Perception and Multimedia Computing

Fourier Analysis in R

Thursday 30 January 2014

This lab will give you hands-on practice using the FFT in R.

PART 1. QUICK EXPLORATION OF FFT AND SPECTRA

0. If you haven't already installed the signal or tuneR libraries, install them now using
`install.packages("signal")`
and
`install.packages("tuneR")`

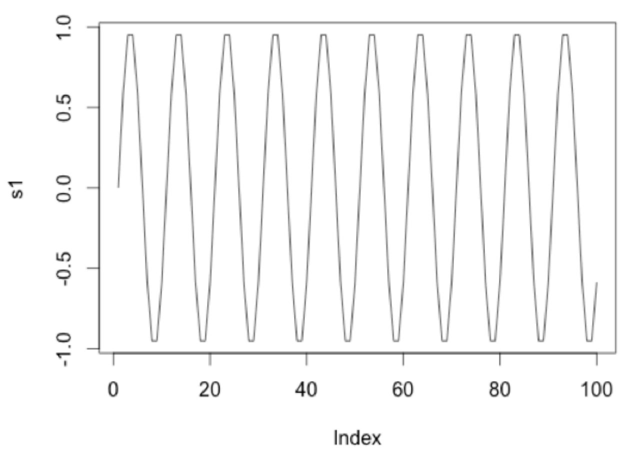
Load these libraries (using `library(signal)` and `library(tuneR)`).

1. If you didn't make it to Part 3 of last week's lab, complete the rest of this section (Part 3 from last week is repeated below). Otherwise, go on to Part 2.

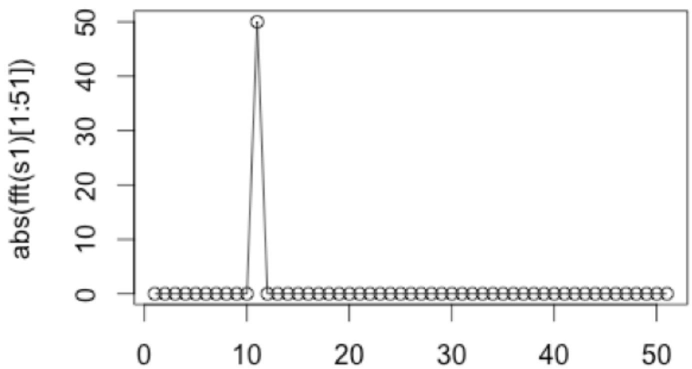
As discussed in class, the magnitude spectrum of a signal will contain peaks around any frequency component present in the sound. The spectrum is typically plotted with frequency along the x-axis, from 0 Hz to the Nyquist frequency (1/2 the sample rate).

For example, the magnitude spectrum for the following signal will look roughly like this:

s1: Sinusoid at 10 Hz, sampled at 100Hz:



The first 51 bins of its spectrum, computed over a 1-second (100-sample) analysis frame, drawn with type="o" for better visibility:



a. For each of the following signals, sketch out by hand what you think the magnitude spectrum will look like:

i. $y1 = \sin(2\pi*300*t)$
ii. $y2 = \cos(2\pi*300*t)$
iii. $y3 = \sin(2\pi*300*t + \pi)$
iv. $y4 = \sin(2\pi*400*t)$
v. $y5 = \sin(2\pi*300*t) + 0.5\sin(2\pi*400*t)$

b. The following code constructs the signal y1 for one second, using a sample rate of 1000 Hz, computes its FFT, and plots the magnitude spectrum between 0Hz and the Nyquist rate:

```
t <- seq(0,1,1/1000)
y1 <- sin(2*pi*300*t)
Y1 <- fft(y1);
plot(abs(Y1)[1:501], type="h")
```

i. Run this code and examine the plot. Does it match your expectations?
ii. Do the same for y2, y3, y4, and y5.
iii. What do you notice about the spectra for y1, y2, and y3? What does this tell you about how the phase of a waveform impacts the magnitude spectrum?
iv. What do you notice about the spectra for y1, y4, and y5? What does this tell you about the spectra of sums of signals?

PART 2. LOOKING MORE CLOSELY AT THE FFT

1. a. Construct a variable y1 to represent the following signal, sampled at 1000 Hz, **for exactly 1000 samples** (i.e., your t variable should range from 0 to 0.999 in increments of 0.001):

$y1 = \sin(2\pi*100*t) + 0.5\cos(2\pi*200*t) + 0.25\sin(2\pi*250*t + \pi/3)$

b. Now compute Y1 to be the 1000-point FFT of y1, computed for the first 1024 samples of y1:

```
Y1 <- fft(y1[1:1000])
```

c. Plot the magnitude spectrum of Y1, using the `abs()` function to calculate the magnitude of each complex value of Y1:

```
plot(abs(Y1), type="h")
```

d. Verify that the magnitude spectrum is symmetric around the Nyquist rate.

e. Using your knowledge that the sample rate is 1000 Hz, and your knowledge of how y1 was constructed, compute the exact bin indexes for which you expect a magnitude significantly above 0. Check that these bins do, in fact, contain the three highest magnitudes in the range of frequencies between 0 and the Nyquist rate. (In particular, be sure to check them against their neighboring bins, and visually check that their magnitudes are in the same range as what you see on the plot.)

f. Plot the phase of each value of Y1 using the following code:

```
plot(Arg(Y1), type="h")
```

Notice that this is **very different** from what I told you the phase would look like in class… Most bins of Y1 have negligible frequency content (very, very near 0 in magnitude), but that does not mean their phase at those frequencies is 0. In fact, the phase you see looks like random noise.

g. Let's apply a fix to the spectrum to take all the bins whose magnitude is very near 0 and set them to be **exactly equal to 0**. Run the following code:

```
for (i in 1:length(Y1)) {
    if (abs(Y1[i]) <1) {
         Y1[i] <- 0
     }
 }
```

h. Now try plotting the phase of Y1 again. What is the phase of the three frequencies with non-zero magnitude? How does this phase relate to the phase offsets of each component sinusoid you used to create the signal y1?

PART 2. PITCH TRACKING EXERCISE

1. The file mysterypitch.wav in
http://www.doc.gold.ac.uk/~mas01rf/is52020b/2013-14/lab13sounds.zip
contains approximately one second of a sine wave at some mystery frequency. Use the FFT to determine which frequency, to the best of your abilities.

Hints:
a. Use the tuneR library to load the file, like this:

```
library(tuneR)
w <- readWave("/PutYourPathNameHere/mysterypitch.wav")
```

b. Then use "@left" to create a new variable, samples, that holds only the samples of the left channel of the wave:

```
samples <- w@left
```

Now, you're free to take the fft of samples.

c. Don't forget that `max.which()` is an R function that will return the **index** of the maximum value of the vector passed as an argument to the function.

2. The file mysterypitches in
http://www.doc.gold.ac.uk/~mas01rf/is52020b/2013-14/lab13sounds.zip
contains approximately two seconds of a sine wave. This wave begins at one frequency and then changes abruptly to another. Use the FFT to determine, as best you can, the frequencies of the two pitches as well as the precise time at which the pitch changes.