fSan Francisco State University

Engineering 451

Laboratory #7 - Upsampling and Downsampling

PURPOSE

The purpose of this laboratory is to design and implement a routine that will resample (upsample and/or downsample) a sequence.

PROCEDURE

To prepare for this laboratory, please follow these links as we consider three cases:

- Downsampling
- Upsampling
- Resampling.

ASSIGNMENT

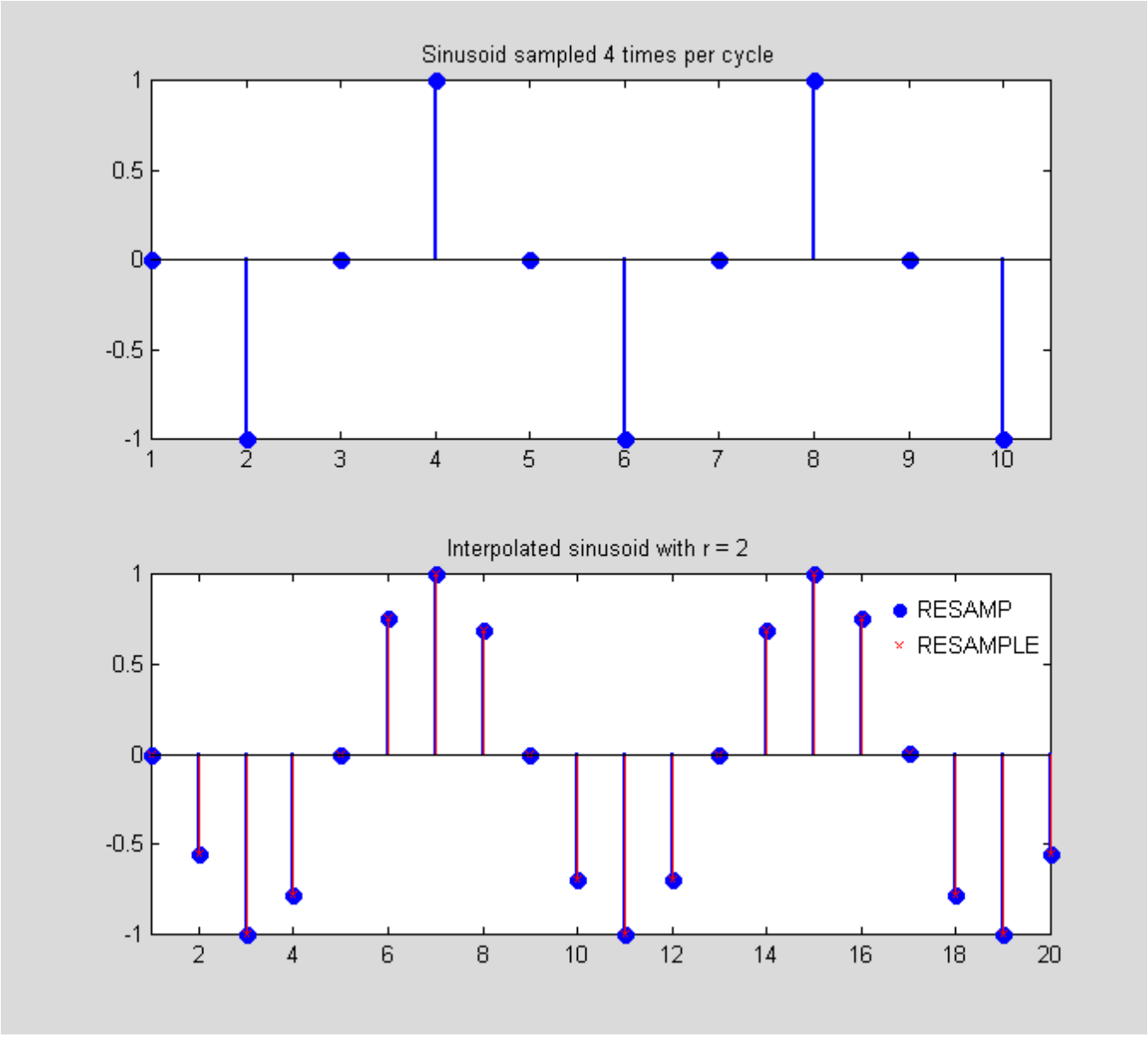
1. Answer in writing all the questions in the "Some questions" sections of the Downsampling and Upsampling links, above.
2. Write a MATLAB function, named `resample`, which has the following first lines:

```
function y = resamp(x, r)
% RESAMP Resample an input sequence x by a factor of r
%          to produce an output sequence y by a combination
%          of upsampling and downsampling.
%          For example,
%             y=resamp(x,1.5);
%          will upsample x by 3 and downsample by 2.
```

   - `X` is just an plain-vanilla array, not one of our Matlab sequence structures.
   - You may want to look at the MATLAB function `rat`, which tries to express a rational number (such as the resampling parameter, *r*) as the ratio of integers.
   - You may use the MATLAB functions `fir1, filter` and/or `kaiser`, to make your Kaiser filter only.
   - Obviously you may not use the MATLAB `resample`, `decimate`, or `interpolate` functions. However, you should feel free to use these functions to check your `resamp` function. You should read about making the Kaiser filter here.
   - You will want to make sure that the *bandwidth* and *gain* of your Kaiser filter are correct, given the value of your resampling parameter, *r*.
   - You may use MATLAB's `conv` routine if you wish.
3. Test your `resamp` against MATLAB's `resample` function. Since MATLAB's resample function may use a different lowpass filter, we can't simply compare point by point.

However, we can look at the output of the two functions for a simple sinusoidal input to see that it does the right thing. Here is a small program called `test_resamp` to do just that. What `test_resamp` does is to take an input sequence and plot the first few points resampled by a given factor using both Matlab's `resample` function and your `test_resamp` function. To use `test_resamp`, just put it in the directory with your `resamp` function, and run it. For example, to test upsampling of a little sin sequence by a factor of two, just do this:

```
» x = -sin(2 * pi * (0:10) / 4);
» test_resamp(x, 2);
```



The output of your `resamp` function is in blue and the output of MATLAB's `resample`

function is in red. They should match very well. You can do `help test_resamp` to find out the other arguments of the function.

4. Test your `resamp` function by upsampling or downsampling a segment of speech. You can get this sample in several ways:
    - Take my speech sample, `seashell.wav`, which we played earlier.
    - Find another `.wav` file on the net
    - Record your own `.wav` file.
5. Once you have a `.wav` file, read the data into your program using MATLAB's `wavread` function. .
6. Resample *x* to some other, non-integer rate, using your `resamp` function:
   ```
   » y = resamp(x, 1.5);
   ```
7. To check that you resampled correctly, playback the resampled signal using the `soundsc` function at the new sample rate, for example:
   ```
   » soundsc(resamp(x, 1.5), 1.5 * fs);
   ```

   The resampled function should sound very similar to the original function, at least if it is upsampled.
8. You should download the following file and put it in your directory : lab7.m. Then type `publish lab7`, print the output, and submit it.
