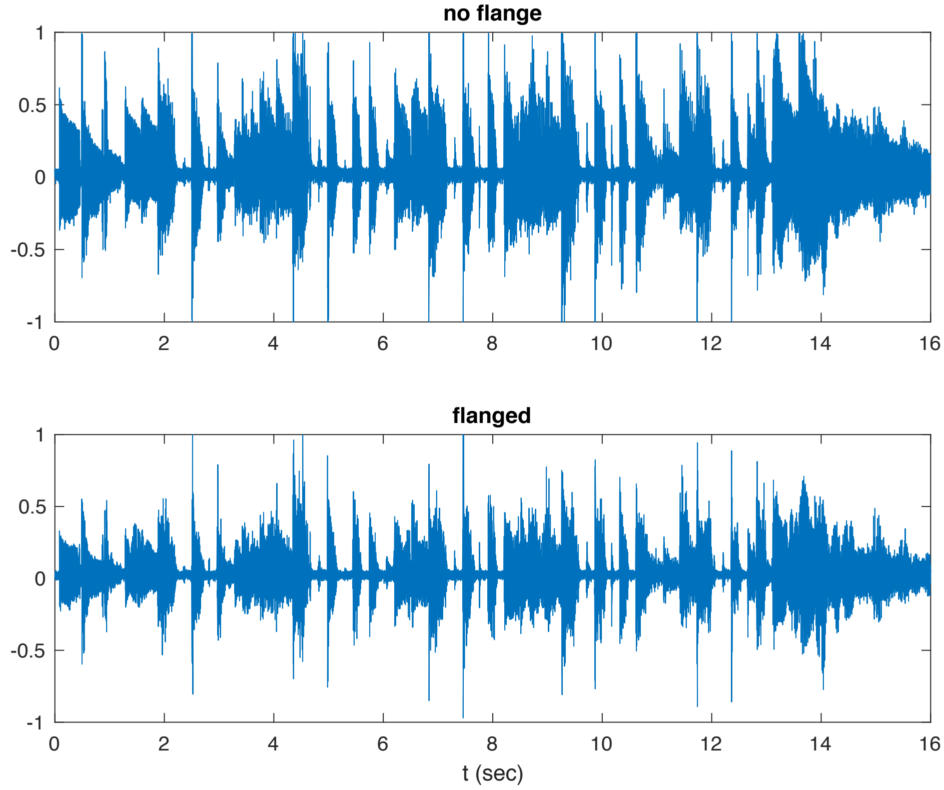

Table of Contents

Appendix A.6: Flanger

```
% Import original audio
[x,fs] = audioread('noflange.wav');

% Delay value
D = round(0.003*fs);
% For input to cosine function
F = 2/fs;
% Internal delay buffer for x(n)
w = zeros(1, D + 1);
% Delay buffer index variable
q = 1;
% Amplitude
a = 0.9;

% Determine length of input signal
[N,k] = size(x);

% Loop through input signal
for n = 1:N
    d = round((D/2)*(1 - cos(2*pi*F*n)));
    tap = q + d;
    if tap < 1
        tap = tap + (D + 1);
    end
    if tap > (D + 1)
        tap = tap - (D + 1);
    end
    y(n) = x(n) + a*w(tap);
    w(q) = x(n);
    q = q - 1;
    if q < 1
        q = D + 1;
    end
end

% Normalize y(n)
ymax = max(y);
y = y/ymax;

% Listen to results
sound(y,fs);
```

Plot Results

```
t = 1:N;
subplot(2,1,1)
plot(t/fs,x),title('no flange')
subplot(2,1,2)
plot(t/fs,y); title('flanged'), xlabel('t (sec)');
```

Output Results

```
audiowrite('FlangedFile.wav',y,fs);
```

Published with MATLAB® R2017b