Problems 4.9–10

EECS 246, Prof. Michael Branicky

Problem 4.9

This problem reinforces convolution properties by making a relation to polynomial multiplication.

The convolution operation can be used to compute the coefficients of the product of two polynomials. Suppose two polynomials are given by

$$\begin{aligned} p(x) &= a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \\ q(x) &= b_0 + b_1 x + b_2 x^2 + \cdots + b_m x^m \end{aligned}$$

and that their product is

$$r(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{m+n} x^{m+n}.$$

Then the vector of coefficients $c = [c_0 \; c_1 \; \cdots \; c_{n+m}]$ can be obtained via convolution as $c_k = \sum_{i=0}^{k} a_i b_{k-i}$, *assuming that* $a_j = 0$, $j > n$ *and* $b_j > 0$, $j > m$. In Matlab, one can compute c using the following **template**:

```
a = [a0 a1 ... an];
b = [b0 b1 ... bm];
c=conv(a,b)
```

For each $p_i(x)$ below, use the association to compute the product shown, and report `a`, `b`, and `c`. **Also** report your answer as a polynomial in x.

(a) $p_1(x) = (2 + x)(1 + x)$.

(b) $p_2(x) = (x + 2)(x - 3)$, $p_3(x) = (x - 3)(x + 2)$.

(c) $p_4(x) = (x - 3)p_1(x)$, $p_5(x) = p_3(x)(x + 1)$.

(d) $p_6(x) = (x + 1)^5$, $p_7(x) = x^3(x + 1)^5$, $p_8(x) = x^{-1}(x + 1)^5$.

What do your results and the properties of polynomial multiplication have to say about the properties of convolution?

Problem 4.10

This problem explores the utility of a knowledge of convolution for signal processing applications.

Recall the camera example given in class. We modeled the camera as having an impulse response

$$h[n] = a^{|n|}, \qquad 0 < a < 1.$$

(a) Use Matlab to computationally verify our calculations of the camera's response to step input, when $a = 0.8$. You can do this by using the `conv` command. As you saw in Problem 4.8 above, however, there are some subtleties since both the sample response h and the input $x[n] = u[n]$ have infinite extent. Plot h, x, and their convolution. Take care that the axes are labeled correctly. Over what range is your approximation a good one? What happens if `N` below is made smaller?

The following code fragment will get you started:

```
a = 0.8;
N = 20;
win = -N:N;   % choose finite window size to approximate infinite signals
happrox = a.^abs(win);
x = unitstep(-N:10*N);  % make step long w.r.t. approximation window
y=conv(x,happrox);
```

(b) With the above variables defined in Matlab, type `xhat=deconv(y,happrox)`. What happened? You can type `help deconv`, but it's fairly cryptic. You have actually just used a filter to re-construct the original, incident image from that recorded by the camera.

(c) You can compute the system that performs the reconstruction using your knowledge of convolution and its properties. Let's suppose that our filter has an impulse response of $\hat{h}[n]$. Then the cascade system from incident image to reconstruction is as depicted in Figure 1.

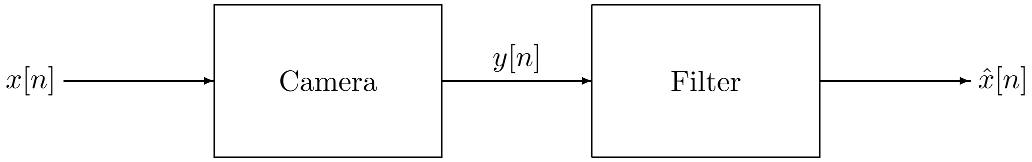

Figure 1. Image taking and processing system.

The impulse response for the entire system is $h_{tot}[n] = h[n] * \hat{h}[n]$. We would like the entire system to exactly reproduce the incident image. This will happen if $h_{tot}[n] = \delta[n]$. Why is this so?

In such a case, the constructed filter is called a *deconvolving filter.* For the $h[n]$ given above, the sample response of the deconvolving filter is

$$\hat{h}[n] = \begin{cases} \frac{1+a^2}{1-a^2}, & n = 0, \\ \frac{-a}{1-a^2}, & n = 1, -1, \\ 0, & \textit{otherwise.} \end{cases}$$

Plot $\hat{h}[n]$ for the case $a = 0.8$. Also, use Matlab to plot the convolution $h[n] * \hat{h}[n]$ (again noting valid ranges).

(d) Finally, let's examine what happens if our camera and our reconstructing filter are "out of sync". Suppose the camera *actually* has a response given by $a = 0.75$. Use Matlab to compute its output to a step. Then run that output through the deconvolving filter of part (c)—which was designed for the case a=0.8. Comment on your results.