EE253, S14, Term Project: Part 3 of 3 (from Buck et al., 2002)

7.3 Upsampling and Downsampling

In this exercise, you will examine how upsampling and downsampling a discrete-time signal affects its discrete-time Fourier transform (DTFT). If a discrete-time signal was originally obtained by sampling an appropriately bandlimited continuous-time signal, the upsampled or downsampled signal is the set of samples that would have been obtained by sampling the original continuous-time signal at a different sampling rate. For this reason, upsampling and downsampling are often referred to as sampling-rate conversion. Just as with sampling in continuous time, if a discrete-time signal is not sufficiently bandlimited, downsampling may introduce aliasing, which will destroy information. Individually, these operations can only increase or decrease the sampling rate by integer factors, but sampling-rate conversion by any rational factor can be achieved through a combination of upsampling and downsampling.

Basic Problems

(a). For most of this exercise, you will be working with finite segments of the two signals

$$x_1[n] = \left(\frac{\sin(0.4\pi(n-62))}{0.4\pi(n-62)}\right)^2, \tag{7.4}$$

$$x_2[n] = \left(\frac{\sin(0.2\pi(n-62))}{0.2\pi(n-62)}\right)^2. \tag{7.5}$$

Define x1 and x2 to be these signals for $0 \le n \le 124$ using the `sinc` command. Plot both of these signals using `stem`. If you defined the signals properly, both plots should show that the signals are symmetric about their largest sample, which has height 1. Analytically confirm your signals have their zero-crossings in the correct locations.

(b). Analytically compute the DTFTs $X_1(e^{j\omega})$ and $X_2(e^{j\omega})$ of $x_1[n]$ and $x_2[n]$ as given in Eqs. (7.4) and (7.5), ignoring the effect of truncating the signals. Use `fft` to compute the samples of the DTFT of the truncated signals in x1 and x2 at $\omega_k = 2\pi k/2048$ for $0 \le k \le 2047$ and store the results in X1 and X2. Generate appropriately labeled plots of the magnitudes of X1 and X2. How do these plots compare with your analytical expressions?

(c). Define the expansion of the signal $x[n]$ by L to be the process of inserting $L-1$ zeros between each sample of $x[n]$ to form

$$x_e[n] = \begin{cases} x[n/L], & n = kL, k \text{ integer}, \\ 0, & \text{otherwise}. \end{cases} \tag{7.6}$$

If x is a row vector containing $x[n]$, the following commands implement expanding by L:

```
>> xe = zeros(1,L*length(x));
>> xe(1:L:length(xe)) = x;
```

Based on this template, define xe1 and xe2 to be x1 and x2 expanded by a factor of 3. Also, define Xe1 and Xe2 to be 2048 samples of the DTFT of these expanded signals computed using fft. Generate appropriately labeled plots of the magnitude of these DTFTs. Expanding by L should give a DTFT $X_e(e^{j\omega}) = X(e^{j\omega L})$. Do your plots agree with this?

If you want to increase the sampling rate by L, you need to interpolate between the samples of $x_e[n]$ with a lowpass filter with cutoff frequency $\omega_c = \pi/L$. This filter will remove the compressed copies of $X(e^{j\omega})$ located every $2\pi/L$, except the ones centered at $\omega = 2\pi k$. The resulting spectrum is that which would have been obtained if the original bandlimited continuous-time signal had been sampled L times faster. For this reason, the combination of expansion and interpolation is often referred to as upsampling a signal.

Intermediate Problems

If the row vector x contains the signal $x[n]$, the following MATLAB command will implement downsampling by an integer factor M:

```
>> xd = x(1:M:length(x));
```

(d). Based on the DTFTs you found analytically in Part (b), state for both $x_1[n]$ and $x_2[n]$ if the signal can be downsampled by a factor of 2 without introducing aliasing. If downsampling introduces aliasing, indicate which frequencies are corrupted by the aliasing and which are not affected. If the signal can be downsampled without introducing aliasing, sketch the magnitude of $X_d(e^{j\omega})$, the DTFT of downsampled signal.

