ECE 497-3: Inverse Problems in Engineering

Homework #4

Due: Friday January 10, 2003

Motivation

In this homework, you will write MATLAB code to implement both the "brute force" and dynamic programming solution to single-input/ single-output state variable models, such as

$$\begin{aligned} \underline{x}_{j+1} &= \mathbf{M}\underline{x}_j + \mathbf{P}\underline{g}_j \\ \underline{z}_j &= \mathbf{Q}\underline{x}_j \end{aligned}$$

If you wish, you may want to write m-files for single-input/single-output systems, and separate m-files for the multi-input/multi-output systems.

1) Write an m-file to implement the dynamic programming method that produces the values in Table 1.3 of the textbook. For this $b = 1 \times 10^{-5}$. Use the values of d given in Table 1.3 (there is an error in the d values in Table 1.2). Assume the initial state is taken from Table 1.3.

2) Write a program (m-file) to implement the "brute-force" method of solving the dynamic equations to duplicate the results (i.e. estimate the g_i) from problem 1. You should get the same answers as in problem 1.

3) Now write a program (m-file) that will construct a system with added noise and solve it with both the brute force or the dynamic programming methods. For now, assume the initial state is known. The inputs to the program should be:

ng	=	the number of g's
sigma	=	the standard deviation of the noise
b	=	the value of the regularization parameter

Your code should have at least two procedures within it. One procedure (function) should do the dynamic programming, and another procedure (function) should do the brute force method.

Your program must be able to handle variations in any of these input parameters! In order to generate some data, use something like the following pieces of code so you will know the true input.

```
xold = [0;0];
x1 = xold;
Q = [1 2];
M = [0.9 2; 0.0 0.90001];
P = [2; 3];
A = 1;
```

```
%
%==========================
%  first test case
%==========================
%  tpoints = ng+1;
%  d = zeros(tpoints,1);
%  d(1) = Q*xold;
%  t = linspace(0,2*pi,ng);
%  g = sin(t);
%  for i=2:tpoints
%     xnew = M*xold + P*g(i-1);
%     d(i) = Q*xnew;
%     xold = xnew;
%  end;
%
%===========================
%  second test case
%===========================
%  t = ones(ng,1);
%  g = [0*t; t; 0*t; t; 0*t; t];
%  ng=length(g);
%  tpoints = ng+1;
%  d = zeros(tpoints,1);
%  d(1) = Q*xold;
%  for i=2:tpoints
%    xnext = M*xold + P*g(i-1);
%    d(i) = Q*xnext;
%    xold = xnext;
%  end;
%
%============================
%  add some measurement noise
%============================
%
%  d = d + sigma*randn(tpoints,1);
```

a) Show that, for a few sample points, the results with dynamic programming and the brute force method produce the same results for the same inputs.

b) Compare the true solution and the estimated solutions for the case of no added noise ($\sigma = 0$) and no regularizer ($b = 0$). You should get very close to the true input. This may be easiest to do with a plot (see part c).
c) Run both cases with various values of the input parameters. Plot the true value of g, both the dynamical programming and brute force estimate of g, and the least squares estimate of

g on a single plot (you can get the least squares estimate by setting $b = 0$ in the brute force method.) Be sure to use use the legend command so I can see which is which (the dynamic programming and brute force should lie on top of each other and will be hard to tell apart.) Try to find cases where the regularized solution is better than the least squares solutions. You may have to make b quite large.

Write-up Your write up for this homework should be neat and organized. Discuss your results and findings. You will receive extra credit for creativity and exploring different things that were not required, such as differing noise levels, different test inputs, etc. You will only learn how to solve these problems by experience!