ECE565: Estimation Detection and Filtering (Fall 2011) - Dr. Raviv Raich
Homework assignment 2 (due Oct. 25, 2011)

1. **(Constrained LS)** Consider the nonlinear input-output transfer function given by:

$$y = \sum_{k=0}^{4} a_k x^k. \tag{1}$$

You are asked to estimate the a_k's using LS from a set of points $(x(n), y(n))$ for $n = 1, 2, \ldots, N$. Assume that curve is defined over the interval $x \in [0, 1]$. You are also given that the curve contains the points $(x, y) = (0, 1)$ and $(x, y) = (1, 0)$ and that its slope at these points is zero.

   (a) Determine θ and H (for simplicity assume five points are measured).

   (b) Determine A and b in the linear form of the constraints.

   (c) Derive the constrained LS estimator for θ.

   (d) Evaluate your estimator of θ and plot the corresponding polynomial using the following Matlab code:

```
%%% Generate five (x,y) measurements
c=0;
x=(0.3:0.1:0.7)';
y=1+(c-3)*x.^2+2*(1-c)*x.^3+c*x.^4+0.05*randn(5,1);

%%% Estimate the polynomial coefficients
H=[ones(5,1),x,x.^2,x.^3,x.^4];
a_ls=inv(H'*H)*(H'*y); %%% LS
a_cls=?

%%%% uncomment one for plotting
%a_est=a_ls;
%a_est=a_cls;

%%% Plotting true polynomial, noisy measurements, and estimated polynomial
xt=(0.01:0.01:1)';
Ht=[ones(size(xt)),xt,xt.^2,xt.^3,xt.^4];
yt=Ht*[1 0 (c-3) (2-2*c) c]';
ye=Ht*a_est;
plot(x,y,'x',xt,yt,xt,ye)
axis([0 1 -0.5 1.5])
legend('Noisy measurements','True polynomial','Estimated polynomial')
```

   Fill-in the code for evaluating the constrained LS estimator and present the results. Compare to the results obtained by the unconstrained LS estimator (provided in the code).

2. **(NLLS)** You are asked to calculate the speed of a wheel based on a marker positioned R meters away from the center of the wheel. The radius of the wheel is also R and hence the velocity of the wheel is given by $v = \omega R$. The marker coordinates are given by

$$x(t) = R\cos(\omega t) \tag{2}$$
$$y(t) = R\sin(\omega t) \tag{3}$$

and are sampled at $t = \frac{T}{N}, \frac{2T}{N}, \ldots, T$.

   (a) Using NLLS find an explicit cost function that would allow to find $\hat{v}_{LS}$ for this problem.

   (b) How does this cost relate to the Fourier transform? (Hint: let $z(t) = x(t) + jy(t)$ and recall Euler formula for e^{jwt}.)

   (c) Suppose that $\omega T \ll 1$. Linearize the model, and derive the LS estimator for the linearized model.

3. **(RLS)** Recursive least squares (RLS) is a recursive implementation of the LS estimator that is well-suited for the on-line adaptive case. Here you are asked to derive the RLS algorithm in a guided fashion. As in the LS

problem we minimize $\|\mathbf{y}_n - \mathbf{H}_n\theta\|^2$ with respect to θ and the LS estimator is given by $\hat{\boldsymbol{\theta}}_n = (\mathbf{H}_n^T\mathbf{H}_n)^{-1}\mathbf{H}_n^T\mathbf{y}_n$, where

$$\mathbf{y}_n = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(n) \end{bmatrix}, \quad \mathbf{H}_n = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \vdots \\ \mathbf{h}_n^T \end{bmatrix}. \tag{4}$$

The goal is to express $\hat{\boldsymbol{\theta}}_{n+1} = (\mathbf{H}_{n+1}^T\mathbf{H}_{n+1})^{-1}\mathbf{H}_{n+1}^T\mathbf{y}_{n+1}$ based on the previous time estimate $\hat{\boldsymbol{\theta}}_n$ and the new vector $\mathbf{h}_{n+1}$ and the new sample $y(n+1)$.

(a) Show that

$$\hat{\boldsymbol{\theta}}_{n+1} = (\mathbf{H}_n^T\mathbf{H}_n + \mathbf{h}_{n+1}\mathbf{h}_{n+1}^T)^{-1}(\mathbf{H}_n^T\mathbf{y}_n + \mathbf{h}_{n+1}y(n+1)) \tag{5}$$

(b) Use the matrix inversion Lemma to show that

$$(\mathbf{H}_n^T\mathbf{H}_n + \mathbf{h}_{n+1}\mathbf{h}_{n+1}^T)^{-1} = (\mathbf{H}_n^T\mathbf{H}_n)^{-1} - \frac{(\mathbf{H}_n^T\mathbf{H}_n)^{-1}\mathbf{h}_{n+1}\mathbf{h}_{n+1}^T(\mathbf{H}_n^T\mathbf{H}_n)^{-1}}{1 + \mathbf{h}_{n+1}^T(\mathbf{H}_n^T\mathbf{H}_n)^{-1}\mathbf{h}_{n+1}} \tag{6}$$

(c) Use this result in (5) to show that

$$\hat{\boldsymbol{\theta}}_{n+1} = \hat{\boldsymbol{\theta}}_n + \frac{(\mathbf{H}_n^T\mathbf{H}_n)^{-1}\mathbf{h}_{n+1}}{1 + \mathbf{h}_{n+1}^T(\mathbf{H}_n^T\mathbf{H}_n)^{-1}\mathbf{h}_{n+1}}(y(n+1) - \mathbf{h}_{n+1}^T\hat{\boldsymbol{\theta}}_n) \tag{7}$$

(d) Let $\mathbf{P}_n = (\mathbf{H}_n^T\mathbf{H}_n)^{-1}$. Finally, show that the recursion is given by

$$\hat{\boldsymbol{\theta}}_{n+1} = \hat{\boldsymbol{\theta}}_n + \mathbf{k}_n e(n) \tag{8}$$

$$\mathbf{P}_{n+1} = (\mathbf{I} - \mathbf{k}_n\mathbf{h}_{n+1}^T)\mathbf{P}_n, \tag{9}$$

where

$$\mathbf{k}_n = \frac{\mathbf{P}_n\mathbf{h}_{n+1}}{1 + \mathbf{h}_{n+1}^T\mathbf{P}_n\mathbf{h}_{n+1}} \tag{10}$$

$$e(n) = y(n+1) - \mathbf{h}_{n+1}^T\hat{\boldsymbol{\theta}}_n. \tag{11}$$

If you were able to show (d), you have just derived the recursive formulation of LS. The advantage here is that at every time step the computational complexity is $O(m^2)$ due to the matrix update of $\mathbf{P}_n$. The total complexity of computing all the estimates $\hat{\boldsymbol{\theta}}_1, \hat{\boldsymbol{\theta}}_2, \ldots, \hat{\boldsymbol{\theta}}_n$ is therefore of the order $O(m^2 n)$, which is the same as the computational complexity of the LS estimator for only $\hat{\boldsymbol{\theta}}_n$.