(e). Define xd1 and xd2 to be the result of downsampling x1 and x2 by 2. Define Xd1 and Xd2 to be samples of the DTFTs of the downsampled signals computed at 2048 evenly spaced samples between 0 and 2π. Generate appropriately labeled plots of the magnitudes of both DTFTs. Do the plots agree with your sketch(es) from Part (d)?

(f). A common practice is to process a signal with a lowpass filter whose cutoff frequency is π/M before it is downsampled by M. This filter is known as an anti-aliasing filter. Even if the signal is not bandlimited to π/M, the output of the anti-aliasing filter can be downsampled by M without introducing aliasing. If the original signal is not bandlimited to π/M, the anti-aliasing filter will destroy information, but usually more information is preserved after downsampling than if the signal were downsampled without being processed by the anti-aliasing filter.

In Part (e), you should have found that one of the signals suffered from aliasing when you downsampled it. You can type

```
>> h = 0.5*sinc(0.5*(-32:32)).*(hamming(65)');
```

to define h as the impulse response of a lowpass filter with cutoff $\pi/2$. Use the signal which was aliased in Part (e) as the input to this filter, and then downsample the output of the filter by 2. Use `fft` to compute 2048 samples of the DTFT of the downsampled signal. Generate appropriately labeled plots of the magnitude of the DTFT. Based on this plot, determine how much of the original DTFT is preserved.

Advanced Problems

(g). The method used to downsample x in Part (e) keeps the odd-indexed samples because the index of x starts at 1. If you change the command to keep the even-indexed samples, the resulting DTFT is significantly different for the signal which suffered from aliasing. Generate a plot of the magnitude of the DTFT that results if you keep the even-indexed samples. Let $X_{\mathrm{od}}(e^{j\omega})$ be the DTFT of the downsampled sequence that results from keeping the odd-indexed samples and let $X_{\mathrm{ev}}(e^{j\omega})$ be the DTFT that results from keeping the even-indexed samples. Prove that if the signal downsampled is finite-length, real-valued and symmetric about a sample, at least one of $X_{\mathrm{od}}(e^{j\omega})$ and $X_{\mathrm{ev}}(e^{j\omega})$ will be zero at $\omega = \pi$.

(h). For this problem, you will need the two data files `orig.mat` and `orig10k.mat`, which are in the Computer Explorations Toolbox. The file `orig10k.mat` contains a speech signal sampled at 10240 Hz. The effective sample rate of a discrete-time signal can be converted by any rational amount by the correct cascade of upsampling, downsampling, interpolation filters, and anti-aliasing filters. In this problem, you will design a system to process this speech signal to obtain the samples that would have been obtained by sampling at 8192 Hz. If `orig10k.mat` is already in your MATLABPATH, you can load the signal by typing

```
>> load orig10k;
>> who
Your variables are:
x10k
```

Determine the correct integer factors for upsampling and downsampling to convert the sampling rate from 10240 Hz to 8192 Hz. Should the converted signal have more or less samples than `x10k`? Does it matter whether you upsample or downsample first? Are both an interpolating and an anti-aliasing filter required? Implement your system, and play the original and converted signals at 8192 Hz using `sound` to confirm the converted signal. Load `orig.mat`, which contains the original speech signal sampled at 8192 Hz, and confirm that your converted signal sounds the same as the `x` in `orig.mat`.

(i) Re-design the LPF used for the sample rate converter above to be based on the Parks-McClellan algorithm. Explore the possibility of a computationally more efficient cascade implementation of the LPF and compare the number of multiplications required to generate each output sample for the direct and cascade implementations. Plot the overall frequency response of the direct and cascade LPFs and compare. In your designs, you may assume A_p = 1 dB and A_s=60 dB.