Creative Computing II

Signals and Convolution

24th November 2009

This lab sheet covers basic signal representation through vector construction, signal visualization and the convolution of signals with simple LTI system impulse responses.

1. In Octave, construct vectors representing each of the following signals for discrete time $0 \leq k < 10$:

   (a) $x[k] = k$;
   `[0:9]`

   (b) $x[k] = \sin(0.4\pi k)$;
   `sin(0.4*pi*[0:9])`

   (c) $x[k] = \sin(0.3\pi k)$
   `sin(0.3*pi*[0:9])`

   (d) $x[k] = 1$;
   `ones(1,10)`

   (e) $x[k] = \begin{cases} 1 & k \text{ even ;} \\ -1 & k \text{ odd .} \end{cases}$
   `cos(pi*[0:9])` or `(-1).^[0:9]`

   (f) $x[k]$ = a random number between -1 and 1.
   `2*rand(1,10)-1` or `(rand(1,10)-0.5)*2`

2. Plot each of the signals from part 1 individually. Next, plot a pair of signals; try to find a method of plotting that makes *both* signals clear.

   You can get a simple plot of a signal `x` using the `stem` function: `stem(x)`, for example. However, on that plot the discrete-time index is wrong, as Octave defaults to start numbering from 1. Providing `stem` with two arguments means that the first is interpreted as the x-axis locations of the second argument: `stem([0:9],x)` in our case.

   To plot two signals intelligibly, try displacing one of them slightly and changing the colour:

```
stem([0:9], x1);
hold on;
stem([0:9]+0.1, x2, "b");
```

3. For each of the signals from part 1, construct the unit-delayed signal using

(a) directly zero-padding;
For a signal `x`*, use* `[0 x]`*.*

(b) zero-padding and the `shift` operator;
For a signal `x`*, use* `shift([x 0], 1)`*.*

(c) the `conv` operator with the unit delay kernel `[0 1]`.
For a signal `x`*, use* `conv(x, [0 1])`

Verify that your three methods give the same answer. Visualize each original signal along with the unit-delayed signal from this part.

The equality predicate in Octave is `==`*. Applying it to the answers from the three methods above checks for equality elementwise, returning a vector of* `1`*s for true and* `0`*s for false. Two vectors are equal if* all *their elements are equal, so for two vectors* `v1` *and* `v2` *we can check for equality with* `all(v1 == v2)`*.*

4. For each of the signals in part 1, investigate the action of the systems with impulse responses `[0.5 0.5]` and `[0.5 -0.5]`. Try to describe in words the effect of these two systems.

   You should have seen that the first system smooths out sharp differences between successive elements, while the second emphasises them: this will be clearest on the signals in parts 1d and 1e, which will each be sent to zero by one of the systems and left unchanged by the other (except at the edges). In essence the first system is an averaging or low-pass filter, which allows slow variations to pass through unchanged but removing fast variation; the second system is the inverse: a differencing or high-pass filter.

Other resources:

- Oppenheim, A. V., A. S. Willsky and S. Hamid, *Signals and Systems*, Chapters 1–2.
- Eaton, J. W., *The Octave Manual.* Available at `http://www.gnu.org/software/octave/doc/interpreter/`
- `http://en.wikipedia.org/wiki/Convolution`.