

13.8 Project: Changing the Speed of Sound

Old-timers from the days before compact disks may recall the effects of playing a record at too fast or too slow a speed. We can achieve the same audible effect by using a faster or slower playback frequency. For example, here’s Winston Churchill (recorded in the file `fh.wav`):

```
>> [x, fs, bits] = wavread('fh.wav');
>> sound(x, fs)
```

and here he is again, sounding like a fast-talking, helium eater:

```
>> sound(x, fs*1.5)
```

and again in slow motion:

```
>> sound(x, fs*0.6)
```

Alternatively, we could change the duration of a signal by deleting samples or interpolating samples; for example, taking every n th sample and playing it at the original sampling frequency:

```
>> y = x(floor(1:1.6:length(x)))
>> sound(y, fs)
```

It’s sometimes desirable to be able to speed up or slow down a recording without changing the quality of the sound. In this project, you are to write a function `speedsound` that takes three arguments: a sound waveform, a sampling frequency, and a fraction that indicates how long the new sound should be relative to the old one. For example,

```
>> newx = speedsound(x, fs, .8)
```

should produce a waveform that sounds just like `x` but plays back in only 80% of the time.

Here are some hints:

- To speed up the sound, rather than deleting samples that are evenly spaced throughout the waveform, delete entire segments of perhaps 50 to 100 ms in length. This is an important time scale for human vocal sounds. You can calculate the number of samples in such segments from the sampling frequency `fs`.
- To slow down the sound, insert segments of a similar length. You can construct the inserted segments by copying from the sound just before the insertion point.
- For a better quality of sound, you may want to delete quiet segments of the waveform (e.g., the space between words). Similarly, you can insert quiet segments between words to lengthen the sound. You can identify those segments that are quiet by using the standard deviation (`std`) of the samples in the segment.

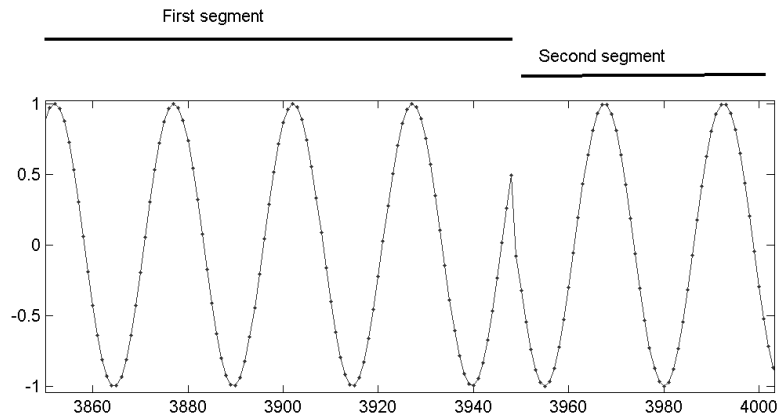


Figure 13.11. Two segments of a pure-tone sine signal have been concatenated with a mismatch between the endpoints of the segments.

For an even better quality of sound, avoid concatenating segments whose endpoints are very different. For example, Figure 13.11 shows a transition between two segments whose endpoints don’t match. Such sharp transitions produce a “pop” in the sound. When such pops occur frequently, the sound picks up a noisy, metallic quality.